Richard Crandall, Carl Pomerance

Prime Numbers

A Computational Perspective

# Простые числа

## Криптографические и вычислительные аспекты

Р. Крэндалл

К. Померанс

2 3 5 7 11 13 17 19 23 29 3

URSS

Richard Crandall, Carl Pomerance

# Prime Numbers

## A Computational Perspective

Ричард Крэндалл, Карл Померанс

# Простые числа

## Криптографические и вычислительные аспекты

Richard Crandall, Carl Pomerance

# Prime Numbers

## A Computational Perspective

Second Edition

Springer

Ричард Крэндалл, Карл Померанс

# Простые числа

## Криптографические и вычислительные аспекты

Перевод со второго дополненного
английского издания
А. В. Бегунца, Я. В. Вегнера, В. В. Кнотько,
С. Н. Преображенского и И. С. Сергеева

Под редакцией и с предисловием
В. Н. Чубарикова

МОСКВА
URSS

ББК 22.130 22.19 32.811

**Крэндалл Ричард, Померанс Карл**

**Простые числа: Криптографические и вычислительные аспекты.** Пер. с англ. / Под ред. и с предисл. В. Н. Чубарикова. — М.: УРСС: Книжный дом «ЛИБРОКОМ», 2011. — 664 с.

Простые числа дразнят воображение начинающего математика: ведь даже ребенку можно объяснить, что такое простое число, но в то же время есть ряд несложных на вид задач, над которыми лучшие умы человечества ломают головы на протяжении нескольких тысячелетий. Во второе английское издание книги «Простые числа» авторы Ричард Крэндалл и Карл Померанс включили актуальный материал из теоретической, вычислительной и алгоритмической областей. Это издание оказалось очень успешным. В нем излагаются новые результаты, которые включают AKS-тест для распознавания простых чисел, вычислительные свидетельства справедливости гипотезы Римана, быстрый бинарный алгоритм вычисления наибольшего общего делителя, неоднородные быстрые преобразования Фурье и многое другое. Авторы также приводят новые рекорды из вычислительной области и дают обзор последних результатов в теории простых чисел, например интереснейшее доказательство существования сколь угодно длинной конечной арифметической прогрессии, составленной из простых чисел, и полное решение проблемы Каталана. Во второе издание добавлены также многочисленные упражнения.

Эту книгу можно изучать на разных уровнях. Для тех, кто хочет получить общее впечатление об этой красивой науке и об основных методах работы с простыми числами, книга является прекрасным введением в предмет. Для тех же, кто хочет глубже вникнуть в подробности новейших методов вычислений с простыми числами, в книге приводится соответствующий материал, а также ссылки на обширную литературу по теме. Студенты смогут проверить свое понимание с помощью интересных упражнений, подчас занимательных и нестандартных. Наконец, для тех, кто хочет начать или углубить свои исследования по вычислительной теории простых чисел, по тексту и в упражнениях щедро разбросаны многочисленные нерешенные проблемы, которые предоставляют богатую почву для дальнейшего анализа.

Книга будет интересна студентам, преподавателям и научным работникам, специализирующимся в области теории чисел и дискретной математики, а также специалистам в области криптографии и защиты информации.

Translation from the English language edition:
*Prime Numbers. A Computational Perspective* by Richard Crandall, Carl Pomerance

Издательство «Книжный дом "ЛИБРОКОМ"». 117335, Москва, Нахимовский пр-т, 56
Формат 70×100/16. Печ. л. 41,5. Заказ № 2534.
Отпечатано в полном соответствии с качеством предоставленных материалов в ОАО «Дом печати — ВЯТКА»
610033, г Киров, ул Московская, 122 Факс (8332) 53-53-80, 62-10-36
http://www.gipp.kirov.ru, e-mail pto@gipp.kirov.ru

**ISBN 978–5–453–00016–6**
**(УРСС)**

**ISBN 978–5–397–02060–2**
**(Книжный дом «ЛИБРОКОМ»)**





9918 ID 123740
9 785397 020602

# От редактора перевода

Книга американских математиков Р. Крэндалла и К. Померанса «Простые числа: Криптографические и вычислительные аспекты» посвящена важному современному направлению теории чисел — вычислительной (или алгоритмической) теории чисел. Данное направление стало интенсивно развиваться с 70-х гг. XX в., после открытия асимметричных криптосистем Диффи—Хеллмана и RSA. Этому развитию во многом способствовал бурный прогресс в области вычислительной техники. Тем не менее, как неоднократно подчеркивают авторы, в этой книге мы находим идеи, восходящие к великим математикам прошлого — к Ферма, Эйлеру, Гауссу, Чебышёву.

Книга охватывает следующую тематику:

классическая теория простых чисел;
классические алгоритмы теории чисел;
проверка простоты (больших) целых чисел;
разложение целых чисел на множители;
дискретное логарифмирование;
эллиптические кривые;
приложения простых чисел в криптографии и других областях;
арифметические операции с большими целыми числами и вещественными числами высокой точности.

Неудивительно, что на первом месте стоит классическая теория простых чисел — как мы уже упоминали, многие идеи современных алгоритмов восходят к классикам. Кроме того, без фундаментальных результатов аналитической теории чисел, к которым относится, например, асимптотический закон распределения простых чисел, открытый Ж. Адамаром и Ш. Валле Пуссеном в 1896 г., по существу невозможно провести анализ сложности теоретико-числового алгоритма.

С другой стороны, умение вычислить теоретико-числовую величину или построить теоретико-числовой объект позволяет лучше понять, как они устроены. Это может помочь пониманию сути теоретического понятия и даже привести к новым открытиям. К примеру, Л. Эйлер был, кстати, и непревзойденным вычислителем.

Авторы отмечают, что эта двойственность помогала им в работе над книгой. Объединившись, они представили оригинальный взгляд на вычислительные и криптографические аспекты теории чисел. Многие важнейшие теоретико-числовые алгоритмы, описанные в книге, созданы ее авторами. Для книги характерна идейная ясность, простота и «многоуровневость» изложения, благодаря которой читатель сможет пройти путь от самых основ теории до самых последних результатов и проблем. Книга содержит тексты всех представленных алгоритмов (их более 100) и конкретные примеры их работы.

Таким образом, читатель, знакомый с основами теории чисел и основами программирования, освоив эту книгу, получит полное представление о современном состоянии вычислительной теории чисел.

Мы уже упоминали, что бурное развитие вычислительной теории чисел и ее активное проникновение в различные сферы жизнедеятельности, включая сферу экономики и финансов, связаны с криптографией. Читатель, интересующийся приложениями теории чисел в области криптографии, найдет в этой книге основные понятия и алгоритмы.

Подводя итог, можно сказать, что эта книга будет интересна студентам, преподавателям и научным работникам, специализирующимся в области теории чисел и дискретной математики, а также специалистам в области криптографии и защиты информации.

Работа по переводу книги на русский язык была выполнена сотрудниками механико-математического факультета МГУ им. М. В. Ломоносова — А. В. Бегунцем (гл. 1, 3), Я. В. Вегнером (гл. 2 (2.3–2.5), 9, Приложение), В. В. Кнотько (гл. 4, 5), С. Н. Преображенским (гл. 6, 7), И. С. Сергеевым (гл. 2 (2.1–2.2), 8).

В заключение хочется отметить, что предлагаемая вниманию читателей книга вне всякого сомнения служит примером того, как фундаментальные понятия арифметики находят весьма неожиданные практические применения.

Авторы включили в книгу обстоятельный список литературы на английском языке. Редактором перевода составлен дополнительный список литературы о простых числах. Он приведен в конце книги.

Редактор перевода и переводчики считают своим приятным долгом поблагодарить авторов, любезно приславших уточнения и исправления, которые были учтены при подготовке русского издания.

*В. Н. Чубариков*

# Предисловие

*Мы посвящаем этот труд нашим родителям — Дженис, Гарольду, Хильде и Леону, каждый из которых по-своему прекрасно и самобытно научил детей тому, как следует рассуждать.*

На страницах этой книги мы постарались построить связующее звено (надеемся, что даже мост) между «теорией» и «практикой» в области простых чисел. Разумеется, речь идет о теории чисел и о вычислительной практике. Созданы великие книги, рассказывающие об абстрактных свойствах простых чисел, и каждый из нас обращается к ним как к любимой классике. Однако экспериментальная сторона этого вопроса относительно молода, ведь несмотря на небеспочвенные утверждения о том, что теория вычислительных машин и систем ни в коем случае не является новой (к настоящему времени в ней уже произошли четыре или пять «революций»), необходимо принять во внимание многовековую и даже тысячелетнюю историю теоретических исследований в области простых чисел. Поэтому мы полагаем, что для научных трудов, основанных одновременно на знаменитых классических методах и современных вычислительных подходах, все еще остается место.

### Замысел и тематика этой книги

Книга сочетает по сути взаимно дополняющие области компетенции обоих авторов (Р. Крэндалл — скорее компьютерщик, а К. Померанс — скорее теоретик). Первые главы носят более теоретический характер, хотя в них и описано несколько явных алгоритмов, а алгоритмическая составляющая существенно возрастает по мере дальнейшего продвижения читателя вглубь книги. Как в теоретических, так и в вычислительных сторонах вопроса мы старались продемонстрировать наиболее современный взгляд на теорию чисел. Чего мы не делали — это не стремились к обоснованию всех тонкостей каждой детали. Это не только потребовало бы гораздо большего объема книги, но, как мы указываем в начале первой главы, можно сказать, что ни один человеческий ум не в состоянии охватить полной картины знаний о простых числах. Видимо, и команда из *двух* исследователей тоже не может обладать таким всеведением, по крайней мере именно так обстоит дело с нашей парой. Своей задачей мы сочли обеспечение ссылок на огромное число источников, содержащих подробное описание тех аспектов, которых нам не удалось раскрыть в достаточном объеме. Кроме того, совершенно ясно, что к моменту выхода книги в свет многие теоретико-числовые рекорды, отраженные в ней, будут уже побиты. На самом

деле, некоторые из них были превзойдены даже пока мы писали это предисловие. Завершая работу над книгой, мы, можно сказать, жили в том состоянии, которое инженеры-электронщики называют «состояние гонок», поскольку и через Интернет, и устно новые результаты о простых числах сообщались нам даже быстрее, чем мы могли вносить правку в эту книгу. Поэтому в некоторый момент мы приняли решение поставить точку, указав в книге адреса сайтов, на которых можно найти наиболее актуальные на данный момент результаты. Состояние гонок стало естественной частью игры особенно в последние годы, когда в дело включилась мощная вычислительная техника.

**Упражнения и проблемы для исследования**

В конце каждой главы приведены упражнения, которые обычно следуют в порядке изложения материала данной главы и бывают как очень простыми, так и чрезвычайно сложными. Для того чтобы еще раз продемонстрировать связь между теорией чисел и практикой, мы придали ряду упражнений исследовательский характер и поместили их в качестве проблем для исследования после обычных упражнений (тем не менее, в тексте мы также называем эти проблемы «упражнениями»). Нельзя сказать, что все обычные упражнения легки, однако мы старались относить вопрос к проблемам для исследования в тех случаях, когда его можно было рассматривать как часть длительного и значимого, по нашему мнению, исследования.

**Алгоритмы и псевдокод**

Мы приложили немало сил (иногда даже на грани нервного срыва), работая над текстом приведенных в книге алгоритмов. Существует точка зрения, согласно которой современное искусство «правильного» псевдокода (т.е. не исполняемого на машине, а предназначенного для чтения человеком) находится в тяжелом кризисе. В большинстве сегодняшних книг, содержащих псевдокод, преобладает несовместимость между его читаемостью и краткостью записи. По-видимому, в настоящее время невозможно достичь обеих этих целей одновременно.

В поисках равновесия в качестве базы для псевдокода нашей книги мы выбрали стиль языка C. В приложении описаны явные примеры того, как перевести на машинный язык различные блоки приведенных в тексте алгоритмов. Мы будем полагать, что наш подход к псевдокоду удался, если произойдут две вещи:

(1) программисты смогут на основе наших алгоритмов с легкостью создавать программы;

(2) все читатели сочтут изложение алгоритмов понятным.

Мы дошли даже до того, что обратились к нескольким талантливым программистам с просьбой перевести алгоритмы нашей книги в настоящие программы, тем самым до какой-то степени проверяя нашу цель (1) (тексты этих программ доступны в разделе *Mathematica* на сайте http://www.perfsci.com). Тем не менее, как было сказано выше, удовлетворяющий всех симбиоз математики и

псевдокода может возникнуть не ранее наступления той эры, когда машины станут более «человекоподобными».

**Примечание ко второму изданию**

Источников материала и стимулов для подготовки настоящего второго издания было несколько. Во-первых, наши проницательные читатели (которым мы глубоко признательны) обнаружили в первом издании разнообразные ошибки и просили прояснить ряд моментов, а некоторые из читателей даже предлагали включить в книгу новые идеи рассуждений. Во-вторых, постоянно растущие возможности вычислительной теории чисел побуждают нас осветить новые результаты. В-третьих, оба автора преподают, поэтому усовершенствование читаемых ими лекционных курсов способствовало расширению материала первого издания. Помимо исправления ошибок, более доступного изложения и обновленных (на начало 2005 г.) вычислительных достижений, во второе издание включены дополнительные алгоритмы, описанные на введенном нами языке псевдокода. Некоторые из этих алгоритмов можно смело назвать поистине захватывающими открытиями.

Примеры дополнений, связанных с вычислительными успехами

- Обновлено значение самого большого известного простого числа (на апрель 2005 г.) (см. табл. 1.2), а также данные о поиске простых чисел Мерсенна.
- Приведены последние результаты поиска простых близнецов, длинной арифметической прогрессии, состоящей из простых чисел, новые успехи в области распознавания простоты и т. д. (см., например, гл. 1 и упражнения в ее конце).
- Описаны недавние достижения в области разложения на множители, большинство из которых (но не все) связаны с субэкспоненциальными методами (см. разд. 1.1.2).
- Приведены недавние рекорды в области дискретного логарифмирования и дискретного логарифмирования на эллиптических кривых (DL и EDL, соответственно) (см. разд. 5.2.3 о задаче DL и разд. 8.1.3 о задаче EDL).
- Приведены новые границы справедливости гипотезы Римана (разд. 1.4.2).

Примеры дополнений, связанных с алгоритмическим прогрессом

- Добавлен теоретический материал и приведены алгоритмы нового AKS-метода и его усовершенствованных вариантов, которые позволяют установить простоту за полиномиальное время (см. разд. 4.5).
- Представлен новый быстрый алгоритм Бернштейна для нахождения чисел, все простые делители которых малы, в большом множестве, даже если это множество не имеет регулярной структуры, что позволяет ускорить методы решета (см. разд. 3.3).
- Описан новейший и эффективный алгоритм нахождения наибольшего общего делителя Штеле—Циммермана (см. алгоритм 9.4.7).

- Даны ссылки на новые результаты в области «промышленных алгоритмов», таких как подсчет числа точек на эллиптической кривой (см. разд. 7.5.2), алгебраические операции на эллиптических кривых, находящие применение в смарт-картах (см., например, упр. 8.6), и «гигаэлементные» быстрые преобразования Фурье (БПФ), т. е. преобразования, способные принимать на входе миллиарды комплексных элементов (конец раздела 9.5.2).
- Неоднородное БПФ, играющее растущую роль в вычислительной теории чисел, включено во второе издание в виде алгоритма 9.5.8 (см. также упр. 1.62).

Примеры достижений в теории, добавленных во второе издание

- Обсуждается новая сенсационная теорема Грина и Тао о существовании сколь угодно длинной арифметической прогрессии, состоящей исключительно из простых чисел (см. конец разд. 1.1.5).
- Рассматриваются последние результаты, связанные с гипотезой Ферма–Каталана о том, что существует не более чем конечное множество взаимно простых натуральных чисел $x^p, y^q, z^r$, для которых $x^p + y^q = z^r$ и $1/p + 1/q + 1/r \leqslant 1$. Описан также частный случай, когда одно из этих чисел равно 1: тогда существует единственное решение $8 + 1 = 9$, как следует из недавнего чудесного результата Михайлеску (см. разд. 8.4), который тем самым решил исходную проблему Каталана.

Разнообразные изменения коснулись списка упражнений. Включены новые упражнения, часто связанные с добавленными алгоритмами. Некоторые упражнения усовершенствованы. Например, в том месте, где в первом издании, по существу, говорилось «Найдите метод для реализации X», во втором издании значится: «Развейте приведенный план алгоритма для реализации X. Расширьте этот метод для решения (более трудной) задачи Y».

**Благодарности**

Авторы выражают глубокую благодарность своим коллегам, друзьям и всем, кто оказал им поддержку (включая внимательных читателей первого издания). Перечислим их имена: S. Arch, E. Bach, D. Bailey, A. Balog, M. Barnick, P. Bateman, D. Bernstein, F. Beukers, O. Bonfim, D. Bleichenbacher, J. Borwein, D. Bradley, N. and P. Bragdon, R. Brent, D. Bressoud, D. Broadhurst, N. Bruin, Y. Bugeaud, L. Buhler, G. Campbell, M. Campbell, D. Cao, P. Carmody, E. Catmull, H. Cohen, D. Copeland, D. Coppersmith, J. Cosgrave, H. Darmon, T. Day, K. Dilcher, J. Doenias, G. Effinger, N. Elkies, T. Engelsma, J. Essick, J. Fessler, J. Fix, W. Galway, B. Garst, M. Gesley, G. Gong, A. Granville, D. Griffiths, R. Harley, E. Hasibar, D. Hayes, D. Hill, U. Hofmann, N. Howgrave-Graham, J. Huang, S. Jobs, A. Jones, B. Kaliski, W. Keller, M. Kida, K. Kim, J. Klivington, K. and S. Koblik, D. Kohel, D. Kramer, A. Kruppa, S. Kurowski, S. Landau, A. Lenstra, H. Lenstra, M. Levich, D. Lichtblau, D. Lieman,

I. Lindemann, D. Loebenberger, M. Martin, E. Mayer, F. McGuckin, M. Mignotte, P. Mihăilescu, V. Miller, D. Mitchell, V. Mitchell, T. Mitra, P. Montgomery, W. Moore, V. Müller, G. Nebe, A. Odlyzko, H. Oki, F. Orem, J. Papadopoulos, N. Patson, A. Perez, J. Pollard, A. Powell, J. Powell, L. Powell, J. Renze, P. Ribenboim, B. Salzberg, A. Schinzel, T. Schulmeiss, J. Seamons, J. Shallit, M. Shokrollahi, J. Solinas, L. Somer, D. Stehlé, D. Symes, D. Terr, E. Teske, A. Tevanian, R. Thompson, M. Trott, S. Wagon, S. Wagstaff Jr., M. Watkins, P. Wellin, N. Wheeler, M. Wiener, T. Wieting, J. Williams, P. Winkler, S. Wolfram, G. Woltman, A. Wylde, A. Yerkes, A. Zaccagnini, Z. Zhang, P. Zimmermann.

Оказанная нам помощь была связана с технической, теоретической, вычислительной, литературной и этической сторонами подготовки рукописи. Особую благодарность нам хотелось бы выразить нашему старинному другу и коллеге Джо Булеру, который в своей неподражаемой энциклопедической манере бескорыстно помогал нам преодолеть многочисленные теоретические и вычислительные неувязки при работе над текстом. Благодаря вкладу всех этих прекрасных коллег, книга получилась несоизмеримо лучше, чем могла бы быть без их участия.

Портленд, штат Орегон, США — *Ричард Крэндалл*
Ганновер, штат Нью-Гемпшир, США — *Карл Померанс*

Декабрь 2000
Декабрь 2001 (дополнительный тираж, с исправлениями)
Апрель 2005 (второе издание)

Глава 1

# ПРОСТЫЕ ЧИСЛА!

Простые числа принадлежат недосягаемому миру мыслимых сущностей. Эти чудесные объекты допускают простое, изящное описание, и в то же время приводят к невероятным — можно даже сказать невообразимым — трудностям в изучении их свойств. Само понятие простоты можно объяснить и ребенку, но тем не менее, ни один человеческий мозг не в состоянии охватить целиком всю картину. В наши дни, когда теоретики продолжают схватку со всей лавиной простых чисел, громадные силы и средства направлены на исследование вычислительных аспектов данной проблемы: на задачи отыскания, классификации и применения простых чисел в других областях знаний. Именно этим вычислительным аспектам мы и посвятим дальнейшие главы. Однако же, нам придется время от времени обращаться к теории для того, чтобы осветить, подчеркнуть и обосновать практическую значимость вычислительных алгоритмов.

Итак, натуральное число $p$ называется *простым*, если оно имеет лишь два делителя, а именно 1 и $p$. Натуральное число называется *составным*, если оно превосходит единицу и не является простым. (Считается, что число 1 — не простое и не составное.) Таким образом, число $n$ является составным тогда и только тогда, когда оно допускает так называемое нетривиальное разложение $n = ab$, где целые числа $a, b$ лежат строго между 1 и $n$. Несмотря на изысканную ясность определения простоты, получающаяся последовательность простых чисел $2, 3, 5, 7, \ldots$ будет далеко не тривиальным постоянным объектом нашего внимания. Изумительные свойства простых чисел, а также связанные с ними известные результаты и открытые проблемы многочисленны и разнообразны. Мы затронем в этой главе те моменты, которые нам видятся наиболее красивыми и интересными с теоретической точки зрения, а также проясним и некоторые практические аспекты. В то же время мы коснемся важной проблемы разложения на множители — темы, тесным образом связанной с изучением самих простых чисел.

В оставшейся части настоящей главы мы предложим способ вычисления характеров и распознавания простых чисел, а также приведем сопутствующие практические сведения.

## 1.1. Прогресс и проблемы

### 1.1.1. Основные проблемы и теоремы

Первый результат состоит в том, что простые числа являются мультипликативным базисом множества натуральных чисел.

**Теорема 1.1.1** (основная теорема арифметики). *Каждое натуральное число $n$ можно единственным образом представить в виде*

$$n = p_1^{a_1} p_2^{a_2} \dots p_k^{a_k},$$

*где все показатели $a_i$ — натуральные числа, а числа $p_1 < p_2 < \dots < p_k$ — простые.*

(Если число $n$ является простым, то его представление по этой теореме совпадает с ним самим, при этом $k = 1$ и $a_1 = n$. В случае $n = 1$ теорема имеет место в смысле пустого произведения, т. е. $k = 0$.) Доказательство теоремы 1.1.1 естественным образом распадается на две части: существование разложения числа $n$ на простые множители и его единственность. Существование доказывается очень легко (достаточно рассмотреть наименьшее число, не имеющее искомого разложения, представить его в виде произведения двух множителей и получить противоречие). Единственность — несколько более тонкая задача. Ее можно вывести из более простого результата, а именно, из «первой теоремы» Евклида (см. упр. 1.2).

Основная теорема арифметики дает начало тому, что можно назвать «основной задачей арифметики»: найти разложение данного целого числа $n > 1$ на простые множители. Обратимся теперь к текущему положению дел в вычислительной области.

## 1.1.2. Технологический и алгоритмический прогресс

В действительности больших чисел не существует: любое указанное натуральное число можно назвать «маленьким». В самом деле, сколько бы мы ни выписали в числе знаков и какую бы ни возвели башню из показателей, будет лишь конечное количество чисел, меньших рассматриваемого, и бесконечное бóльших. Но хотя мы и обречены вечно иметь дело лишь с малыми числами, нам все-таки остается задача разбираться с числами, бóльшими уже изученных. И на этом пути достигнут значительный прогресс. Сейчас мы можем разлагать на множители числа, количество знаков в которых в восемь раз больше, чем 30 лет назад, а количество знаков в числах, простоту которых мы можем запросто распознать, примерно в 500 раз больше.

Обратим внимание читателя на тот факт, что вычислительный прогресс имеет две стороны: технологическую и алгоритмическую. Несомненно, надо отдать должное качеству и количеству вычислительной техники, но, что также несомненно, не в полной мере. Если бы мы до сих пор использовали алгоритмы, созданные до 1975 г., то даже с помощью наилучшего доступного сегодня оборудования мы не смогли бы разложить на множители или установить простоту числа, состоящего более чем из 40 знаков.

Итак, что же возможно в наши дни? Текущий рекорд для разложения на множители произвольных чисел — более 170 знаков, а проверить на простоту сегодня можно любое число вплоть до $10^{15000}$. В колонке «Математические игры» журнала «Scientific American» М. Гарднером [Gardner 1977] была постав-

лена задача разложения на множители 129-значного числа. Число

$$\begin{aligned}\mathrm{RSA129} = \quad & 1143816257578888676692357799761466120102182967212423 6\backslash \\ & 2562561842935706935245733897830597123563958705058989 0\backslash \\ & 7514759929002687954354 1\end{aligned}$$

было предложено в качестве теста для новой криптосистемы RSA (см. гл. 8). Несмотря на прогнозы некоторых исследователей о том, что решение данной задачи займет 40 квадрильонов лет, это число было разложено на множители в 1994 г. методом квадратичного решета (QS) (см. гл. 6) Д. Аткинсом, М. Граффом, А. Ленстрой и П. Лейландом. Число RSA129 оказалось равным

$$\begin{aligned}& 3490529510847650949147849619903898133417764638493387843990820577 \\ & \times \\ & 3276913299326670954996198819083446141317764296799294253979828853 3,\end{aligned}$$

а тайное сообщение после расшифровки оказалось таким: "THE MAGIC WORDS ARE SQUEAMISH OSSIFRAGE"[1].

За прошедшее десятилетие были решены многие подобные и родственные знаковые задачи. Например, сегодня господствует метод решета числового поля (NFS): на момент выхода второго издания настоящей книги с его помощью было разложено на множители 174-разрядное[2] число RSA-576, а «специальный» метод SNFS позволил справляться и с 248-разрядными числами. Метод эллиптических кривых (ECM) достиг 59-разрядных чисел (для нахождения простого делителя, не являющегося наибольшим для данного числа). Такие достижения можно найти на постоянно обновляющемся сайте [Zimmermann 2000]. Ниже мы вернемся к рассказу об этом списке рекордов.

Еще одним из представляющих интерес достижений было отыскание делителей различных чисел Ферма $F_n = 2^{2^n} + 1$, что обсуждается в разд. 1.3.2. Некоторые из начальных чисел Ферма, такие как $F_9, F_{10}, F_{11}$, были разложены на множители полностью, а для более гигантских $F_n$ были найдены впечатляющие делители.

В зависимости от величины числа Ферма для решения этой задачи применяется метод решета числового поля (NFS) (для малых чисел Ферма, например, для $F_9$) или метод эллиптических кривых (ECM) (см. гл. 6 и 7). Последний метод позволяет отыскать делители, состоящие более чем из 30 и даже 40 знаков. Представленные в различных разделах настоящей книги методы позволяют реализовать тест на простоту для таких больших чисел Ферма, как $F_{24}$, которое содержит более пяти миллионов цифр. Снова отметим, что подобные достижения оказались возможными как за счет продвижения в области компьютерных технологий и программ, так и за счет усовершенствования алгоритмов. Одна из возможных технологий будущего — квантовый компьютер — может привести к такому потрясающему прогрессу в технике, что

[1] Волшебные слова — SQUEAMISH OSSIFRAGE. SQUEAMISH — брезгливый, привередливый, обидчивый; OSSIFRAGE — скопа (вид птиц семейства выпи). — *Прим. ред.*

[2] Здесь и далее подразумевается десятичная запись числа. — *Прим. перев.*

процесс разложения на множители лет через двадцать станет невообразимо быстрым. Квантовые вычисления обсуждаются в разд. 8.5.

Как уже было сказано, простые числа играют заметную роль в современной криптографии — науке, занимающейся шифрованием и расшифровкой тайных сообщений. В связи с тем, что многие криптографические системы используют свойства простых чисел, разложение на множители, а также другие родственные теоретико-числовые задачи, технологические и алгоритмические достижения приобрели еще большую важность. Тот факт, что наша способность находить большие простые числа (распознавать их простоту) опережает наши возможности в области разложения на множители, заложен в основу ряда криптографических алгоритмов. На момент написания книги самым большим найденным простым числом было гигантское простое число Мерсенна $2^{25964951} - 1$, которое можно себе представить примерно как объемную книгу, заполненную цифрами. В главе 9 приводятся алгоритмы, позволяющие быстро выполнять арифметические операции над такими громадными числами. Но опять, наряду с такими алгоритмическими успехами происходит и техническое продвижение. Продолжая мысль о сопоставлении возможностей тесного союза машин и алгоритмов, отметим, что в 2004 г. проверка простоты упоминавшегося «наибольшего простого числа» вида $2^q - 1$ заняла бы неделю работы процессора обычного современного персонального компьютера (см. постоянно обновляющийся сайт [Woltmann 2000]). Для сравнения, за десять лет до этого подобная задача для числа порядка $2^{20000000}$ потребовала бы несколько *лет* (если не пятилеток!) работы процессора обычного персонального компьютера того времени. Разумеется, такой прогресс стал возможным за счет как новой техники, так и новых алгоритмов. Продолжая сравнение далее, отметим, что сейчас, в начале XXI века, стандартная рабочая станция, снабженная соответствующими программами, способна выполнить умножение двух чисел, имеющих миллион знаков, за долю секунды. Как сказано в конце разд. 9.5.2, современное кластерное оборудование способно производить умножение двух *миллиардзначных* чисел примерно за минуту.

Специальный вид чисел Мерсенна, а именно, $2^q - 1$, делает доказательство их простоты сравнительно несложным. Существует очень быстрый тест Люка—Лемера, описанный в гл. 4. Что же касается простых чисел, не имеющих специального вида (так сказать, «случайных» простых чисел), то современные тесты на простоту годятся лишь для тех из них, которые состоят лишь из нескольких тысяч цифр. Первым, кто занялся практическим решением этой проблемы, был Ф. Морейн. На основе идей А. Аткина и других он развил эффективный метод эллиптических кривых для доказательства простоты (ECPP), а также новый быстрый метод эллиптических кривых (fastECPP) (см. гл. 7). Поистине впечатляющим результатом метода ECPP на рубеже веков стало доказательство Майером и Морейном (см. [Morain 1998]) простоты числа $(2^{7331} - 1)/458072843161$, состоящего из 2196 знаков. Сенсационное заявление сделали в июле 2004 г. Франке, Кляйнюнг, Морейн и Вирт: метод fastECPP

позволил установить, что 15071-значное число Лейланда

$$4405^{2638} + 2638^{4405}$$

является простым.

Помимо современных достижений в области разложения на множители и проверки простоты, совершается великое множество рекордных попыток добиться успеха в еще более специальных областях. Время от времени открываются новые наибольшие простые близнецы (пары простых чисел $p, p+2$), исключительно длинные арифметические прогрессии, состоящие из простых чисел, а также любопытные примеры простых чисел, обладающих другими частными свойствами. Ведутся также поиски простых чисел, которые мы надеемся когда-либо найти, но однако до сих пор не обнаружили (таких, как новые экземпляры простых чисел Вифериха, Вильсона и Уолла—Сана—Сана). В различных разделах настоящей книги мы обращаемся к некоторым задачам из огромного множества подобных проблем, особенно к тем, решение которых связано с вычислительными аспектами, не выходящими за рамки нашей книги.

Оставляя в стороне лишние подробности и частные отклонения, сообщаем читателю, что существует широко распространенная «культура» вычислительных исследований. Примером книги, легко читаемой и способной заинтересовать успехами в области простых чисел и «рекордов» разложения на множители, может служить книга [Ribenboim 1996], а также популярный и подробный информационный бюллетень С. Вагштафа мл. о текущем положении дел в области разложения на множители чисел Каннингема (чисел вида $b^n \pm 1$, где $b \leqslant 12$). Сводка этого бюллетеня доступна на сайте [Wagstaff 2004]. Перечислим несколько недавних рекордов.

- В 2003 г. Ф. Циммерман ($p-1$)-методом Полларда (см. разд. 5.4) нашел 57-значные делители чисел $6^{396}+1$ и $11^{260}+1$.
- Метод эллиптических кривых (ECM) (см. разд. 7.4.1) позволил найти 57-значный делитель числа $2^{997}-1$ (см. [Wagstaff 2004]), 58-значный делитель числа $8 \cdot 10^{141}-1$ (Р. Бэкстром, 2003 г.), а также 59-значный делитель числа $10^{233}-1$ (Б. Додсон, 2005 г.). (Удивительным является то обстоятельство, что за последнее десятилетие методы ($p-1$) и ECM довольно редко приводили к нахождению близких по порядку делителей.)
- В конце 2001 г. методом квадратичного решета (QS) (см. разд. 6.1), а именно его вариантом с тремя большими простыми, было разложено на множители 135-значное число $2^{803}-2^{402}+1$. Похоже, что это было, так сказать, последним вздохом метода QS на фоне господствования более современных методов NFS и SNFS для разложения чисел такой величины.
- Метод общего решета числового поля (GNFS) позволил, как отмечалось выше, найти делители 174-значного числа RSA-576. Для чисел специального вида метод специального решета числового поля (SNFS) (см. разд. 6.2.7) может отыскивать делители и более чем 200-значных чисел, таких как рекордное 248-значное число $2^{821}+2^{411}+1$.

Детали, связанные с получением ряда рекордных разложений, можно найти в упомянутом выше бюллетене Вагштафа. Далее в нашей книге (в частности, после алгоритма 7.4.4 и иных аналогичных местах) приводятся более ранние достижения, которые вошли в первое издание и мы решили их сохранить по причине исторической важности. Кроме того, мы предоставляем читателю возможность не только отслеживать последние результаты, но и видеть путь, пройденный до их получения.

Сейчас, в начале XXI века, бóльшая часть приводимых вычислений является широко доступной. Хорошее изложение содержится и в книге [Peterson 2000]. Другое описание достижений в области больших чисел можно найти в статье [Crandall 1997a]. В последней работе приводится оценка, позволяющая хотя бы грубо ответить на вопрос: «Сколько вычислительных операций было произведено всеми вычислительными машинами за все время их существования?» Можно говорить об основных операциях, таких как логическое «и», численное сложение, умножение и т. д. Ответ на этот вопрос важен для ряда рассматриваемых в этой книге проблем и может быть назван «правилом моля». Грубо говоря, к рубежу XX и XXI веков примерно один моль, т. е. число Авогадро $6 \cdot 10^{23}$ из химии (мы его будем считать равным $10^{24}$), операций было сделано всеми машинами за всю историю их существования. Несмотря на обычные загадочность и трепет, вызываемые понятием индустриальных и правительственных супервычислений, существует немало персональных компьютеров, умеющих оперировать величиной $10^{24}$. Существенным моментом является то, что разложение целого числа $N \approx 10^{50}$ на простые множители напрямую является безнадежным в том смысле, что для этого пришлось бы выполнять вычислительные действия в количестве, превосходящем число операций, произведенных за всю историю человечества. Продолжим идею сопоставления. Стандартный пример разложения на множители или распознавания простоты требует в настоящее время от $10^{16}$ до $10^{18}$ машинных операций. Аналогичным образом, полнометражный современный фильм (например, диснеевский фильм студии Pixar «В поисках Немо» 2003 г.) включает порядка $10^{18}$ операций. Удивительно, что после проведения такого огромного количества операций можно получить как простой ответ (делитель, а может быть и единственный информационный бит «простое/составное»), так и создать полнометражную анимационную картину, между которой и однобитовым ответом простирается пропасть. Интересно, что вычислительная задача, содержащая $10^{18}$ операций, является одной десятимиллионной частью всего исторического вычислительного процесса на Земле.

### 1.1.3. Бесконечность множества простых чисел

Хотя современные технологии и алгоритмы и позволяют находить простые числа впечатляющей величины, давным-давно подмечено, что множество неизведанных простых чисел никогда не иссякнет. В самом деле, как было показано в 300 г. до н. э. Евклидом, бывшим в ту пору профессором великого Александрийского университета, существует бесконечное множество простых чисел

(см. [Archibald 1949]). Это достижение можно назвать началом абстрактной теории простых чисел. Знаменитое доказательство следующей теоремы также принадлежит Евклиду.

**Теорема 1.1.2** (Евклид). *Простых чисел бесконечно много.*

Доказательство. Предположим, что существует лишь конечное множество простых чисел. Обозначим самое большое простое число через $p$ и рассмотрим число, на единицу большее произведения всех простых чисел:

$$n = 2 \cdot 3 \cdot 5 \ldots p + 1.$$

Ясно, что $n$ не может делиться ни на одно из простых чисел от 2 до $p$, ведь остаток от каждого такого деления равен 1. Но мы предположили, что простых чисел, превосходящих $p$, не существует, а значит, число $n$ не имеет ни одного простого делителя, что противоречит теореме 1.1.1. Полученное противоречие означает справедливость утверждения теоремы. □

Будет небезынтересным отметить, что Евклид никогда явно не формулировал основную теорему арифметики (теорему 1.1.1). Однако та часть этой теоремы, которая утверждает существование простого делителя у любого целого числа, превосходящего единицу, была Евклиду известна, и именно ее он использовал при доказательстве теоремы 1.1.2.

Существует множество вариантов этой классической теоремы, отличающихся как формулировкой, так и методом доказательства (см. разд. 1.3.2 и 1.4.1). Рассмотрим один любопытный вариант, подчеркивающий тот факт, что основная теорема арифметики несет информацию о распределении простых чисел. Обозначим множество всех простых чисел через $\mathcal{P}$. Определим функцию действительного переменного $x$, считающую простые числа, так:

$$\pi(x) = \#\{p \leqslant x : p \in \mathcal{P}\},$$

т. е. значение $\pi(x)$ равно количеству простых чисел, не превосходящих $x$. Основная теорема арифметики утверждает, что для каждого натурального числа $x$ количество решений неравенства

$$\prod p_i^{a_i} \leqslant x,$$

где $p_i$ обозначает $i$-е простое число, а показатели $a_i$ неотрицательны, равно в точности числу $x$. Поскольку каждый из множителей $p_i^{a_i}$ не превосходит $x$, количество возможных вариантов выбора показателя $a_i$, включая нулевое значение, ограничено сверху величиной $\lfloor 1 + (\ln x)/(\ln p_i) \rfloor$. Отсюда следует, что

$$x \leqslant \prod_{p_i \leqslant x} \left\lfloor 1 + \frac{\ln x}{\ln p_i} \right\rfloor \leqslant \left(1 + \frac{\ln x}{\ln 2}\right)^{\pi(x)},$$

откуда для всех $x \geqslant 8$ имеем неравенство

$$\pi(x) > \frac{\ln x}{2 \ln \ln x}.$$

Несмотря на то, что эта оценка сравнительно слаба, она, тем не менее, позволяет доказать бесконечность множества простых чисел как следствие основной теоремы арифметики.

Для доказательства теоремы 1.1.2 Евклид применил идею построения нового простого числа на основе уже имеющихся. Можно ли таким путем получить все простые числа? Вот несколько возможных интерпретаций этого вопроса.

(1) Зададим последовательность простых чисел $q_1, q_2, \ldots$ индуктивно: $q_1 = 2$ и $q_{k+1}$ есть наименьший простой делитель числа $q_1 \ldots q_k + 1$. Содержит ли последовательность $(q_i)$ все простые числа?

(2) Зададим последовательность простых чисел $r_1, r_2, \ldots$ индуктивно: $r_1 = 2$ и $r_{k+1}$ есть наименьшее до сих пор не выбранное простое число, которое делит хотя бы одно из чисел вида $d + 1$, когда $d$ пробегает все делители произведения $r_1 \ldots r_k$. Содержит ли последовательность $(r_i)$ все простые числа?

(3) Зададим последовательность простых чисел $s_1, s_2, \ldots$ индуктивно: $s_1 = 2$, $s_2 = 3$ и $s_{k+1}$ есть наименьшее до сих пор не выбранное простое число, которое делит хотя бы одно из чисел вида $s_i s_j + 1$, где $1 \leqslant i < j \leqslant k$. Содержит ли последовательность $(s_i)$ все простые числа? Является ли последовательность $(s_i)$ бесконечной?

Последовательность $(q_i)$ вопроса (1) рассматривалась Гаем и Новаковским, а позднее и Шенксом. Вагштаф [Wagstaff 1993] нашел первые 43 члена этой последовательности. Вычислительной проблемой, стоящей на пути отыскания последующих членов, является огромная величина тех чисел, которые необходимо разложить на множители. Уже число $q_1 \ldots q_{43}$ содержит 180 знаков.

Как было показано в неопубликованной работе Померанса, последовательность $(r_i)$ пункта (2) содержит все простые числа. Более того, при $i \geqslant 5$ элемент $r_i$ является $i$-м по счету простым числом. Доказательство содержит непосредственный вычислительный перебор примерно одного миллиона первых членов, а также ряд явных оценок из аналитической теории чисел, более подробная речь о которой пойдет в этой главе далее. Это доказательство является одним из ярких примеров того, какую важную роль играют вычислительные аспекты.

Что касается пункта (3), неизвестно даже то, бесконечна ли последовательность $(s_i)$, хотя наверняка это так, причем, вероятнее всего, она содержит и все простые числа. Нам неизвестно, пытался ли кто-нибудь решить эту проблему вычислительными методами. Возможно, Вы, читатель, захотите испытать свои силы? Эта задача была поставлена М. Ньюманом из Австралийского национального университета.

Таким образом, отталкиваясь даже от самых начальных и древних идей, связанных с простыми числами, можно быстро добраться до переднего края современных исследований.

### 1.1.4. Асимптотические соотношения и порядковая терминология

Предвидя появление большого числа асимптотических результатов и сложных вычислительных оценок, договоримся об обозначениях, используемых на протяжении всей нашей книги. Запись

$$f(N) \sim g(N)$$

читается так: «функции $f$ и $g$ *асимптотически эквивалентны* при $N$, стремящемся к бесконечности», и означает существование определенного предела, равного единице:

$$\lim_{N\to\infty} f(N)/g(N) = 1.$$

Обозначение

$$f(N) = O(g(N)),$$

читается так: «$f$ есть $O$-большое от $g$» и утверждает существование такого положительного числа $C$, что для всех чисел $N$, начиная с некоторого (или для всех чисел $N$ из некоторого указываемого множества), выполнено неравенство

$$|f(N)| \leqslant C|g(N)|.$$

Обозначение «$o$-малое» используется тогда, когда одна функция значительно превосходит другую, т. е. запись

$$f(N) = o(g(N))$$

означает, что

$$\lim_{N\to\infty} f(N)/g(N) = 0.$$

Приведем ряд примеров использования этих обозначений. Коль скоро функция $\pi(x)$, равная количеству простых чисел, не превосходящих $x$, сама не превосходит $x$ для всех положительных значений переменной, то имеет место соотношение

$$\pi(x) = O(x).$$

Не столь очевидным и требующим некоторой работы (см. упр. 1.11 и 1.13, в которых доказательство разбито на два этапа) является тот факт, что

$$\pi(x) = o(x). \tag{1.1}$$

Уравнение (1.1) можно понимать как утверждение, что простые числа в некотором смысле редки, причем чем дальше мы движемся по натуральному ряду, тем реже они попадаются. Пусть $\mathcal{A}$ — подмножество множества натуральных чисел и $A(x)$ обозначает количество элементов из $\mathcal{A}$, не превосходящих $x$. Тогда, если $\lim_{x\to\infty} A(x)/x = d$, то мы будем называть число $d$ *асимптотической плотностью* множества $\mathcal{A}$. Таким образом, уравнение (1.1) означает, что множество простых чисел имеет асимптотическую плотность нуль. Заметим, что не все подмножества множества натуральных чисел имеют асимптотическую

плотность. В самом деле, предел, указанный в определении, может и не существовать. В качестве примера приведем множество чисел, имеющих четное количество цифр.

В нашей книге, говоря о вычислительной сложности алгоритмов, мы почти всегда будем придерживаться обозначения «$O$», несмотря на то, что некоторые авторы обозначают битовую и операционную сложности через $O_b$ и $O_{op}$ соответственно. Поэтому, когда сложность алгоритма записана через $O$, мы для каждого случая будем по возможности указывать, какую именно сложность мы имеем в виду. Необходимо учитывать, что эти величины не обязательно пропорциональны, поскольку это зависит от того, производятся ли «операции» в поле, являются они сложением, умножением или сравнением (что имеет место для условных операторов). Например, в гл. 9 мы увидим, что базовый метод умножения с помощью БПФ требует $O(D \ln D)$ операций над числами с плавающей точкой, каждое из которых имеет $D$ разрядов (по подходящему основанию), в то время как существуют методы, имеющие битовую сложность $O(n \ln n \ln\ln n)$, где $n$ обозначает общее число битов во множителях. В этом случае отсутствует явная пропорциональность; соотношения между числом разрядов, основанием и битовым размером $n$ нетривиальны (особенно когда в вычислениях возникают ошибки, связанные с плавающей точкой) и т. д.

Еще одно нетривиальное сравнение связано с дзета-функцией Римана, для которой существуют методы вычислений ее значений с $D$ точными знаками за $O(D)$ операций, причем здесь имеется в виду *полная точность*, т. е. операции проводятся с $D$ разрядами. В свою очередь, битовая сложность вычисления $D$ точных знаков (или пропорционального числа битов) растет быстрее. Сравнить эти сложности нетрудно: произведение двух больших целых чисел требует *одну* (высокоточную) операцию, а для реализации этого умножения требуется множество битовых действий. Таким образом, можно сказать, что между этими сложностями нет явного соотношения. Можно спросить: если эти величины (битовая и основанная на операциях сложности) настолько различны, не является ли одна из них более предпочтительной или сильной, чем другая? Ответ таков: нет, не обязательно. Может случиться так, что доступные технические и программные средства пригодны для всех вычислений с полной точностью, т. е. каждое сложение и умножение производится с $D$ знаками. В этом случае мы заинтересованы в оценке операционной сложности. Если же, с другой стороны, мы желаем начать все «с нуля» и создать специальные операции с оптимальной битовой сложностью, точность которых изменяется динамически во время вычислений, то нас может скорее заинтересовать оценка битовой сложности. Вообще говоря, можно сказать, что выбор между битовой и операционной сложностями зачастую аналогичен выбору между яблоками и апельсинами.

Поскольку словосочетание «время счета» приобрело определенную популярность, мы будем иногда использовать этот термин как синоним битовой сложности. Разумеется это связано с тем, что реальное, физическое время работы программы пропорционально общему количеству производимых битовых операций. Тем не менее, в будущем эта равносильность вполне может исчез-

нуть — благодаря квантовым компьютерам, массовому параллелизму, прогрессу в пословной обработке арифметических действий и т. д. Аналогичным образом, под словосочетанием «полиномиальное время» мы подразумеваем, что число битовых операций ограничено сверху фиксированной степенью количества битов входящих операндов. Например, ни один из преобладающих сегодня алгоритмов разложения на множители (ECM, QS, NFS) не работает за полиномиальное время, а простое сложение, умножение, возведение в степень и т. д. характеризуются полиномиальным временем. В частности, возведение в степень, т. е. вычисление $x^y \bmod z$, при помощи простых процедур имеет битовую сложность $O(\ln^3 z)$ (если размеры натуральных чисел $x, y, z$ сравнимы), а следовательно, работает за полиномиальное время. Аналогичным образом, вычисление наибольшего общего делителя также занимает полиномиальное время.

### 1.1.5. Распределение простых чисел

В 1737 г. Л. Эйлер опубликовал новое доказательство бесконечности множества простых чисел. Он доказал, что ряд, составленный из чисел, обратных к простым, расходится, а следовательно, содержит бесконечно много членов (см. упр. 1.20).

В середине XIX века П. Л. Чебышёв доказал результат, который показывает истинный порядок роста функции $\pi(x)$.

**Теорема 1.1.3** (Чебышёв). *Существуют такие положительные числа $A$ и $B$, что при всех $x \geqslant 3$ справедливо неравенство*

$$\frac{Ax}{\ln x} < \pi(x) < \frac{Bx}{\ln x}.$$

Например, теорема 1.1.3 имеет место при $A = 1/2$ и $B = 2$. Это было чрезвычайно значимым достижением, поскольку с 1791 г., когда четырнадцатилетний (!) Гаусс высказал гипотезу об асимптотическом поведении функции $\pi(x)$, за полтора века (до работы Чебышёва) в этом направлении было сделано крайне мало успешных шагов. Упомянутая гипотеза Гаусса теперь носит название «теорема о простых числах»[3].

**Теорема 1.1.4** (Адамар и де ля Валле Пуссен). *При $x \to +\infty$ имеем*

$$\pi(x) \sim \frac{x}{\ln x}.$$

Таким образом, П. Л. Чебышёв был близок к доказательству асимптотического закона распределения простых чисел. В частности, ему было даже известно, что если функция $\pi(x)$ асимптотически эквивалентна функции вида $Cx/\ln x$, то постоянная $C$ непременно равна 1. Но трудность в установлении асимптотического закона заключается именно в доказательстве существования предела $\lim_{x\to+\infty} \pi(x)/(x/\ln x)$. Этот заключительный шаг был сделан через

[3] В русскоязычной литературе общепринят термин «асимптотический закон распределения простых чисел». — *Прим. перев.*

полвека, в 1896 г., Ж. Адамаром и Ш. де ля Валле Пуссеном независимо друг от друга. Фактически было доказано более сильное утверждение: существует такая положительная постоянная $C$, что

$$\pi(x) = \operatorname{li}(x) + O\left(xe^{-C\sqrt{\ln x}}\right), \tag{1.2}$$

где через $\operatorname{li}(x)$ обозначен интегральный логарифм числа $x$, определяемый формулой

$$\operatorname{li}(x) = \int_2^x \frac{1}{\ln t}\,dt \tag{1.3}$$

(несколько иное определение дано в упр. 1.36). Поскольку $\operatorname{li}(x) \sim x/\ln x$, интегрирование по частям в соотношении (1.3) (или применение правила Лопиталя) приводит к утверждению теоремы 1.1.4. На протяжении всего двадцатого века величина остаточного члена $\pi(x) - \operatorname{li}(x)$ была предметом интенсивного изучения, но существенных продвижений не произошло. В разд. 1.4 мы вернемся к теме асимптотического закона распределения простых чисел. А сейчас отметим одну полезную, хотя и эвристическую, трактовку этой теоремы: для большого случайного целого числа $x$ «вероятность» того, что $x$ — простое число, примерно равна $1/\ln x$.

Будет небезынтересным отметить способ, позволивший Гауссу выдвинуть упомянутую знаменитую гипотезу. Из таблиц простых чисел видно, что они становятся все реже и реже по мере своего возрастания, но локальное их распределение крайне нерегулярно. Рассмотрение таблиц простых чисел натолкнуло Гаусса на мысль разбить множество простых чисел на блоки длины 1000. Этого оказалось достаточным для того, чтобы несколько «сгладить» нерегулярность и выявить закон: «вероятность» того, что случайное число $\approx x$ является простым, приблизительно равна $1/\ln x$. Отсюда Гаусс и сделал вывод, что функция $\pi(x)$ должна вести себя, как интегральный логарифм.

Несмотря на то, что Гаусс приступил к изучению функции $\pi(x)$ за в первые годы XVIII в., результаты его работы были опубликованы лишь несколько десятилетий спустя. Тем временем Лежандр независимо выдвинул асимптотический закон распределения простых чисел, но в несколько иной форме

$$\pi(x) \sim \frac{x}{\ln x - B}, \tag{1.4}$$

где $B = 1.08366\ldots$. Поскольку вне зависимости от выбора постоянной $B$ справедливо соотношение $x/\ln x \sim x/(\ln x - B)$, возникает вопрос о том, какое приближение (включающее число $B$ или предложенный Гауссом $\operatorname{li}(x)$) является более точным. Оказывается, что оценка Гаусса гораздо предпочтительнее. Равенство (1.2) означает, что $|\pi(x) - \operatorname{li}(x)| = O(x/\ln^k x)$ для каждого $k > 0$ (причем постоянная в символе $O$ зависит от $k$). Поскольку

$$\operatorname{li}(x) = \frac{x}{\ln x} + \frac{x}{\ln^2 x} + O\left(\frac{x}{\ln^3 x}\right),$$

наилучшим выбором численного значения постоянной $B$ в соотношении (1.4) является отличное от предложенного Лежандром число $B = 1$. В свою очередь,

| $x$ | $\pi(x)$ |
|---|---|
| $10^2$ | 25 |
| $10^3$ | 168 |
| $10^4$ | 1229 |
| $10^6$ | 78498 |
| $10^8$ | 5761455 |
| $10^{12}$ | 37607912018 |
| $10^{16}$ | 279238341033925 |
| $10^{17}$ | 2623557157654233 |
| $10^{18}$ | 24739954287740860 |
| $10^{19}$ | 234057667276344607 |
| $10^{20}$ | 2220819602560918840 |
| $10^{21}$ | 21127269486018731928 |
| $10^{22}$ | 201467286689315906290 |
| $4 \cdot 10^{22}$ | 783964159847056303858 |

**Таблица 1.1.** Значения функции количества простых чисел $\pi(x)$. Для получения современных результатов при больших $x$ применены распределенные сетевые вычисления.

оценка

$$\pi(x) \approx \frac{x}{\ln x - 1}$$

удобна для вычислений на карманном калькуляторе.

Поразительно, насколько точно́ приближение функции $\pi(x)$ интегральным логарифмом. Положим $x = 10^{21}$ и воспользуемся табл. 1.1. Благодаря недавним вычислениям Гурдона, основанным на более ранних работах Делеглиза, Ривата и Циммермана, мы знаем, что

$$\pi\left(10^{21}\right) = 21127269486018731928,$$

в то время как

$$\mathrm{li}\left(10^{21}\right) \approx 21127269486616126181.3$$

и

$$\frac{10^{21}}{\ln(10^{21}) - 1} \approx 21117412262909985552.2.$$

В самом деле, приближение функцией $\mathrm{li}\,(x)$ действительно впечатляет! К этой теме мы еще вернемся в настоящей главе, когда будем обсуждать гипотезу Римана (RH) (см. гипотезу 1.4.1 и последующие замечания).

Самые свежие значения из табл. 1.1, а именно, $\pi(10^{22})$, $\pi(4 \cdot 10^{22})$, принадлежат Гурдону и Себа [Gourdon and Sebah 2004]. Однако эти исследователи, работая над вычислением значения $\pi(10^{23})$, недавно обнаружили неточность в своей программе, приводящую к численному расхождению с результатами частичного просеивания. Поэтому пока эта проблема не решена или не про-

ведены независимые подтверждающие вычисления, опубликованные значения $\pi(10^{22})$ и $\pi(4 \cdot 10^{22})$, видимо, следует считать экспериментальными.

Перейдем к еще одному вопросу исторической важности. Какие арифметические прогрессии, т. е. последовательности целых чисел $\{a, a+d, a+2d, \ldots\}$, содержат простые числа, и если содержат, то насколько часто простые числа встречаются в такой прогрессии? Этот вопрос можно поставить и так: в каких классах вычетов $a \bmod d$ можно встретить простые числа и какова относительная плотность множества простых чисел в этом классе? Если числа $a$ и $d$ имеют общий простой делитель $p$, то и каждый член соответствующей арифметической прогрессии (или элемент класса вычетов) кратен этому делителю, а следовательно, эта прогрессия (класс вычетов) не может содержать простых чисел, отличных от $p$. Основной классический результат в этом направлении заключается в том, что описанное препятствие является единственным на пути существования бесконечного числа простых в арифметической прогрессии (классе вычетов).

**Теорема 1.1.5** (Дирихле). *Пусть целые числа $a$ и $d > 0$ взаимно просты (т. е. не имеют общих простых делителей). Тогда арифметическая прогрессия $\{a, a+d, a+2d, \ldots\}$ содержит бесконечно много простых чисел. Более того, ряд, составленный из всех чисел, обратных к простым из такой прогрессии, расходится.*

Эта чудесная (и нетривиальная) теорема была впоследствии уточнена. Было доказано, что если обозначить через $\pi(x; d, a)$ количество простых чисел рассматриваемой прогрессии, не превосходящих $x$, то для фиксированных взаимно простых чисел $a$ и $d > 0$ справедливы соотношения

$$\pi(x; d, a) \sim \frac{1}{\varphi(d)}\pi(x) \sim \frac{1}{\varphi(d)}\frac{x}{\ln x} \sim \frac{1}{\varphi(d)}\operatorname{li}(x), \tag{1.5}$$

где $\varphi$ обозначает функцию Эйлера: $\varphi(d)$ равно количеству натуральных чисел из промежутка $[1, d]$, взаимно простых с числом $d$. Поскольку классы вычетов по модулю $d$, состоящие из чисел, не взаимно простых с числом $d$, могут содержать не более одного простого числа, все остальное бесконечное множество простых чисел оказывается распределенным по оставшимся $\varphi(d)$ классам вычетов по модулю $d$, причем соотношение (1.5) утверждает, что каждый такой класс вычетов по модулю $d$ получит, в асимптотическом смысле, равную долю простых чисел. Таким образом, соотношение (1.5) представляется естественным. Ниже мы рассмотрим ключевые уточнения остаточного члена в рассмотренных асимптотических формулах. Утверждение (1.5) известно как теорема о простых числах в классах вычетов или асимптотический закон распределения простых чисел в арифметических прогрессиях.

Здесь уместно отметить, что также представляет интерес и вопрос о конечных арифметических прогрессиях, состоящих исключительно из простых чисел. Например, числа

$$\{1466999,\ 1467209,\ 1467419,\ 1467629,\ 1467839\}$$

образуют арифметическую прогрессию из пяти простых чисел с разностью

$d = 210$. Вот более длинная прогрессия, состоящая из меньших простых чисел: $\{7, 37, 67, 97, 127, 157\}$. Любопытно, что если бы мы позволили себе говорить и об отрицательных простых числах, то этот пример можно было бы дополнить слева числами $\{-113, -83, -53, -23\}$. Дальнейший материал на эту тему можно найти в упр. 1.41, 1.42, 1.45 и 1.87.

В 2004 г. был получен поистине сенсационный результат – доказательство существования сколь угодно длинных арифметических прогрессий, состоящих исключительно из простых чисел. Проведенные авторами рассуждения не только не следуют стандартным приемам решения подобного рода задач, но совершают настоящий прорыв, привлекая методы гармонического анализа. Поразительно, как обращение к методам из других областей математики может обогатить имеющийся арсенал и открыть новые горизонты! Подробности см. в работе [Green and Tao 2004]. Много лет назад Эрдёш и Туран выдвинули гипотезу о том, что если $S$ — такое подмножество множества натуральных чисел, что ряд, составленный из обратных к ним величин, расходится, то из элементов множества $S$ можно составить сколь угодно длинные арифметические прогрессии. Поскольку, как показал Л. Эйлер, ряд чисел, обратных ко всем простым, расходится (см. рассуждения в окрестности формулы (1.19) и упр. 1.20), если гипотеза Эрдёша—Турана справедлива, то множество всех простых чисел обязано содержать сколь угодно длинные арифметические прогрессии. Поэтому могло показаться, что для доказательства этого свойства простых чисел потребуется лишь расходимость упомянутого ряда обратных величин. Но увы: Грин и Тао воспользовались в рассуждениях иными свойствами простых чисел, оставив гипотезу Эрдёша—Турана по-прежнему открытой.

Доказательство Грина и Тао опиралось на результат, который на первый взгляд кажется неприменимым, а именно, теорему Семереди — ослабленный вариант гипотезы Эрдёша—Турана: подмножество $S$ множества натуральных чисел, содержащее положительную долю натуральных чисел (т. е. limsup отношения числа элементов множеств $S \cap [1, x]$ и $\{1, 2, \ldots, \lfloor x \rfloor\}$ есть положительное число) непременно содержит арифметические прогрессии сколь угодно большой длины. В самом деле, этот результат не применим непосредственно, поскольку множество простых чисел *не* содержит положительную долю натуральных чисел. Однако Грину и Тао удалось доказать вариант теоремы Семереди, в котором вся совокупность натуральных чисел может быть заменена на некоторые другие множества. Далее они переходят к заданию подходящего множества, содержащего простые числа, для которого справедлив аналог теоремы Семереди, и в котором простые составляют положительную долю. В целом, техника и подход Грина—Тао потрясает.

## 1.2. Знаменитые гипотезы и загадки

Мы уже отмечали, что несмотря на всю простоту определения простого числа, вопросы, связанные с простыми числами, могут быть необыкновенно сложными. В этом разделе мы коснемся ряда знаменитых исторических загадок, поиск

ключей к которым приводит к неожиданным открытиям глубины тех игр, в которые вовлечены простые числа.

### 1.2.1. Простые близнецы

Простыми близнецами называют пару простых чисел, разность которых равна 2. Ими являются, например, пары чисел 11, 13 и 197, 199. Не столь просто, но все-таки возможно, найти сравнительно большие пары. Наибольшими из известных являются пары

$$835335 \cdot 2^{39014} \pm 1$$

(найдена в 1998 г. Боллинджером и Галó),

$$361700055 \cdot 2^{39020} \pm 1$$

(открыта в 1999 г. Лифшицем),

$$2409110779845 \cdot 2^{60000} \pm 1$$

(получена в 2000 г. Вассингом, Джараем и Индлекофером) и

$$665551035 \cdot 2^{80025} \pm 1$$

(найдена П. Кармоди) (см. [Caldwell 1999]), а текущий рекорд

$$2003663613 \cdot 2^{195000} \pm 1$$

объявлен на сайте Колдвелла `http://primes.utm.edu`.

Бесконечно ли множество пар простых близнецов? Можем ли мы высказать асимптотическое предположение о количестве таких чисел, не превосходящих заданной границы? Давайте попробуем провести эвристические рассуждения в стиле юного Гаусса, который, как мы помним, на основе того соображения, что вероятность случайного числа $\approx x$ оказаться простым примерно равна $1/\ln x$, выдвинул гипотезу, что $\pi(x) \approx \int_2^x dt/\ln t$ (см. разд. 1.1.5). Давайте теперь выберем *два* натуральных числа $\approx x$. Если бы события, состоящие в том, что каждое из этих двух чисел — простое, были независимыми, то вероятность получить пару простых чисел приблизительно равнялась бы $1/\ln^2 x$. Следовательно, если для подсчета пар простых близнецов ввести функцию

$$\pi_2(x) = \#\{p \leqslant x \ : \ p, p+2 \in \mathcal{P}\},$$

где через $\mathcal{P}$ обозначено множество всех простых чисел, то можно высказать такое предположение:

$$\pi_2(x) \sim \int_2^x \frac{1}{\ln^2 t}\, dt.$$

Однако было бы несколько неосмотрительным считать факты простоты чисел $p$ и $p+2$ независимыми друг от друга. В самом деле, если одно из этих чисел простое, то шансы второго оказаться простым отличны от этих шансов в том случае, если первое число составное. Например, поскольку все простые числа $p > 2$ нечетны, число $p+2$ также нечетно, а следовательно, имеет «преимущество» оказаться простым: у случайного нечетного числа шансы оказаться

простым в два раза выше, чем у случайного числа, четность которого заранее не известна. Но проверка на четность является лишь первым из серии «тестов», которую должно пройти потенциальное простое число: для каждого фиксированного простого числа $q$, не превосходящего тестируемого числа, проверяемое число должно выдержать «$q$-тест», т. е. на выходе теста должен быть получен ответ «не кратно $q$». Для случайного простого числа $p$ и рассматриваемого $q > 2$ вероятность того, что $p+2$ успешно пройдет $q$-тест, равна $(q-2)/(q-1)$. В самом деле, согласно соотношениям (1.5), существует в точности $\varphi(q) = q-1$ классов вычетов по модулю $q$, попадание в каждый из которых числа $p$ равновероятно, и для ровно $q-2$ из них число $p+2$ не кратно $q$. Но поскольку вероятность того, что абсолютно случайное натуральное число выдержит $q$-тест равна $(q-1)/q$, мы должны уточнить выдвинутую выше гипотезу, введя корректирующую постоянную $2C_2$, где $C_2 = 0.6601618158\ldots$ — так называемая «постоянная простых близнецов»:

$$C_2 = \prod_{2<q\in\mathcal{P}} \frac{(q-2)/(q-1)}{(q-1)/q} = \prod_{2<q\in\mathcal{P}} \left(1 - \frac{1}{(q-1)^2}\right). \tag{1.6}$$

Итак, наша гипотеза такова:

$$\pi_2(x) \sim 2C_2 \int_2^x \frac{1}{\ln^2 t}\, dt, \tag{1.7}$$

или, проще говоря,

$$\pi_2(x) \sim 2C_2 \frac{x}{\ln^2 x}.$$

Эти асимптотические соотношения равносильны, в чем можно убедится интегрированием по частям. Причиной, по которой мы привели менее складную формулу (1.7), является то обстоятельство, что это приближение, как и $\pi(x) \approx \operatorname{li}(x)$, может оказаться чрезвычайно точным.

И в самом деле, давайте испытаем наше приближение (1.7) при $x = 5.4\cdot 10^{15}$. Согласно статье [Nicely 2004],

$$\pi_2\left(5.4\cdot 10^{15}\right) = 5761178723343,$$

в то время как

$$2C_2 \int_2^{5\,4\,10^{15}} \frac{1}{\ln^2 t}\, dt \approx 5761176717388.$$

Это можно назвать эвристическими рассуждениями в действии! Недавно П. Себа нашел, что

$$\pi_2\left(10^{16}\right) = 10304195697298,$$

как следует из работы [Gourdon and Sebah 2004].

Но какими бы убедительными не были численные свидетельства, мы до сих пор не знаем даже того, существует ли бесконечно много пар простых близнецов, т. е. является ли функция $\pi_2(x)$ неограниченной. Эта задача остается одной из величайших нерешенных математических проблем. Ближе всех к решению этой проблемы подошел Чен Джин-ран, доказавший в 1966 г., что существует бесконечное множество простых чисел $p$, таких, что число $p+2$

или простое, или равно произведению двух простых чисел (см. [Halberstam and Richert 1974]).

Выдающийся результат, связанный с простыми близнецами, получил в 1915 г. В. Брун. Он доказал верхнюю оценку вида

$$\pi_2(x) = O\left(x\left(\frac{\ln\ln x}{\ln x}\right)^2\right), \tag{1.8}$$

причем через год ему удалось заменить $\ln\ln x$ на 1 (см. [Halberstam and Richert 1974]). Таким образом, можно сказать, что гипотеза о простых близнецах (1.7) частично доказана. Из равенства (1.8) можно вывести следующий факт (см. упр. 1.50).

**Теорема 1.2.1** (Брун). *Сумма всех чисел, обратных к простым близнецам, конечна. Иными словами, если обозначить через $\mathcal{P}_2$ множество всех простых чисел $p$, таких, что число $p+2$ также простое, то будет выполнено неравенство*

$$\sum_{p\in P_2}\left(\frac{1}{p}+\frac{1}{p+2}\right) < \infty.$$

(Отметим, что существует только одно простое число, обладающее тем свойством, что существуют *две* пары простых близнецов, его содержащих: лишь число 5 *дважды* учитывается в выписанной сумме. Разумеется, этот факт не оказывает никакого влияния на сходимость суммы.) Теорема Бруна замечательна еще и тем, что, как мы знаем, ряд, составленный из всех чисел, обратных к простым, расходится (пусть и медленно, см. разд. 1.1.5). Численное значение суммы из теоремы Бруна

$$B' = (1/3+1/5)+(1/5+1/7)+(1/11+1/13)+\ldots,$$

называется *постоянной Бруна*. Таким образом, хотя множество простых близнецов может оказаться и бесконечным, можно с уверенностью утверждать, что это множество является значительно более разреженным, чем множество всех простых чисел.

Любопытной стороной вопроса о распределении простых близнецов является вычисление постоянной Бруна $B'$. Эта тема имеет долгую историю, а в наши дни наилучший результат здесь принадлежит Найсли: согласно статье [Nicely 2004], $B'$ *вероятно* приблизительно равна 1.902160583. Этот результат был получен путем очень точного вычисления суммы обратных величин к простым близнецам, не превосходящим $10^{16}$, и экстраполяции на оставшийся бесконечный промежуток с использованием соотношения (1.7) для оценки остатка ряда. (Необходимо отметить, что для постоянной $B'$ строго доказано лишь то, что она больше некоторого числа, немного превосходящего 1.83, и меньше определенного числа, несколько меньшего, чем 2.347.) Производя вычисления в начале своей работы над постоянной Бруна (1995 г.), Найсли обнаружил ныне известную ошибку в арифметике с плавающей точкой процессора Pentium, по некоторым расчетам стоившую производителю Pentium'а Intel'у миллионы долларов. Авторы высказывают предположение, что В. Брун

в 1915 г. и представить себе не мог, какие технологические последствия может вызвать его выдающаяся теорема!

### 1.2.2. $k$-кортежи простых и гипотеза H

Гипотеза о простых близнецах является частным случаем так называемой гипотезы о «$k$-кортежах простых чисел», которая, в свою очередь, есть частный случай «гипотезы H». Раскроем смысл этих загадочно звучащих названий.

Гипотеза о «$k$-кортежах простых чисел» начинается с вопроса о том, какие условия достаточно наложить на целые числа $a_1, b_1, \ldots, a_k, b_k$ для того, чтобы $k$ линейных выражений $a_1 n + b_1, \ldots, a_k n + b_k$ одновременно равнялись простым числам для бесконечного множества натуральных значений $n$. Как видно, этот вопрос включает в себя первую часть теоремы Дирихле 1.1.5, соответствующую случаю $k = 1$, а также гипотезу о простых близнецах, в которой рассматривается пара линейных выражений $n, n + 2$.

Попытаемся начать ответ на поставленный вопрос с изучения необходимых условий, которым должны удовлетворять числа $a_i, b_i$. Исключим из рассмотрения те случаи, когда найдется $a_i = 0$, поскольку тогда задача сводится к более простой проблеме. Ясно, что мы должны потребовать выполнения условий $a_i > 0$ и НОД $(a_i, b_i) = 1$ для всех значений индекса $i$. Однако пример пары $n, n + 1$ показывает, что этого недостаточно: существует лишь конечное множество натуральных чисел $n$, для которых числа $n$ и $n + 1$ одновременно просты! В этом примере простое число 2 лишает пару чисел $n$ и $n + 1$ шансов оказаться простыми (кроме случая $n = 2$), поскольку одно из двух последовательных чисел обязательно четно, а 2 — единственное четное простое число. Обобщая этот пример, мы видим, что еще одно необходимое условие состоит в том, чтобы для каждого простого числа $p$ нашлось такое значение $n$, для которого ни одно из чисел $a_i n + b_i$ не делится на $p$. Это условие автоматически выполняется для всех простых чисел $p > k$ и, кроме того, из него следует, что НОД $(a_i, b_i) = 1$ для всех значений индекса $i$. Гипотеза о $k$-кортежах простых чисел (см. [Dickson 1904]) утверждает, что рассмотренные нами условия являются достаточными.

**Гипотеза 1.2.1** (гипотеза о $k$-кортежах простых чисел). *Пусть целые числа $a_1, b_1, \ldots, a_k, b_k$ таковы, что $a_i > 0$ и НОД $(a_i, b_i) = 1$ для всех значений индекса $i$. Пусть, далее, для каждого простого числа $p \leqslant k$ существует такое число $n$, для которого ни одно их чисел $a_i n + b_i$ не кратно $p$. Тогда числа $a_i n + b_i$ являются одновременно простыми для бесконечного множества натуральных значений $n$.*

В то время как гипотеза о $k$-кортежах простых чисел связана с линейными выражениями, выдвинутая Шинцелем (см. [Schinzel and Sierpiński 1958]) гипотеза H затрагивает произвольные неприводимые многочлены с целыми коэффициентам и является обобщением более раннего предположения Буняковского, высказанного касательно единственного неприводимого многочлена.

**Гипотеза 1.2.2** (гипотеза H). *Пусть неприводимые целочисленные многочле-*

*ны $f_1, \dots, f_k$ имеют положительные старшие коэффициенты и для каждого простого числа $p$ существует такое число $n$, что ни одно из значений $f_1(n), \dots, f_k(n)$ не кратно $p$. Тогда числа $f_i(n)$ являются одновременно простыми для бесконечного множества натуральных значений $n$.*

Знаменитый частный случай гипотезы H связан с многочленом $n^2+1$. Как и в случае простых близнецов, нам неизвестно, бесконечно ли множество простых чисел вида $n^2+1$. Вообще говоря, единственным доказанным частным случаем гипотезы H является теорема Дирихле 1.1.5!

Метод, которым воспользовался В. Брун для доказательства формулы (1.8), может быть обобщен для получения верхних оценок функций, соответствующих числу значений параметра $n$ в гипотезе H, для которых все числа $f_i(n)$ одновременно просты (по этому вопросу см. [Halberstam and Richert 1974]).

О многочленах от двух переменных известно несколько больше. Например, Гаусс доказал, что бесконечно много простых чисел представимы в виде $a^2+b^2$, а не так давно в статье [Friedlander and Iwaniec 1998] было доказано аналогичное утверждение для многочлена $a^2+b^4$.

## 1.2.3. Проблема Гольдбаха

В 1742 г. К. Гольдбах в письме Л. Эйлеру высказал уверенность в том, что каждое натуральное число, превосходящее 5, представимо в виде суммы трех простых чисел, например, $6 = 2+2+2$ и $21 = 13+5+3$. Эйлер ответил, что это утверждение следует из того, что каждое четное число, большее двух, представимо в виде суммы двух простых чисел. Впоследствии последнее утверждение получило имя проблемы (или гипотезы) Гольдбаха. Эта гипотеза относится исключительно к сфере задач аддитивной теории чисел, изучающей способы представления целых чисел в виде различных сумм. Описанная проблема, как и ряд подобных аддитивных задач, обладает тем поразительным свойством, что существует огромное множество численных примеров и эвристических соображений в ее пользу. В частности, даже большие четные числа допускают разнообразные представления в виде суммы двух простых.

Обозначим число гольдбаховых представлений четного числа $n$ через

$$R_2(n) = \#\{(p,q) \ : \ n = p+q;\ p,q \in \mathcal{P}\}.$$

При помощи эвристических рассуждений, аналогичных приведенным выше, можно предположить, что для четных значений $n$ справедливо соотношение

$$R_2(n) \sim \sum_{p \leqslant n-3} \frac{1}{\ln(n-p)},$$

поскольку «вероятность» того, что случайное число $\approx x$ окажется простым, примерно равна $1/\ln x$. Но при помощи теоремы Чебышёва 1.1.3 можно показать, что эта сумма $\sim n/\ln^2 n$ (см. упр. 1.40). Еще более печальным является тот факт, что для разрешения проблемы Гольдбаха достаточно доказать лишь *положительность* функции $R_2(n)$ для всех четных значений $n > 2$. Можно

так же, как и при выводе соотношения (1.7), провести более точные эвристические рассуждения и предположить, что для всех четных значений $n > 2$

$$R_2(n) \sim 2C_2 \frac{n}{\ln^2 n} \prod_{p \mid n,\, p>2} \frac{p-1}{p-2}, \tag{1.9}$$

где $C_2$ — постоянная простых близнецов, как и в формуле (1.6). Метод Бруна позволяет установить, что $R_2(n)$ есть $O$ от правой части соотношения (1.9) (см. [Halberstam and Richert 1974]).

Вычисления показывают, что $R_2(10^8) = 582800$, в то время как правая часть формулы (1.9) приблизительно равна 518809. Более точное приближение получается, если использовать асимптотически эквивалентное представление

$$\mathcal{R}_2(n) = 2C_2 \int_2^{n-2} \frac{dt}{(\ln t)(\ln(n-t))} \prod_{p \mid n,\, p>2} \frac{p-1}{p-2}, \tag{1.10}$$

которое при $n = 10^8$ приближенно дает 583157.

Как и в случае простых близнецов, Чен (см. [Chen 1966]) доказал выдающуюся теорему, связанную с проблемой Гольдбаха: каждое достаточно большое четное число представимо в виде суммы простого числа и числа, являющегося или простым, или произведением двух простых.

С конца 1930-х гг. известно, что почти все четные числа имеют гольдбахово представление $p + q$ (см. [Ribenboim 1996]), причем термин «почти все» означает, что множество исключительных четных чисел, т. е. тех, которые *не* допускают представления в виде двух простых, имеет асимптотическую плотность 0 (определение асимптотической плотности см. в разд. 1.1.4). Более того, можно показать, что количество исключительных четных чисел, не превосходящих $x$, есть $O\left(x^{1-c}\right)$ для некоторой постоянной $c > 0$ (см. упр. 1.41).

Гипотеза Гольдбаха проверена на ЭВМ для всех четных чисел до $10^{14}$ (см. [Deshouillers et al. 1998]), до $4 \cdot 10^{14}$ (см. [Richstein 2001]) и затем до $10^{17}$ (см. [Silva 2005]). Вычисления показали, что и в самом деле каждое четное число от 4 до $10^{17}$ включительно представимо в виде суммы двух простых чисел.

Как отметил еще Эйлер, следствием гипотезы Гольдбаха является представимость каждого нечетного числа, большего пяти, в виде суммы трех простых чисел. Это следствие известно как тернарная проблема Гольдбаха. Несмотря на всю сложность подобных задач аддитивной теории чисел, в 1937 г. И. М. Виноградов решил тернарную проблему Гольдбаха, доказав, что всякое достаточно большое нечетное число $n$ представимо в виде суммы трех простых чисел $n = p + q + r$. Как показали в 1989 г. Чен и Ванг, слова «достаточно большое» можно заменить на неравенство $n > 10^{43000}$ (см. [Ribenboim 1996]). И. М. Виноградов получил также и асимптотическую формулу для функции

$$R_3(n) = \#\{(p,q,r) \ : \ n = p+q+r;\, p,q,r \in \mathcal{P}\} \tag{1.11}$$

вида

$$R_3(n) = \Theta(n) \frac{n^2}{2\ln^3 n} \left(1 + O\left(\frac{\ln\ln n}{\ln n}\right)\right), \tag{1.12}$$

где $\Theta$ есть так называемый сингулярный ряд тернарной проблемы Гольдбаха:

$$\Theta(n) = \prod_{p\in\mathcal{P}} \left(1 + \frac{1}{(p-1)^3}\right) \prod_{p\,|\,n, p\in\mathcal{P}} \left(1 - \frac{1}{p^2 - 3p + 3}\right).$$

Нетрудно видеть, что функция $\Theta(n)$ при нечетных значениях $n$ ограничена снизу некоторой положительной постоянной. Сингулярный ряд допускает и еще несколько любопытных альтернативных записей (см. упр. 1.68). И. М. Виноградов внес неоценимый вклад в развитие аналитической теории чисел (очень краткий обзор его результатов содержится в разд. 1.4.4).

Д. Зиновьев (см. [Zinoviev 1997]) показал, что из расширенной гипотезы Римана (ERH) (гипотезы 1.4.2) следует разрешимость тернарной проблемы Гольдбаха для всех нечетных значений $n > 10^{20}$. С другой стороны, Саутер (см. [Saouter 1998]), отталкиваясь от текущей границы $4 \cdot 10^{11}$ для бинарной проблемы Гольдбаха, проверил утверждение тернарной проблемы Гольдбаха для всех нечетных чисел вплоть до $10^{20}$ (этот результат не является условным). Таким образом, сочетание этих двух результатов позволяет утверждать полную разрешимость тернарной проблемы Гольдбаха при условии выполнения ERH.

Из теоремы Виноградова следует существование такого числа $k$, что всякое число, начиная с 2, является суммой не более чем $k$ простых чисел. Это следствие было доказано ранее Л. Г. Шнирельманом совершенно иным способом. Шнирельман применил метод решета Бруна и показал, что множество четных чисел, представимых в виде суммы двух простых, содержит подмножество, имеющее положительную асимптотическую плотность (это предшествовало упомянутой выше теореме о такой представимости почти всех четных чисел), и на основе лишь этого факта доказал существование числа $k$ с нужным свойством (подробности см. в упр. 1.44). Наименьшее число $k_0$, такое, что каждое натуральное число, начиная с 2, представимо в виде суммы не более, чем $k_0$ простых, носит имя постоянной Шнирельмана. Из гипотезы Гольдбаха следует, что $k_0 = 3$. Но поскольку в наши дни решенной является лишь тернарная проблема, предположив справедливость ERH, можно показать, что $k_0 \leqslant 4$. Наилучшим безусловным результатом в настоящее время является неравенство $k_0 \leqslant 7$, принадлежащее Рамарэ (см. [Ramaré 1995]) (часть работы выполнена совместно с Румели), которое содержит изрядный объем теоретико-числовых вычислений.

## 1.2.4. Проблема выпуклости

Богатым источником возникновения разнообразных гипотез, связанных с простыми числами, является теоретический вопрос о плотности их распределения (например, на заданных промежутках или при определенных ограничениях). Существуют ли такие числовые промежутки, на которых простые числа встречаются особенно часто? Или особенно редко? Иногда возникают удивительные дилеммы, как, например, такая. Много лет назад Харди и Литтлвуд высказали следующую гипотезу о «выпуклости» распределения простых чисел.

**Гипотеза 1.2.3.** *Для целых чисел $x \geqslant y \geqslant 2$ мы имеем $\pi(x+y) \leqslant \pi(x)+\pi(y)$.*

На первый взгляд, это предположение кажется логичным. В самом деле, поскольку при движении вперед по натуральному ряду простые числа встречаются все реже и реже, их количество на промежутке $[x, x+y]$ должно быть меньше, чем на промежутке $[0, y]$. Но, тем не менее, и это поразительно, доказано, что гипотеза 1.2.3 *не* совместима с гипотезой о $k$-кортежах простых чисел 1.2.1 (см. [Hensley and Richards 1973]).

Итак, какая же из гипотез верна? Возможно, что и никакая, но в наши дни преобладает та точка зрения, что предположение Харди—Литтлвуда является ложным, а гипотеза о $k$-кортежах простых чисел имеет место. Может показаться, что опровергнуть гипотезу о выпуклости очень просто: достаточно лишь предъявить пару целых чисел $x$ и $y$, для которых выполняется неравенство $\pi(x+y) > \pi(x)+\pi(y)$. Но дело обстоит так, что, по-видимому, значение $x$, требуемое для опровержения этой гипотезы, должно быть огромным (по этому вопросу см. упр. 1.92).

## 1.2.5. Формула для простых чисел

Поиск формулы, задающей простые числа, стал популярным развлечением с тех пор, как Эйлер обратил внимание на то обстоятельство, что многочлен

$$x^2 + x + 41$$

принимает простые значения для всех целых чисел $x$ от 0 до 39 включительно. Вооружившись современной вычислительной техникой, сегодня можно для различных многочленов эмпирически анализировать вероятность принять простое значение на разнообразных промежутках (см. упр. 1.17).

Приведем несколько примеров, вызывающих неоднозначную оценку у специалистов по вычислениям (тем не менее, см. упр. 1.5 и 1.77).

**Теорема 1.2.2** (примеры формул для простых чисел). *Существует такое действительное число $\theta > 1$, что для каждого натурального значения $n$ число*

$$\left\lfloor \theta^{3^n} \right\rfloor$$

*является простым. Существует также такое действительное число $\alpha$, что $n$-е простое число представимо в виде*

$$p_n = \left\lfloor 10^{2^{n+1}} \alpha \right\rfloor - 10^{2^n} \left\lfloor 10^{2^n} \alpha \right\rfloor .$$

Первое из этих утверждений следует из нетривиальной теоремы о распределении простых чисел в «коротких» интервалах (см. [Mills 1947]), а вторая формула получается из рассмотрения равенства $\alpha = \sum p_m 10^{-2^{m+1}}$ (ряд, очевидно, сходится).

Подобные формулы, даже когда являются тривиальными (или почти тривиальными), могут быть весьма колоритны. Из теоремы Вильсона и обратной к ней (теорема 1.3.6) можно вывести, что если $n$ — наибольшее натуральное

число, не превосходящее $x$, то

$$\pi(x) = \sum_{j=2}^{n} \left( \left\lfloor \frac{(j-1)!+1}{j} \right\rfloor - \left\lfloor \frac{(j-1)!}{j} \right\rfloor \right).$$

Правда, эта формула не оказала существенного влияния на развитие теории функции $\pi(x)$. Тем не менее, в упражнениях в конце настоящей главы можно найти еще несколько формул как для простых чисел, так и для функции $\pi(x)$.

Несмотря на свою красоту, формулы для простых чисел зачастую совершенно бесполезны. Однако есть знаменитое исключение: в связи с окончательным решением десятой проблемы Гильберта, заключавшейся в нахождении детерминированного алгоритма, позволяющего определить, имеет ли данный многочлен от нескольких переменных с целыми коэффициентами хотя бы один целочисленный корень, был получен интересный побочный результат, а именно, был построен такой многочлен от нескольких переменных с целыми коэффициентами, что множество его *положительных* значений на множестве натуральных аргументов в точности совпадает со множеством всех простых чисел (см. разд. 8.4).

## 1.3. Простые числа специального вида

Под простыми числами специального вида мы понимаем такие простые числа, которые можно представить в определенном, зачастую элегантном, алгебраическом виде. Например, числа Мерсенна $M_q$ и числа Ферма $F_n$, имеющие вид

$$M_q = 2^q - 1, \quad F_n = 2^{2^n} + 1,$$

иногда оказываются простыми. Такие числа интересны как сами по себе, так и своей историей, поскольку их изучение послужило мощным стимулом для развития вычислительной теории чисел.

### 1.3.1. Числа Мерсенна

Нахождение простых чисел Мерсенна можно назвать исследовательской проблемой (или, быть может, развлечением) с вековой историей. Нетрудно получить ряд ограничений на показатель $q$, облегчающих поиск простых чисел вида $M_q = 2^q - 1$. Начнем со следующего факта.

**Теорема 1.3.1.** *Если число $M_q = 2^q - 1$ является простым, то число $q$ также простое.*

Доказательство. Если число $c$ является составным, то число $2^c - 1$ имеет собственный делитель вида $2^d - 1$, где $d$ — произвольный собственный делитель числа $c$. □

Отсюда следует, что при поиске простых чисел Мерсенна можно ограничиться лишь простыми значениями показателя $q$. Важно отметить, что утверждение, обратное к доказанной теореме, неверно. Например, число $2^{11} - 1$ составное, хотя 11 — простое число. Тем не менее, эта теорема позволяет суще-

| | | | |
|---|---|---|---|
| $2^2-1$ | $2^3-1$ | $2^5-1$ | $2^7-1$ |
| $2^{13}-1$ | $2^{17}-1$ | $2^{19}-1$ | $2^{31}-1$ |
| $2^{61}-1$ | $2^{89}-1$ | $2^{107}-1$ | $2^{127}-1$ |
| $2^{521}-1$ | $2^{607}-1$ | $2^{1279}-1$ | $2^{2203}-1$ |
| $2^{2281}-1$ | $2^{3217}-1$ | $2^{4253}-1$ | $2^{4423}-1$ |
| $2^{9689}-1$ | $2^{9941}-1$ | $2^{11213}-1$ | $2^{19937}-1$ |
| $2^{21701}-1$ | $2^{23209}-1$ | $2^{44497}-1$ | $2^{86243}-1$ |
| $2^{110503}-1$ | $2^{132049}-1$ | $2^{216091}-1$ | $2^{756839}-1$ |
| $2^{859433}-1$ | $2^{1257787}-1$ | $2^{1398269}-1$ | $2^{2976221}-1$ |
| $2^{3021377}-1$ | $2^{6972593}-1$ | $2^{13466917}-1$ | $2^{20996011}-1$ |
| $2^{24036583}-1$ | $2^{25964951}-1$ | | |

**Таблица 1.2.** Найденные простые числа Мерсенна (на апрель 2005 г.). Максимальное из них состоит более чем из 7 миллионов знаков.

ственно сократить работу при нахождении простых чисел Мерсенна, отбросив огромное количество составных показателей.

Еще большую помощь для отсеивания лишних показателей при поиске простых чисел Мерсенна оказывает следующее утверждение о простых делителях числа $M_q$.

**Теорема 1.3.2** (Эйлер). *Пусть $q > 2$ — простое число и $r$ — произвольный простой делитель числа $M_q = 2^q - 1$. Тогда $r \equiv 1 \pmod q$ и, кроме того, $r \equiv \pm 1 \pmod 8$.*

Доказательство. Пусть числа $q$ и $r$ удовлетворяют условиям теоремы. Тогда $2^q \equiv 1 \pmod r$, а поскольку число $q$ — простое, наименьшим положительным показателем $h$, для которого $2^h \equiv 1 \pmod r$, должно быть само число $q$. Следовательно, в мультипликативной группе ненулевых вычетов по модулю $r$ (ее порядок равен $r-1$) элемент 2 имеет порядок $q$. Отсюда немедленно вытекает, что $r \equiv 1 \pmod q$, так как порядок элемента группы делит порядок самой группы. Поскольку простое число $q$ нечетно, мы получаем, что $q \mid \frac{r-1}{2}$, а следовательно, $2^{\frac{r-1}{2}} \equiv 1 \pmod r$. Согласно критерию Эйлера (2.6), число 2 является квадратичным вычетом по модулю $r$, а это, в свою очередь, с учетом (2.10), означает, что $r \equiv \pm 1 \pmod 8$. □

Итак, стандартный поиск простых чисел Мерсенна происходит следующим образом. Из заданного множества простых показателей $Q$ исключаем элементы $q \in Q$, проверяя сравнения

$$2^q \equiv 1 \pmod r$$

для последовательности малых простых чисел $r \equiv 1 \pmod q$ и $r \equiv \pm 1 \pmod 8$. К оставшимся элементам применяем известный тест Люка—Лемера, являющийся строгим тестом на простоту (см. разд. 4.2.1).

Все известные на настоящий момент простые числа Мерсенна указаны в табл. 1.2.

За период с 1979 г. по 1996 г. Словински нашел семь простых чисел Мерсен-

на, от $2^{44497}-1$ до $2^{1257787}-1$ включительно, за исключением числа $2^{110503}-1$ (первое из семи чисел было найдено совместно с Нельсоном, а последние три — с Гейджем). «Недостающее» простое число $2^{110503}-1$ обнаружили Колквит и Уэльш мл. в 1988 г. Рекорд открытия *последовательных* простых чисел Мерсенна до сих пор принадлежит Робинсону, который в 1952 г. нашел пятерку, начиная с $2^{521}-1$. Число $2^{1398269}-1$ было найдено в 1996 г. Арменго и Вольтманом, а $2^{2976221}-1$ — в 1997 г. Спенсом и Вольтманом. Простое число $2^{3021377}-1$ нашли в 1998 г. Кларксон, Вольтман, Куровски и др., а позднее независимым способом его проверил Словински. Далее, в 1999 г. Хайратвала, Вольтман и Куровски обнаружили $2^{6972593}-1$ (проверено Майером). Простота $2^{13466917}-1$ установлена в ноябре 2001 г. Камероном, Вольтманом и Куровски и подтверждена Майером, Новарезе и Валором. В ноябре 2003 г. Шафер, Вольтман и Куровски распознали $2^{20996011}-1$. Простое число Мерсенна $2^{24036583}-1$ нашли в мае 2004 г. Дж. Файндли, Вольтман и Куровски. Затем, в феврале 2005 г. М. Новак, Вольтман и Куровски обнаружили $2^{25964951}-1$. Каждое из двух последних простых чисел Мерсенна содержит более 7 миллионов знаков.

Последние восемь чисел таблицы 1.2 были найдены посредством метода быстрого умножения ВДПИО (см. гл. 9, теорему 9.5.18 и алгоритм 9.5.19). Этот алгоритм более чем в два раза повысил эффективность поиска по сравнению с прежними методами.

Можно сказать, что современный поиск простых чисел Мерсенна схож со стрельбой по мишени: для проверки случайных кандидатов $2^q-1$ используют соответствующие случайные показатели $q$. И в самом деле, как мы видели выше, некоторые простые числа Мерсенна были найдены «вне очереди». Но производится также и систематическое тестирование. На момент написания книги были проверены все показатели $q \leqslant 12830000$. Многие показатели соответствуют составным числам Мерсенна, причем в этом случае можно указать и некоторый простой делитель. Например, если простое число $q$ сравнимо с 3 (mod 4), а число $p=2q+1$ — простое, то $p \mid M_q$ (см. также упр. 1.47, 1.81.) Для оставшихся значений $q$ был применен тест Люка—Лемера (см. разд. 4.2.1). Более того, все показатели $q \leqslant 9040000$, для которых не удалось найти делитель числа $M_q$, были пропущены через тест Люка—Лемера дважды (актуальная информация размещается на сайте [Woltman 2000]).

Как отмечалось в разд. 1.1.2, простое число $M_{25964951}$ является не только наибольшим известным на сегодняшний день простым числа Мерсенна, но и наибольшим из найденных явно простых чисел вообще. За редкими исключениями рекорд максимального открытого простого числа всегда принадлежал именно простым числам Мерсенна. Одним из таких исключений было найденное в 1989 г. «Амдальской шестеркой» число

$$391581 \cdot 2^{216193}-1,$$

превосходящее рекордное на тот момент простое число Мерсенна $2^{216091}-1$ (см. [Caldwell 1999]). Отметим, что в 1997 г. Янг обнаружил еще большее не мерсенновское простое число

$$5 \cdot 2^{240937}+1,$$

а в 2001 г. Косгрейв доказал простоту числа

$$3 \cdot 2^{916773} + 1.$$

Сегодня пятое по величине явное простое число

$$5359 \cdot 2^{5054502} + 1$$

(найдено Сандквистом в 2003 г.) — не мерсенновское.

Существует знаменитая связь между простыми числами Мерсенна и рассматриваемыми еще с древности так называемыми совершенными числами. Совершенным называется такое натуральное число, которое совпадает с суммой всех своих положительных делителей, не равных самому этому числу. Например, числа $6 = 1+2+3$ и $28 = 1+2+4+7+14$ являются совершенными. Еще один способ определения совершенного числа дает равенство $\sigma(n) = 2n$, где через $\sigma(n)$ обозначена сумма всех делителей числа $n$.

Следующая известная теорема позволяет получить полное описание всех четных совершенных чисел.

**Теорема 1.3.3** (Евклид—Эйлер). *Для того чтобы четное число $n$ было совершенным, необходимо и достаточно, чтобы оно имело вид*

$$n = 2^{q-1} M_q,$$

*где $M_q = 2^q - 1$ — простое число.*

Доказательство. Пусть $n = 2^a m$ — четное число, причем $m$ — его наибольший нечетный делитель. Тогда все делители числа $n$ представимы в виде $2^j d$, где $0 \leqslant j \leqslant a$ и $d \mid m$. Обозначим через $D$ сумму всех делителей числа $m$, за исключением $m$, и положим $M = 2^{a+1} - 1 = 2^0 + 2^1 + \ldots + 2^a$. Тогда сумма всех делителей числа $n$ равна $M(D+m)$. Если число $M$ простое и $M = m$, то $D = 1$, а значит, сумма всех делителей числа $n$ равна $M(1+m) = 2n$, так что $n$ — совершенное число. Тем самым, достаточность доказана.

Пусть теперь число $n = 2^a m$ является совершенным. Тогда

$$M(D+m) = 2n = 2^{a+1} m = (M+1)m.$$

Вычитая из крайних частей этих равенств $Mm$, получаем $m = MD$. Если $D > 1$, то числа $D$ и 1 являются различными делителями числа $m$, меньшими $m$, что противоречит определению числа $D$. Следовательно, $D = 1$, число $m$ простое и $m = M = 2^{a+1} - 1$. □

Достаточность была впервые доказана Евклидом, а необходимость установил спустя примерно две тысячи лет Эйлер. Очевидно, что каждое новое открытое простое число Мерсенна немедленно порождает и новое (четное) совершенное число. С другой стороны, до сих пор неизвестно, существует ли хотя бы *одно* нечетное совершенное число. На сегодняшний день принято давать отрицательный ответ на этот вопрос. Бóльшая часть исследований в этой области носит чисто вычислительный характер. В частности, известно, что если нечетное совершенное число $n$ существует, то оно должно удовлетворять неравенству $n > 10^{300}$ (см. [Brent et al. 1993]) и иметь не менее восьми различных простых делителей. Последний результат получен независимо Чейном и

Хеджисом [Ribenboim 1996]. К вопросу о совершенных числах мы вернемся в упр. 1.30.

С простыми числами Мерсенна связано множество интереснейших открытых проблем. Например, нам до сих пор неизвестно, бесконечно их множество или нет. Более того, мы даже не знаем, бесконечно ли множество *составных* чисел Мерсенна $M_q$, где $q$ — простое число. Однако из гипотезы о $k$-кортежах простых чисел 1.2.1 вытекает положительный ответ на последний вопрос. В самом деле, нетрудно видеть, что если числа $q \equiv 3 \pmod 4$ и $2q+1$ являются простыми, то $2q+1$ делит $M_q$. Например, 23 делит $M_{11}$. А гипотеза 1.2.1 утверждает существование бесконечного множества таких чисел $q$.

Еще одна проблема, так называемая «обновленная гипотеза Мерсенна», была выдвинута Бейтманом, Селфриджем и Вагштафом. Она берет начало от высказанного в 1644 г. утверждения самого Мерсенна о том, что показателями $q$, для которых число $2^q - 1$ простое и $29 \leqslant q \leqslant 257$, являются числа 31, 67, 127 и 257. (К тому времени меньшие показатели были уже изучены, а также было известно, что число $2^{37} - 1$ является составным.) Поразительно, как Мерсенн, имея численные данные о том, что каждое простое число, меньшее 29, за исключением 11 и 23, порождает простое число Мерсенна, пришел к выводу о столь разреженной последовательности показателей и оказался прав: показатели и в самом деле достаточно редки, однако, истинная их последовательность такова: 31, 61, 89, 107 и 127. Сегодня никто не может с уверенностью объяснить, как, совершив лишь пять ошибок, Мерсенн смог выдвинуть свою гипотезу. Однако было замечено, что нечетные показатели Мерсенна, меньшие 29, обладают следующим свойством: они или отстоят на 1 от степени 2, или на 3 от степени 4 (а числа 11 и 23 этим свойством не обладают). Поэтому Мерсенн мог просто продолжить список таких чисел (возможно, число 61 было отброшено как «ошибка эксперимента»). В статье [Bateman et al. 1989] авторы высказывают обновленную гипотезу Мерсенна, состоящую в том, что любые два из следующих высказываний влекут третье:

(a) простое число $q$ или отстоит на 1 от степени 2, или на 3 от степени 4;
(b) $2^q - 1$ — простое число;
(c) $(2^q + 1)/3$ — простое число.

Среди небольших чисел очень трудно найти хотя бы одно простое число $q$, удовлетворяющее каким-либо двум из этих условий, и возможно, что до 127 таких чисел нет. Таким образом, вероятно, что гипотеза верна, но пока это всего лишь утверждение, основанное на исследовании очень небольшого множества простых чисел.

Существует также гипотеза о том, что каждое число Мерсенна $M_q$, соответствующее простому показателю $q$, является бесквадратным (т. е. не делится ни на один квадрат, отличный от 1), но мы не можем доказать даже того, что это происходит бесконечно часто. Обозначим через $\mathcal{M}$ множество таких простых чисел, которые делят некоторое число $M_q$, где $q$ — простое число. Можно показать, что количество элементов множества $\mathcal{M}$, не превосходящих $x$, есть $o(\pi(x))$ и, кроме того, из расширенной гипотезы Римана (гипотеза 1.4.2) сле-

дует, что ряд, составленный из всех чисел, обратных к элементам множества $\mathcal{M}$, сходится (см. [Pomerance 1986]).

Приведем эвристические рассуждения в пользу того предположения, что существует $\sim c \ln x$ простых чисел $q \leqslant x$, для которых число $M_q$ также является простым, где $c = e^{\gamma}/\ln 2$ и $\gamma$ — постоянная Эйлера. В частности, из этой формулы вытекает, что на промежутке $[x, 10000x]$ найдется примерно 23.7 таких чисел $q$. Если считать, что достаточно численной проверки всех показателей Мерсенна вплоть до 12000000, то реальное количество рассматриваемых показателей $q$ на промежутке $[x, 10000x]$ равно 23, 24 и 25 для $x = 100, 200, \ldots, 1200$, причем чаще всех встречается значение 24. Несмотря на хорошую согласованность с результатами эксперимента, некоторые специалисты склонны считать, что истинным значением постоянной $c$ является $2/\ln 2$ или какое-то иное число. Но кто здесь оказался прав мы сможем определить, лишь когда теорема будет доказана.

Итак, начнем наши эвристические рассуждения. Для начала скажем, что согласно теореме о простых числах 1.1.4 вероятность того, что случайное натуральное число, близкое к $M_q = 2^q - 1$, окажется простым, примерно равна $1/\ln M_q$. Однако мы также должны сравнить шансы числа $M_q$ оказаться простым с шансами случайного числа того же размера. Как показывает теорема 1.3.2, мы должны получить разные значения. Отвлечемся на время от этой теоремы и будет пользоваться лишь тем свойством, что число $M_q$ не содержит простых делителей из промежутка $[1, q]$, где $q$ имеет порядок $\lg M_q$ (здесь и далее lg обозначает $\log_2$). Каковы шансы на то, что случайное число, близкое к $x$, наименьший простой делитель которого превосходит $\lg x$, окажется простым? Можно дать строгий ответ на этот вопрос. Сначала рассмотрим вероятность того, что наименьший простой делитель случайного числа, близкого к $x$, превосходит $\lg x$. Интуиция подсказывает, что она должна равняться

$$P \coloneqq \prod_{p \leqslant \lg x} \left(1 - \frac{1}{p}\right),$$

поскольку вероятность того, что простое число $p$ делит данное случайное число, равна $1/p$, а для разных простых чисел эти события, как может показаться, независимы. Однако, на самом деле, здесь независимости нет: например, ни одно число из промежутка $[1, x]$ не обладает свойством делимости на два простых числа из полуинтервала $(x^{1/2}, x]$, хотя чисто вероятностное рассуждение говорит о том, что положительная доля чисел промежутка $[1, x]$ должна обладать этим свойством! Тем не менее, для очень небольших простых чисел (в частности, как в нашем случае, меньших $\lg x$) наша эвристическая догадка доказуема. Далее, поскольку каждое простое число, близкое к $x$, не делится ни на одно простое число $p \leqslant \lg x$, оно сохранится при этом $\lg x$-просеивании. Следовательно, вероятность того, что близкое к $x$ число, которое прошло через $\lg x$-решето, окажется простым, возрастает с $1/\ln x$ до

$$\frac{1}{P \ln x}.$$

Асимптотика вероятности $P$ известна. Из теоремы Мертенса 1.4.2 вытекает, что $1/P \sim e^\gamma \ln\lg x$ при $x \to +\infty$. Таким образом, можно заключить, что число $M_q$ оказывается простым с «вероятностью» $e^\gamma \ln\lg M_q / \ln M_q$. Но это выражение очень близко́ к значению $e^\gamma \ln q/(q \ln 2)$, суммируя которое по всем простым $q \leqslant x$, мы получаем эвристическую асимптотику для количества простых показателей Мерсенна, не превосходящих $x$: $c \ln x$, где $c = e^\gamma / \ln 2$.

Если вернуться назад и начать более тонкие рассуждения, привлекая теорему 1.3.2, то потребуется учитывать не только ограничения на возможные простые делители числа $M_q$, но также и то, что простое число, удовлетворяющее условию этой теоремы, имеет повышенные шансы оказаться делителем $M_q$. Например, если $p = kq + 1$ — простое число и $p \equiv \pm 1 \pmod 8$, то можно показать, что вероятность события $p \mid M_q$ равна не $1/p$, а гораздо большему числу $2/k$. Оказывается, что эти два условия уравновешивают друг друга: ограниченность множества возможных простых делителей компенсируется возрастающей вероятностью делимости на них, что приводит к тому же эвристическому результату, что и выше. Это более сложное рассуждение было приведено в первом издании нашей книги.

## 1.3.2. Числа Ферма

Знаменитые числа Ферма $F_n = 2^{2^n} + 1$, как и числа Мерсенна, привлекают внимание исследователей уже несколько веков. В 1637 г. Ферма объявил, что все числа $F_n$ являются простыми. И в самом деле, первые пять чисел Ферма, от $F_0 = 3$ до $F_4 = 65537$, просты. Однако можно сказать, что здесь мы имеем дело с тем редким случаем, когда Ферма был не прав, возможно почти *совсем* не прав. Дело в том, что все остальные числа $F_n$, которые удалось проверить на простоту, оказались составными! Первое из таких чисел, $F_5$, было разложено на множители Эйлером.

Замечательный результат, связанный с простыми числами Ферма, был получен Гауссом (и снова в юном возрасте). Гаусс доказал, что правильный $n$-угольник можно построить циркулем и линейкой тогда и только тогда, когда наибольший нечетный делитель числа $n$ является произведением различных простых чисел Ферма. Если числа $F_0, \ldots, F_4$ окажутся *единственными* простыми числами Ферма, то это будет означать, что при помощи циркуля и линейки можно построить лишь те $n$-угольники, для которых $n = 2^a m$, где $m \mid 2^{32} - 1$ (поскольку произведение известных простых чисел Ферма равно $2^{32} - 1$). Сейчас мы покажем, что при поиске простых чисел, на 1 превосходящих степень 2, достаточно ограничиться простыми числами Ферма.

**Теорема 1.3.4.** *Если нечетное число $p = 2^m + 1$ простое, то $m$ есть степень числа 2.*

Доказательство. Предположим, что $m = ab$, где через $a$ обозначен наибольший нечетный делитель числа $m$. Тогда число $p$ делится на $2^b + 1$. Следовательно, для того, чтобы число $p$ было простым, необходимо, чтобы выполнялось равенство $p = 2^b + 1$, т. е. $a = 1$ и $m = b$ есть степень числа 2. □

Как и в случае простых чисел Мерсенна, существует полезный результат, содержащий ограничение на простые делители чисел Ферма.

**Теорема 1.3.5** (Эйлер, Люка). *При $n \geqslant 2$ каждый простой делитель $p$ числа $F_n = 2^{2^n} + 1$ удовлетворяет сравнению $p \equiv 1 \pmod{2^{n+2}}$.*

ДОКАЗАТЕЛЬСТВО. Пусть $r$ — простой делитель числа $F_n$ и пусть число $h$ — наименьший натуральный корень сравнения $2^h \equiv 1 \pmod{r}$. Тогда, поскольку $2^{2^n} \equiv -1 \pmod{r}$, мы получаем $h = 2^{n+1}$. Значит, как мы видели при доказательстве теоремы 1.3.1, $2^{n+1}$ делит $r - 1$. Учитывая ограничение $n \geqslant 2$, мы заключаем, что $r \equiv 1 \pmod{8}$. Принимая во внимание (2.10), отсюда получаем, что 2 есть квадратичный вычет по модулю $r$, а следовательно, $h = 2^{n+1}$ делит $\frac{r-1}{2}$, откуда и следует утверждение теоремы. □

Именно этот результат помог Эйлеру найти делитель числа $F_5$, и стать первым, кто внес вклад в опровержение злополучной гипотезы Ферма. (Версия Эйлера теоремы 1.3.5 содержала более слабое утверждение, чем $p \equiv 1 \pmod{2^{n+1}}$, но его хватило для нахождения делителя 641 числа $F_5$.) И сегодня теорема 1.3.5 применяется в процессе поиска делителей огромных чисел Ферма.

Как и для чисел Мерсенна, для чисел Ферма существуют очень эффективные тесты, позволяющий проверить их простоту. Это и тест Пепена, и схожий с ним тест Суямы (для делителей чисел Ферма). По этому вопросу см. теорему 4.1.2 и упр. 4.5, 4.7, 4.8.

Сочетание различных методов, включая тесты Пепена и Суямы, а также в ряде случаев и более совершенные алгоритмы, позволило разложить на множители изрядное количество чисел Ферма, причем некоторые были разложены частично, но этого было вполне достаточно, чтобы продемонстрировать их составность. Последние данные обо всех числах $F_n$, $n \leqslant 24$, отражены в табл. 1.3.

Приведем обзор интересных с теоретической точки зрения фактов, связанных с табл. 1.3. (Отметим, что многие алгоритмы, успешно применявшиеся для исследования простоты чисел Ферма, описаны в гл. 5, 6 и 7.)

(1) $F_7$ было разложено на множители методом цепных дробей (см. [Morrison and Brillhart 1975]), а для $F_8$ была применена модификация ро-метода Полларда (см. [Brent and Pollard 1981]).

(2) Эффектный 49-значный делитель числа $F_9$ был найден при помощи метода решета числового поля (NFS) (см. [Lenstra et al. 1993a]).

(3) Благодаря методу эллиптических кривых, позволившему разложить на множители числа $F_{10}$ (см. [Brent 1999]) и $F_{11}$ (также результат Брента), наименьшим из не разложенных до конца на множители чисел Ферма является $F_{12}$.

(4) Два наибольших известных простых делителя числа $F_{13}$, а также наибольшие простые делители чисел $F_{15}$ и $F_{16}$, были недавно найдены посредством усовершенствованного метода эллиптических кривых (ECM) (см. [Crandall 1996a], [Brent et al. 2000]) (см. разд. 7.4.1). Этот же способ позволил в 1999 г. Макинтошу и Тардифу открыть и 23-значный делитель числа $F_{18}$.

$$
\begin{aligned}
F_0 &= 3 = P\\
F_1 &= 5 = P\\
F_2 &= 17 = P\\
F_3 &= 257 = P\\
F_4 &= 65537 = P\\
F_5 &= 641 \cdot 6700417\\
F_6 &= 274177 \cdot 67280421310721\\
F_7 &= 59649589127497217 \cdot 5704689200685129054721\\
F_8 &= 1238926361552897 \cdot P\\
F_9 &= 2424833 \cdot 7455602825647884208337395736200454918783366342657 \cdot P\\
F_{10} &= 45592577 \cdot 6487031809 \cdot 4659775785220018543264560743076778192897 \cdot P\\
F_{11} &= 319489 \cdot 974849 \cdot 167988556341760475137 \cdot 3560841906445833920513 \cdot P\\
F_{12} &= 114689 \cdot 26017793 \cdot 63766529 \cdot 190274191361 \cdot 1256132134125569 \cdot C\\
F_{13} &= 2710954639361 \cdot 2663848877152141313 \cdot 3603109844542291969 \cdot\\
&\qquad 319546020820551643220672513 \cdot C\\
F_{14} &= C\\
F_{15} &= 1214251009 \cdot 2327042503868417 \cdot 1687688170295169723830241270169 61 \cdot C\\
F_{16} &= 825753601 \cdot 188981757975021318420037633 \cdot C\\
F_{17} &= 31065037602817 \cdot C\\
F_{18} &= 13631489 \cdot 81274690703860512587777 \cdot C\\
F_{19} &= 70525124609 \cdot 646730219521 \cdot C\\
F_{20} &= C\\
F_{21} &= 4485296422913 \cdot C\\
F_{22} &= C\\
F_{23} &= 167772161 \cdot C\\
F_{24} &= C
\end{aligned}
$$

**Таблица 1.3.** Все, что известно о первых 25 числах Ферма (на апрель 2005 г.). Обозначения: $P$ — простое число, $C$ — составное число. Все явно выписанные множители — простые числа. Следующим по величине потенциальным простым числом Ферма является число $F_{33}$.

(5) Числа $F_{14}, F_{20}, F_{22}, F_{24}$ (и другие числа таблицы, скрывающиеся за символом $C$) на момент написания книги являются «чисто» составными, т. е. мы знаем, что они *не* простые, но ни один их простой делитель нам не известен (см. упр. 1.82, в котором описаны трудности, связанные с так определяемым понятием «чисто» составного числа).

(6) Тест Пепена позволил установить составность чисел $F_{14}$ (см. [Selfridge and Hurwitz 1964]) и $F_{20}$ (см. [Buell and Young 1988]).

(7) При доказательстве составности числа $F_{22}$ (см. [Crandall et al. 1995]) возникла интересная ситуация: совершенно независимая (в смысле программного обеспечения, вычислительной техники и расположения) группа исследователей в Южной Африке (см. [Trevisan and Carvalho 1993]), реализовав тест Пепена, получила для $F_{22}$ тот же результат. На самом деле, эти исследователи нашли те же вычеты Селфриджа—Гурвица, т. е.

значения наименьшего неотрицательного вычета по модулю $F_n$, затем снова взятого по модулю тех же трех взаимно простых модулей $2^{36}$, $2^{36}-1$ и $2^{35}-1$ для контроля по четности с ошибкой, примерно равной $2^{-107}$. Несмотря на угрозу машинной ошибки во время такого продолжительного разового прогона, совпадение полученных независимо результатов не оставляет сомнений в составности числа $F_{22}$.

(8) Составность чисел $F_{24}$ и делителя числа $F_{23}$ была установлена Крэндаллом, Майером и Пападопулосом (см. [Crandall et al. 2003]) в 1999–2000 гг. В этом случае для достоверности результата были реализованы два независимых теста Пепена волновой обработки данных с плавающей точкой (Майер и Пападопулос, работа закончена в августе 1999 г.), а также исключительно целочисленный метод свертки для детерминистической проверки цепи Пепена возведения в квадрат. Поэтому и в этом случае вероятность ошибки при расчетах ничтожно мала. Продолжение этой темы см. в упр. 4.6.

(9) После $F_{24}$ все числа $F_n$ вплоть до $n=32$ имеют как минимум один найденный собственный делитель, причем все эти делители найдены методом пробных делений при помощи теоремы 1.3.5 (в частности, Круппа и Форбс в 2001 г. нашли делитель 46931635677864055013377 числа $F_{31}$). Поэтому следующим числом Ферма неизвестного характера является $F_{33}$. Но *стандартное* оборудование и тест Пепена потребуют тысяч лет для исследования числа $F_{33}$! Поэтому для работы с такими огромными числами Ферма остро необходимы новые алгоритмы.

С числами Ферма связано множество разнообразных любопытных вопросов. Например, поставлена задача отыскания наибольшего составного числа $F_n$. Келлер показал, что $F_{23471}$ делится на $5\cdot 2^{23473}+1$, позднее Янг открыл, что $F_{213319}$ кратно $3\cdot 2^{213321}+1$ (см. [Keller 1999]), а совсем недавно Косгрейв, используя замечательное программное обеспечение Гало, показал, что $F_{382447}$ делится на $3\cdot 2^{382449}+1$ (см. упр. 4.9). Для того чтобы продемонстрировать, насколько тяжелыми были эти исследования, отметим, что найденный Косгрейвом простой делитель *сам* входит примерно в десятку наибольших известных на сегодняшний день простых чисел. Аналогичные исследования были недавно описаны в работе [Dubner and Gallot 2002]. В частности, там приведены найденное Херраненом обобщенное простое число Ферма

$$101830^{2^{14}}+1$$

и огромное простое число Скотта

$$48594^{2^{16}}+1.$$

Собрание численных результатов в области чисел Ферма можно найти на сайте [Keller 1999].

Интересно, что числа Ферма позволяют получить еще одно доказательство теоремы 1.1.2 о бесконечности множества простых чисел. Поскольку все чис-

ла Ферма нечетны, а произведение чисел $F_0, F_1, \ldots, F_{n-1}$ равно $F_n - 2$, мы немедленно заключаем, что ни один из простых делителей числа $F_n$ не делит ни одно из предшествующих $F_j$, а следовательно, множество простых чисел бесконечно.

А как насчет эвристических соображений? Можем ли мы высказать гипотезу о виде асимптотической формулы для количества чисел $n \leqslant x$, для которых число $F_n$ простое? Если провести рассуждения, аналогичные тем, что мы провели для простых чисел Мерсенна, то можно заключить, что существует лишь конечное множество простых чисел Ферма. Это следует из сходимости ряда с общим членом $n/2^n$, сумма которого пропорциональна предполагаемой вероятности простоты числа $F_n$. Если относиться к подобным выкладкам серьезно, то можно сказать, что последним простым числом Ферма является $F_4$, на котором и остановился Ферма, с уверенностью предсказав, что все бóльшие числа Ферма просты! Эвристические рассуждения Ленстры мл., близкие по духу полученной нами выше оценке плотности простых чисел Мерсенна, приводят к тому, что вероятность получить простое число $F_n$ примерно равна

$$\frac{e^{\gamma} \lg b}{2^n}, \tag{1.13}$$

где через $b$ обозначен *текущий* предел просеивания для возможных простых делителей числа $F_n$. Если информация о возможных делителях отсутствует, для числителя можно воспользоваться наименьшей возможной нижней оценкой $b = 3 \cdot 2^{n+2} + 1$, получив грубую априорную вероятность $n/2^n$ простоты числа $F_n$. (Между прочим, схожее вероятностное рассуждение для обобщенных чисел Ферма вида $b^{2^n} + 1$ можно найти в работе [Dubner and Gallot 2002].) Таким образом, можно сказать, что вероятность открытия нового простого числа Ферма равна нулю.

### 1.3.3. Некоторые предположительно редкие простые числа

Традиционно выделяют классы предположительно редких простых чисел. Мы говорим «предположительно», поскольку, хотя строгие оценки плотности распределения таких чисел обычно слишком слабы, численные свидетельства и эвристические рассуждения подсказывают их относительную редкость. Пусть $p$ — произвольное нечетное простое число. Тогда согласно малой теореме Ферма $2^{p-1} \equiv 1 \pmod{p}$. Можно поставить вопрос о существовании простых чисел, удовлетворяющих сравнению

$$2^{p-1} \equiv 1 \pmod{p^2}. \tag{1.14}$$

Такие числа называют простыми Вифериха. Этот специальный вид простых чисел имеет непосредственное отношение к так называемому первому случаю последней теоремы Ферма. А именно, Виферих (см. [Wieferich 1909]) доказал, что если выполняется равенство

$$x^p + y^p = z^p,$$

где простое число $p$ не делит произведение $xyz$, то число $p$ удовлетворяет сравнению (1.14). Эквивалентным образом можно сказать, что $p$ — простое число Вифериха, если частное Ферма

$$q_p(2) = \frac{2^{p-1} - 1}{p}$$

кратно $p$. Можно предположить, что вероятность выполнения последнего условия для случайного числа $p$ примерно равна $1/p$, а поскольку ряд чисел, обратных к простым, расходится (см. упр. 1.20), мы можем предположить существование бесконечного множества простых Вифериха. Кроме того, из-за очень медленной расходимости упомянутого ряда можно сделать вывод о том, что эти простые числа встречаются довольно редко.

Простые Вифериха 1093 и 3511 известны давно. Крэндалл, Дильхер и Померанс при вычислительном содействии Бейли установили, что других простых Вифериха до $4 \cdot 10^{12}$ нет (см. [Crandall et al. 1997]). Позднее Макинтош отодвинул эту границу до $16 \cdot 10^{12}$. До сих пор неизвестно, существует ли хотя бы одно простое Вифериха, большее 3511. Неизвестно даже, существует ли бесконечное множество простых чисел, *не* являющихся простыми Вифериха! (Тем не менее, см. упр. 8.19.)

Перейдем к рассмотрению следующего предположительно редкого класса. Но сначала сформулируем один классический критерий.

**Теорема 1.3.6** (Вильсон—Лагранж). *Для того чтобы целое число $p \geqslant 2$ было простым, необходимо и достаточно, чтобы выполнялось сравнение*

$$(p-1)! \equiv -1 \pmod{p}.$$

Простые числа, удовлетворяющие сравнению

$$(p-1)! \equiv -1 \pmod{p^2}, \tag{1.15}$$

называются простыми числами Вильсона. Каждому простому числу $p$ можно поставить в соответствие частное Вильсона

$$w_p = \frac{(p-1)! + 1}{p},$$

кратность которого числу $p$ является критерием для простых чисел Вильсона. Снова можно считать, что вероятность того события, что $p$ — простое число Вильсона, примерно равна $1/p$, и опять редкость этих чисел показана эмпирически: 5, 13 и 563 являются единственными числами Вильсона, меньшими $5 \cdot 10^8$.

Третий класс предположительно редких простых чисел образуют простые числа Уолла—Сана—Сана, определяемые сравнением

$$u_{p-\left(\frac{p}{5}\right)} \equiv 0 \pmod{p^2}, \tag{1.16}$$

где через $u_n$ обозначено $n$-е число Фибоначчи (определение см. в упр. 2.5) и

$$\left(\frac{p}{5}\right) = \begin{cases} 1, & \text{если } p \equiv \pm 1 \pmod 5, \\ -1, & \text{если } p \equiv \pm 2 \pmod 5, \\ 0, & \text{если } p = 5. \end{cases}$$

Как и в случае простых чисел Вифериха и Вильсона, сравнение (1.16) всегда имеет место (mod $p$). Макинтош показал, что нет ни одного простого числа Уолла—Сана—Сана, меньшего $3.2 \cdot 10^{12}$. Простые числа Уолла—Сана—Сана также имеют отношение к первому случаю последней теоремы Ферма: простой показатель $p$ из уравнения $x^p + y^p = z^p$, где $p$ не делит произведение $xyz$, должен также удовлетворять сравнению (1.16) (см. [Sun and Sun 1992]).

Поиск простых чисел Вифериха, Вильсона и Уолла—Сана—Сана ставит интересные вычислительные задачи, многие из которых описаны в упражнениях, а сейчас мы коснемся лишь нескольких ярких примеров. Во-первых, вычисления (mod $p^2$) удобно проводить, представляя каждый класс вычетов в виде пары $(a, b) = a + bp$. Таким образом, для умножения можно ввести оператор $*$, действующий по формуле

$$(a, b) * (c, d) \equiv (ac, (bc + ad) \pmod p) \pmod{p^2},$$

которая позволяет производить все арифметические действия, необходимые для отыскания редких простых чисел, описанных в настоящем разделе, работая с числами, не превосходящими самого числа $p$. Во-вторых, факториалы могут быть вычислены при помощи различных усовершенствований, которые применяются для вычисления произведения членов арифметической прогрессии и вычисления значений многочлена (см. гл. 9). Например, с помощью полиномиального вычисления соответствующего факториала показано (см. [Crandall et al. 1997]), что при $p = 2^{40} + 5$

$$(p-1)! \equiv -1 - 533091778023p \pmod{p^2}.$$

Следовательно, это число $p$ не является простым числом Вильсона, но, тем не менее, важно отметить, что в наши дни техника может проверять простоту даже 12-значных чисел, используя обращение Лагранжа классической теоремы Вильсона.

При поиске описанных редких простых чисел иногда встречаются числа, про которые можно сказать, что они «почти то, что нам нужно». Видимо, роль таких чисел заключается в поддержке наших эвристических представлений о статистике, например, частных Ферма и Вильсона. Примерами малых (но, к сожалению, ненулевых) значений этих отношений являются следующие:

$$\begin{aligned} p &= 76843523891, & q_p(2) &\equiv -2 \pmod p, \\ p &= 12456646902457, & q_p(2) &\equiv 4 \pmod p, \\ p &= 56151923, & w_p &\equiv -1 \pmod p, \\ p &= 93559087, & w_p &\equiv -3 \pmod p. \end{aligned}$$

(Как мы помним, если частное Ферма или Вильсона равно нулю по модулю $p$, то мы имеем дело с соответствующим простым числом.)

## 1.4. Аналитическая теория чисел

Аналитическая теория чисел привлекает методы анализа для изучения дискретного множества целых чисел. При доказательстве теорем применяется выход в комплексную область, метод контурного интегрирования и другие приемы. Эти мощные и привлекательные инструменты полезны и для исследования алгоритмов, а также сами по себе являются источником ряда интересных алгоритмических проблем. В этом разделе мы коснемся нескольких ключевых результатов аналитической теории чисел.

### 1.4.1. Дзета-функция Римана

Функцию

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s} \tag{1.17}$$

применял в своих исследованиях еще Л. Эйлер, но именно Б. Риман в середине XIX в. начал рассматривать ее как функцию комплексной переменной, что положило начало серии замечательных открытий. Этот ряд сходится абсолютно при $\mathrm{Re}(s) > 1$ и допускает аналитическое продолжение на всю комплексную плоскость, регулярное всюду, за исключением точки $s = 1$, в которой дзета-функция имеет простой полюс с вычетом 1. (Это означает, что функция $(s-1)\zeta(s)$ аналитична на всей комплексной плоскости и в точке $s = 1$ равна 1.) Достаточно легко можно продолжить функцию $\zeta(s)$ из области $\mathrm{Re}(s) > 1$ на полуплоскость $\mathrm{Re}(s) > 0$: при $\mathrm{Re}(s) > 1$ справедливо равенство

$$\zeta(s) = \frac{s}{s-1} - s \int_1^{\infty} (x - \lfloor x \rfloor) x^{-s-1}\, dx,$$

причем интеграл в правой части сходится и при $\mathrm{Re}(s) > 0$, $s \neq 1$, поэтому мы можем рассматривать это интегральное представление как определение дзета-функции в расширенной области. Из рассмотренного равенства также ясно виден полюс в точке $s = 1$, а также, например, факт отсутствия нулей функции $\zeta$ на положительной действительной полупрямой. Существуют другие аналитические выражения, позволяющие продолжить дзета-функцию на *всю* комплексную плоскость.

Связь свойств дзета-функции и простых чисел была подмечена еще Эйлером, считавшим аргумент $s$ действительным числом. Следующее прекрасное соотношение можно рассматривать как аналитический аналог основной теоремы арифметики 1.1.1.

**Теорема 1.4.1** (Эйлер). *Пусть $\mathcal{P}$ обозначает множество всех простых чисел. Тогда в области $\mathrm{Re}(s) > 1$ справедливо равенство*

$$\zeta(s) = \prod_{p \in \mathcal{P}} (1 - p^{-s})^{-1}. \tag{1.18}$$

Доказательство. Множитель Эйлера $(1 - p^{-s})^{-1}$ можно представить в виде геометрической прогрессии $1 + p^{-s} + p^{-2s} + \ldots$. Рассмотрим операцию одновре-

менного умножения всех таких прогрессий. Получившийся общий член будет равен $\prod_{p\in\mathcal{P}} p^{-a_p s}$, где каждое из целых чисел $a_p$ неотрицательно, причем лишь конечное множество чисел $a_p$ отлично от нуля. Следовательно, этот общий член имеет вид $n^{-s}$ для некоторого натурального числа $n$, причем согласно теореме 1.1.1, каждое из чисел $n$ встретится только один раз. Таким образом, правая часть равенства совпадает с левой, что и требовалось доказать. □

Эйлеру был также известен ряд численных значений дзета-функции, таких как

$$\begin{aligned}\zeta(2) &= \pi^2/6,\\ \zeta(4) &= \pi^4/90,\end{aligned}$$

и вообще, $\zeta(n)$ для всех четных значений $n$. Однако до сих пор не найдены аналогичные аналитические выражения для $\zeta(n)$ при нечетных значениях $n > 2$. Но основная информация о дзета-функции, позволяющая получать результаты, связанные с простыми числами, касается ее свойств при $\mathrm{Re}(s) \leqslant 1$. На этом множестве иногда удается установить такие факты, как, например, равенство

$$\zeta(0) = -1/2.$$

Вот лишь несколько классических результатов, связанных с теоретическими приложениями дзета-функции.

(1) Из того факта, что $\zeta(s) \to \infty$ при $s \to 1$, вытекает бесконечность множества простых чисел.
(2) Отсутствие нулей функции $\zeta(s)$ на прямой $\mathrm{Re}(s) = 1$ влечет теорему о простых числах 1.1.4.
(3) Установлена взаимосвязь между свойства дзета-функции в критической полосе $0 < \mathrm{Re}(s) < 1$ и многими глубокими проблемами распределения простых чисел, такими, как уточнение остаточного члена в асимптотической формуле для функции $\pi(x)$.

Проведем рассуждения, соответствующие пункту (1).

Еще одно доказательство бесконечности множества простых чисел. Рассмотрим $\zeta(s)$ как функцию действительного аргумента $s > 1$. Очевидно, что из равенства (1.17) следует, что $\zeta(s) \to +\infty$ при $s \to 1^+$, поскольку гармонический ряд $\sum 1/n$ расходится. Однако проведем строгие рассуждения. При $s > 1$ имеем

$$\begin{aligned}\zeta(s) &> \sum_{n\leqslant 1/(s-1)} n^{-s} = \sum_{n\leqslant 1/(s-1)} n^{-1}n^{-(s-1)}\\ &\geqslant e^{-1/e} \sum_{n\leqslant 1/(s-1)} n^{-1} > e^{-1/e}|\ln(s-1)|,\end{aligned}$$

а $|\ln(s-1)| \to +\infty$ при $s \to 1^+$.

Итак, если бы существовало лишь конечное множество простых чисел, то произведение в формуле (1.18) имело бы конечный предел при $s \to 1^+$. Полученное противоречие завершает доказательство. □

Отметим, что из приведенного доказательства вытекает расходимость ряда, составленного из всех чисел, обратных к простым. Действительно,

$$\ln\left(\prod_{p\in\mathcal{P}}(1-p^{-s})^{-1}\right) = -\sum_{p\in\mathcal{P}}\ln(1-p^{-s}) = \sum_{p\in\mathcal{P}}p^{-s}+O(1) \tag{1.19}$$

равномерно по всем $s>1$, а поскольку левая часть соотношения (1.19) стремится к $\infty$ при $s\to 1^+$ и $p^{-s}<p^{-1}$ при $s>1$, мы заключаем, что ряд $\sum_{p\in\mathcal{P}}p^{-1}$ расходится (ср. упр. 1.20). Именно таким способом Дирихле и доказал теорему 1.1.5 (см. разд. 1.4.3).

На самом деле о поведении частичных сумм ряда $\sum_p 1/p$ можно утверждать нечто большее (здесь и далее, если не оговорено противное, мы всегда будем подразумевать условие $p\in\mathcal{P}$).

**Теорема 1.4.2** (Мертенс). *При $x\to\infty$ справедливо соотношение*

$$\prod_{p\leqslant x}\left(1-\frac{1}{p}\right)\sim\frac{e^{-\gamma}}{\ln x}, \tag{1.20}$$

*где $\gamma$ обозначает постоянную Эйлера. В частности,*

$$\sum_{p\leqslant x}\frac{1}{p} = \ln\ln x + B + o(1), \tag{1.21}$$

*где постоянная Мертенса $B$ определена равенством*

$$B=\gamma+\sum_p\left(\ln\left(1-\frac{1}{p}\right)+\frac{1}{p}\right).$$

Формула (1.21) получается логарифмированием соотношения (1.20), а доказательство можно найти, например, в книге[4] [Hardy and Wright 1979]. Эту теорему можно получить как следствие асимптотического закона 1.1.4, но теорема Мертенса проще и была доказана раньше. С другой стороны, асимптотический закон позволяет получить более точные оценки остаточных членов в формулах (1.20) и (1.21). Кстати говоря, вычисление постоянной Мертенса $B$ также представляет интерес (см. упр. 1.90).

Как мы видели, существует глубокая связь между распределением простых чисел и свойствами дзета-функции Римана. В частности, поведение функции $\zeta(s)$ в критической полосе, т. е. в области $0<\operatorname{Re}(s)<1$, несет подробную информацию о частоте появления простых чисел. Например, существует явное выражение функции $\pi(x)$ через нули функции $\zeta(s)$ в критической полосе. Продемонстрируем аналогичную связь на примере функции $\psi_0(x)$, связанной с функцией $\pi(x)$, но являющейся более удобной для исследования аналитическими методами. Зададим функцию $\psi(x)$ равенством

$$\psi(x)=\sum_{p^m\leqslant x}\ln p=\sum_{p\leqslant x}\ln p\left\lfloor\frac{\ln x}{\ln p}\right\rfloor \tag{1.22}$$

[4] Русскоязычному читателю можно рекомендовать книгу [Прахар 1967]. — *Прим. ред.*

и положим

$$\psi_0(x) = \begin{cases} \psi(x) - \frac{1}{2}\ln p, & \text{если } x = p^m, \\ \psi(x) & \text{в противном случае.} \end{cases}$$

Тогда (см. [Edwards 1974], [Davenport 1980][5], [Ivić 1985]) при $x > 1$ справедливо равенство

$$\psi_0(x) = x - \sum_\rho \frac{x^\rho}{\rho} - \ln(2\pi) - \frac{1}{2}\ln\left(1 - x^{-2}\right), \tag{1.23}$$

где суммирование распространено на все нули $\rho$ функции $\zeta(s)$, для которых $\mathrm{Re}(\rho) > 0$. Поскольку этот ряд не является абсолютно сходящимся, а в критической полосе по обе стороны от действительной оси лежит бесконечное множество нулей $\rho$, мы будет понимать сумму ряда как предел при $T \to +\infty$ конечных сумм по всем нулям $\rho$, удовлетворяющим неравенству $|\rho| < T$. Кроме того, мы считаем, что каждый нуль $\rho$ функции $\zeta(s)$ подсчитывается с учетом его кратности. (Впрочем, имеется гипотеза о том, что все нули дзета-функции Римана простые.)

Риман выдвинул гипотезу, занявшую ключевое место в теории чисел, а быть может, и во всей математике.

**Гипотеза 1.4.1** (гипотеза Римана (RH)). *Все нули функции $\zeta(s)$ в критической полосе $0 < \mathrm{Re}(s) < 1$ лежат на прямой $\mathrm{Re}(s) = 1/2$.*

Гипотеза Римана допускает множество эквивалентных формулировок, об одной из которых мы упоминали в разд. 1.1.5. Рассмотрим теперь функцию Мертенса

$$M(x) = \sum_{n \leqslant x} \mu(n),$$

где через $\mu(n)$ обозначена функция Мёбиуса, равная 1, если бесквадратное число $n$ имеет четное количество простых делителей, $-1$, если нечетное, и 0, если число $n$ не является бесквадратным. (Например, $\mu(1) = \mu(6) = 1$, $\mu(2) = \mu(105) = -1$, $\mu(9) = \mu(50) = 0$.) Связь между функциями $M(x)$ и $\zeta(s)$ видна из соотношения

$$\frac{1}{\zeta(s)} = \sum_{n=1}^{\infty} \frac{\mu(n)}{n^s} = s\int_1^\infty \frac{M(x)}{x^{s+1}}\,dx, \tag{1.24}$$

справедливого при $\mathrm{Re}(s) > 1$. Зависимость распределения простых чисел от поведения функции Мертенса видна из следующего утверждения (здесь и далее в этом разделе постоянная в символе $O$ может зависеть от $\varepsilon$).

**Теорема 1.4.3.** *Асимптотический закон распределения простых чисел эквивалентен равенству*

$$M(x) = o(x),$$

*а гипотеза Римана равносильна тому утверждению, что для любого фиксированного $\varepsilon > 0$ верно равенство*

$$M(x) = O\left(x^{\frac{1}{2}+\varepsilon}\right).$$

[5] Имеется перевод на русский язык. См. [Давенпорт 1971]. — *Прим. ред.*

Удивительно, что функция Мертенса, которую можно рассматривать как случайное блуждание, поскольку каждое слагаемое $\mu(n)$ может изменить сумму для $M$ только на единицу (здесь прослеживается аналогия с подбрасыванием монеты), оказывается настолько сильно связанной с такими великими утверждениями!

Список эквивалентных утверждений из теоремы 1.4.3 можно пополнить множеством других любопытных фактов. Одним из таких элегантных результатов является равносильность теоремы о простых числах 1.1.4 равенству

$$\sum_{n=1}^{\infty} \frac{\mu(n)}{n} = 0,$$

что было установлено Мангольдтом. Отметим, что доказать абсолютную сходимость при $\operatorname{Re}(s) > 1$ ряда в соотношении (1.24) не так трудно, как строго рассчитать значение этой суммы в точке $s = 1$ (см. упр. 1.19). В 1859 г. Риман высказал предположение о том, что при каждом фиксированном $\varepsilon > 0$ справедливо равенство

$$\pi(x) = \operatorname{li}(x) + O\left(x^{1/2+\varepsilon}\right). \tag{1.25}$$

Это утверждение эквивалентно гипотезе Римана, а также, разумеется, и второму соотношению теоремы 1.4.3. Более того, равенство (1.25) равносильно утверждению об отсутствии нулей функции $\zeta(s)$ в области $\operatorname{Re}(s) > 1/2 + \varepsilon$. При $\varepsilon < 1/2$ формула (1.25) ждет своего доказательства.

В 1901 г. фон Кох несколько усилил равенство (1.25), показав, что гипотеза Римана равносильна соотношению $|\pi(x) - \operatorname{li}(x)| = O(\sqrt{x} \ln x)$. На самом деле, при $x \geqslant 2.01$ в качестве постоянной в знаке $O$ этого равенства можно взять 1 (см. упр. 1.37).

Обозначим $n$-е простое число через $p_n$. Согласно соотношению (1.25), если имеет место гипотеза Римана, то

$$p_{n+1} - p_n = O\left(p_n^{1/2+\varepsilon}\right)$$

для любого фиксированного $\varepsilon > 0$. На сегодняшний день лучшим доказанным результатом в этом направлении является оценка $p_{n+1} - p_n = O\left(p_n^{0.525}\right)$, полученная Бейкером, Харманом и Пинтцем. Однако существует гораздо более сильная знаменитая гипотеза Крамера:

$$\limsup_{n\to\infty} (p_{n+1} - p_n)/\ln^2 n = 1.$$

Грэнвилль высказал некоторые сомнения по поводу значения этого предела, предложив для него значение $2e^{-\gamma} \approx 1.123$. Для простых чисел, больших 100, наибольшее из известных значений выражения $(p_{n+1} - p_n)/\ln^2 n$ примерно равно 1.210 (для $p_n = 113$). Следующие по величине значения этого отношения равны $\approx 1.175$ (для $p_n = 1327$) и $\approx 1.138$ (для $p_n = 1693182318746371$). Последний результат принадлежит Найману.

Согласно асимптотическому закону (теорема 1.1.4), расстояния между соседними простыми числами $p_{n+1} - p_n$, так ярко подчеркивающие явную ло-

кальную случайность распределения простых чисел, в среднем эквивалентны $\ln n$. Гипотеза Крамера—Грэнвилля, упомянутая в абзаце выше, утверждает, что эти расстояния бесконечно часто имеют порядок $\ln^2 n$ и никакой больший. Однако лучшее из доказанного на текущий момент состоит в том, что разность $p_{n+1} - p_n$ бесконечно часто есть величина порядка, не меньшего

$$\ln n \ln\ln n \ln\ln\ln\ln n/(\ln\ln\ln n)^2$$

(старый результат Эрдёша и Ранкина). Можно поставить вопрос и о минимальном порядке разности $p_{n+1} - p_n$. Из гипотезы о простых близнецах вытекает, что $(p_{n+1} - p_n)/\ln n$ имеет нижний предел, равный 0, но вплоть до недавнего времени лучшим из известных был результат Майера, согласно которому он не превосходит постоянной, которая немного меньше, чем $1/4$. Однако во время подготовки к печати настоящего второго издания Голдстон, Пинтц и Йилдирим[6] объявили, что ими получен потрясающий результат: нижний предел отношения $(p_{n+1} - p_n)/\ln n$ действительно равен 0.

## 1.4.2. Вычислительные достижения

На сегодняшний день гипотеза Римана (RH) остается не доказанной. Однако благодаря существенному развитию вычислительной техники за последние десятилетия, которое позволило провести столь значительные вычисления, мы знаем, что первые полтора миллиарда нулей дзета-функции в критической полосе (упорядоченных по возрастанию модуля мнимой части) лежат в точности на критической прямой $\mathrm{Re}(s) = 1/2$ (см. [van de Lune et al. 1986]). Замечательным обстоятельством является то, что благодаря некоторой симметрии, присущей дзета-функции, можно получать прекрасные численные значения ее нулей при помощи вычислений ограниченной (но, возможно, высокой) точности. Это достигается путем подсчета числа нулей до различных значений «высоты» $T$ (т. е. числа нулей вида $\sigma + it$, мнимая часть которых принадлежит промежутку $(0, T]$) и количества перемен знака некоторой действительной функции, принимающей нулевое значение тогда и только тогда, когда дзета-функция имеет соответствующий нуль на критической прямой. Если число перемен знака поддается учету, то при помощи вычислений можно найти все нули функции $\zeta(s)$ до высоты $T$ (см. [Brent 1979]).

Вычисления, связанные с нахождением нулей дзета-функции Римана в критической полосе, отражены в недавней статье [Gourdon and Sebah 2004]. В частности, первые $10^{13}$ нулей подтверждают гипотезу Римана. Кроме того, Гурдон вычислил 2 миллиарда нулей в окрестности $t = 10^{24}$. Эти продвинутые исследования основаны на варианте метода параллельного вычисления значений дзета-функции [Odlyzko and Schönhage 1988] (см. разд. 3.7.2). Еще одним лидером в области проверки гипотезы Римана является Веденивски, который является координатором проекта распределенных вычислений «zetagrid» [Wedeniwski 2004].

[6] См. [Goldston et al. 2005]. — *Прим. ред.*

Вычисления могут не только подчеркивать справедливость гипотез, но иногда и опровергать их. Например, знаменитая гипотеза Мертенса утверждала, что

$$|M(x)| < \sqrt{x}. \tag{1.26}$$

К сожалению, как оказалось, это предположение места не имеет. Более ранняя гипотеза, согласно которой в правой части стояло выражение $\frac{1}{2}\sqrt{x}$, было впервые опровергнуто в 1963 г. Нойбауэром. Позднее Коэн нашел наименьшее значение $x$, для которого нарушалось это неравенство:

$$M(7725038629) = 43947.$$

Наконец, сама гипотеза Мертенса (1.26) была опровергнута результатами (см. [Odlyzko and te Riele 1985])

$$\begin{aligned} \limsup_{x\to+\infty} x^{-1/2}M(x) &> 1.06, \\ \liminf_{x\to+\infty} x^{-1/2}M(x) &< -1.009. \end{aligned}$$

Как показал Пинтц, существует такое число $x < 10^{10^{65}}$, что отношение $M(x)/\sqrt{x}$ превосходит 1 (см. [Ribenboim 1996]). Отметим, что согласно математической статистике, сумматорная функция $m(x) = \sum_{n\leqslant x} t_n$ случайного блуждания (если величины $t_n = \pm 1$ случайны и независимы) с вероятностью 1 удовлетворяет соотношению

$$\limsup_{x\to+\infty} \frac{m(x)}{\sqrt{(x/2)\ln\ln x}} = 1.$$

Таким образом, если считать значения функции Мёбиуса $\mu$ достаточно случайными, то можно ожидать неограниченность отношения $M(x)/\sqrt{x}$.

Еще одно вычислительное приложение дзета-функции Римана заключается в оценке значений функции $\pi(x)$ для больших значений переменной $x$. К этому вопросу мы вернемся в п. 3.7.2.

Аналитическая теория чисел изобилует символами $O$. Каждый такой результат порождает вопрос о константе, скрывающейся за этим обозначением, а также о границе, начиная с которой будет выполнено соответствующее неравенство. Например, из точной формулировки теоремы о простых числах вытекает, что $n$-е простое число превосходит $n\ln n$ при достаточно больших значениях $n$. Легко видеть, что это выполнено и при малых значениях $n$. Так всегда ли это верно? Для того чтобы ответить на этот вопрос, требуется пройти по всему аналитическому доказательству, «расшифровывая» каждый встречающийся знак $O$, и, таким образом, раскрыть численный смысл словосочетания «достаточно большое число». Проведя подобные рассуждения, Россер (см. [Rosser 1939]) смог показать, что и в самом деле $n$-е простое число всегда больше, чем $n\ln n$. Позднее, в совместной работе с Шёнфельдом, им был получен ряд других явных результатов, связанных с простыми числами. Такие исследования остаются интересным и чрезвычайно полезным вычислительным направлением аналитической теории чисел.

### 1.4.3. $L$-функции Дирихле

Характеры Дирихле позволяют «закрутить» дзета-функцию Римана. Начнем рассказ о том, что мы имеем в виду, с рассмотрения характеров Дирихле.

**Определение 1.4.4.** Пусть $D$ — натуральное число, и комплекснозначная функция целого аргумента $\chi$ обладает следующими свойствами.

(1) Для всех целых чисел $m, n$ справедливо равенство $\chi(mn) = \chi(m)\chi(n)$.
(2) Функция $\chi$ имеет период $D$.
(3) $\chi(n) = 0$ тогда и только тогда, когда $\text{НОД}(n, D) > 1$.

Тогда функция $\chi$ называется характером Дирихле по модулю $D$.

Например, если целое число $D > 1$ нечетно, то символ Якоби $\left(\frac{n}{D}\right)$ является характером Дирихле по модулю $D$ (см. определение 2.3.3). Это простое следствие определения: если $\chi$ — характер Дирихле (mod $D$) и $\text{НОД}(n, D) = 1$, то $\chi(n)^{\varphi(D)} = 1$, т.е. $\chi(n)$ является корнем из единицы. В самом деле, $\chi(n)^{\varphi(D)} = \chi\left(n^{\varphi(D)}\right) = \chi(1)$, причем последнее равенство следует из теоремы Эйлера (см. (2.2)), согласно которой, если $\text{НОД}(n, D) = 1$, то $n^{\varphi(D)} \equiv 1 \pmod{D}$. Но $\chi(1) = 1$, поскольку $\chi(1) = \chi(1)^2$ и $\chi(1) \neq 0$.

Если $\chi_1, \chi_2$ — характеры Дирихле по модулям $D_1$ и $D_2$ соответственно, то произведение $\chi_1\chi_2$ является характером Дирихле по модулю $\text{НОК}[D_1, D_2]$, причем мы по определению полагаем $(\chi_1\chi_2)(n) = \chi_1(n)\chi_2(n)$. Таким образом, множество характеров Дирихле по модулю $D$ замкнуто относительно операции умножения. Более того, они образуют мультипликативную группу, роль единицы в которой играет $\chi_0$ — главный характер по модулю $D$:

$$\chi_0(n) = \begin{cases} 1, & \text{если } \text{НОД}(n, D) = 1, \\ 0 & \text{в противном случае.} \end{cases}$$

Обратным элементом к характеру $\chi$ в рассматриваемой группе является комплексно сопряженный к нему характер $\overline{\chi}$.

Как и в случае натуральных чисел, характеры допускают в некотором смысле единственное разложение, а именно, если каноническое разложение числа $D$ имеет вид $p_1^{a_1} \dots p_k^{a_k}$, то характер $\chi \pmod{D}$ можно единственным образом представить в виде произведения $\chi_1 \dots \chi_k$, где $\chi_j$ — характер $\pmod{p_j^{a_j}}$.

В дополнение отметим, что характеры по модулю степеней простых чисел допускают несложное построение и наглядное описание. Пусть $q = p^a$ — степень нечетного числа (также рассматривается случай $q = 2$ и 4). Пусть, далее, $g$ — один из первообразных корней (mod $q$) (первообразным корнем по модулю $D$ называется порождающий элемент мультипликативной группы $\mathbf{Z}_D^*$ вычетов по модулю $D$, взаимно простых с числом $D$, который, как можно показать, существует тогда и только тогда, когда $D$ не имеет среди своих собственных делителей число 4, а также не делится одновременно на два нечетных простых числа). Тогда степени числа $g \pmod{q}$ пробегают все классы вычетов (mod $q$), взаимно простые с числом $q$. Итак, если мы выберем любой корень из 1 степени $\varphi(q)$, скажем, $\eta$, то мы получим однозначно определенный характер $\chi \pmod{q}$,

обладающий свойством $\chi(g) = \eta$. Следовательно, существует в точности $\varphi(q)$ различных характеров $\chi \pmod q$.

Случай $q = 2^a$, где $a > 2$, является лишь чуть более сложным, поскольку по таким модулям примитивных корней не существует. Однако порядок элемента $5 \pmod{2^a}$ при $a > 2$ равен $2^{a-2}$, а элемент $2^{a-1} - 1$, порядок которого равен 2, не входит в циклическую подгруппу, порожденную вычетом 5. Поэтому вычеты 5 и $2^{a-1} - 1$ являются образующими мультипликативной группы нечетных вычетов $\pmod{2^a}$. Следовательно, мы можем строить характеры $\pmod{2^a}$, выбирая корень из 1 степени $2^{a-2}$, скажем, $\eta$, и число $\varepsilon \in \{1, -1\}$, тем самым получая единственный характер $\chi \pmod{2^a}$, обладающий свойствами $\chi(5) = \eta$, $\chi(2^{a-1}-1) = \varepsilon$. И снова мы замечаем, что существует ровно $\varphi(q)$ характеров $\chi \pmod q$.

Таким образом, для любого числа $D$ существует в точности $\varphi(D)$ характеров $\pmod D$, причем приведенные выше рассуждения позволяют не только построить все характеры, но и показывают, что группа характеров $\pmod D$ изоморфна мультипликативной группе $\mathbf{Z}_D^*$ вычетов $\pmod D$, взаимно простых с числом $D$.

В заключение краткого обзора характеров Дирихле приведем два (двойственных) соотношения, выражающих некоторое подобие ортогональности:

$$\sum_{\chi \pmod D} \chi(n) = \begin{cases} \varphi(D), & \text{если } n \equiv 1 \pmod D, \\ 0, & \text{если } n \not\equiv 1 \pmod D, \end{cases} \tag{1.27}$$

$$\sum_{n=1}^{D} \chi(n) = \begin{cases} \varphi(D), & \text{если } \chi = \chi_0 \pmod D, \\ 0 & \text{в противном случае.} \end{cases} \tag{1.28}$$

Теперь мы можем перейти к главной теме этого раздела — $L$-функциям Дирихле. $L$-функцией Дирихле, соответствующей характеру Дирихле $\chi$ по модулю $D$, называется функция

$$L(s, \chi) = \sum_{n=1}^{\infty} \frac{\chi(n)}{n^s}.$$

Ряд в правой части равенства сходится в области $\operatorname{Re}(s) > 1$, а если $\chi$ не главный характер, то и в области $\operatorname{Re}(s) > 0$. Справедливо аналогичное (1.18) представление $L$-функции в виде бесконечного произведения

$$L(s, \chi) = \prod_p \left(1 - \frac{\chi(p)}{p^s}\right)^{-1} \tag{1.29}$$

Из этой формулы легко видеть, что при $\chi = \chi_0 \pmod D$ имеет место равенство $L(s, \chi_0) = \zeta(s) \prod_{p \mid D}(1 - p^{-s})$, т. е. свойства функции $L(s, \chi_0)$ аналогичны свойствам дзета-функции Римана.

Дирихле ввел в рассмотрение $L$-функции при доказательстве теоремы 1.1.5 о простых числах в арифметических прогрессиях (в классах вычетов). Идея Дирихле заключалась в логарифмировании равенства (1.29) (подобно (1.19)),

откуда получается соотношение

$$\ln(L(s,\chi)) = \sum_p \frac{\chi(p)}{p^s} + O(1), \tag{1.30}$$

равномерное на множестве $\mathrm{Re}(s) > 1$ для всех характеров Дирихле $\chi$. Следовательно, если число $a$ взаимно просто с модулем $D$, мы получаем

$$\begin{aligned} \sum_{\chi \pmod D} \overline{\chi}(a) \ln(L(s,\chi)) &= \sum_{\chi \pmod D} \sum_p \frac{\overline{\chi}(a)\chi(p)}{p^s} + O(\varphi(D)) \\ &= \varphi(D) \sum_{p \equiv a \pmod D} \frac{1}{p^s} + O(\varphi(D)), \end{aligned} \tag{1.31}$$

причем второе равенство следует из соотношения (1.27) и того факта, что $\overline{\chi}(a)\chi(p) = \chi(bp)$, где число $b$ таково, что $ba \equiv 1 \pmod D$. Таким образом, соотношение (1.31) замечательно тем, что в нем выделены все простые числа $p$, принадлежащие классу вычетов $a \pmod D$. Если нам удастся показать, что левая часть равенства (1.31) бесконечно возрастает при $s \to 1^+$, то мы тем самым докажем существование бесконечного множества простых чисел $p \equiv a \pmod D$ (поскольку ряд из обратных к ним величин расходится). Мы уже знаем, что слагаемое в левой части, отвечающее главному характеру $\chi_0$, стремится к бесконечности, но может случиться и так, что вклад от остальных членов нарушит это свойство для всего выражения. Следовательно, и это является ключевым моментом доказательства теоремы 1.1.5, нам остается показать, что если характер $\chi$ не является главным $\pmod D$, то $L(1,\chi) \neq 0$. Доказательство можно найти, например, в книге[7] [Davenport 1980].

Точно так же, как нули функции $\zeta(s)$ несут важную информацию о распределении всех простых чисел, нули $L$-функций Дирихле $L(s,\chi)$ содержат сведения о распределении простых чисел в классах вычетов. Кроме того, существует следующее расширение гипотезы Римана.

**Гипотеза 1.4.2** (расширенная гипотеза Римана (ERH)). *Пусть $\chi$ — произвольный характер Дирихле. Тогда все нули функции $L(s,\chi)$ в области* $\mathrm{Re}(s) > 0$ *лежат на прямой* $\mathrm{Re}(s) = \frac{1}{2}$.

Более того, существует и еще более общее предположение, так называемая обобщенная гипотеза Римана (GRH), относящаяся к более общим алгебраическим множествам. Тем не менее, мы ограничимся рассмотрением сформулированной расширенной гипотезы. (Отметим, что один из путей качественного понимания сути обобщенной гипотезы Римана заключается в том, что согласно этой гипотезе каждая дзетаподобная функция, которая «не должна» иметь нулей в определенной области, и в самом деле таких нулей не имеет (см. [Bach and Shallit 1996]).) Гипотеза 1.4.2 играет значительную роль и в вычислительной теории чисел. Например, имеет место такой условный результат.

**Теорема 1.4.5.** *Пусть справедлива ERH. Тогда для каждого натурального числа $D$ и каждого не главного характера $\chi \pmod D$ существует такое на-*

[7] Имеется перевод на русский язык. См. [Давенпорт 1971]. — *Прим. ред.*

*туральное число* $n < 2\ln^2 D$, *что* $\chi(n) \neq 1$, *и натуральное число* $m < 3\ln^2 D$, *такое, что* $\chi(m) \neq 1$ *и* $\chi(m) \neq 0$.

Этот результат содержится в статье [Bach 1990]. Тот факт, что для чисел $m, n$ при условии ERH справедлива оценка $O\left(\ln^2 D\right)$, был изначально установлен Анкени в 1952 г. Теорема 1.4.5 лежит в основе ERH-условных тестов на простоту с полиномиальным временем, а также применяется и в других задачах.

Для проверки ERH были проведены соответствующие вычисления, но они не продвинулись настолько далеко, как это было в случае гипотезы Римана. Нам известно, что эта гипотеза справедлива до высоты 10000 для всех характеров $\chi$ по модулям вплоть до 13 и до высоты 2500 для всех характеров $\chi$ по модулям вплоть до 72, а также для ряда других модулей (см. [Rumely 1993]). На основании этих данных в статье [Ramaré and Rumely 1996] были получены явные оценки для распределения простых чисел в определенных классах вычетов. (Румели проверил ERH до высоты 100000 для всех характеров по модулям вплоть до 9.) Отметим, что ERH влечет явную оценку остаточного члена в формуле (1.5) теоремы о простых числах в арифметических прогрессиях. А именно, при $x \geqslant 2$, $d \geqslant 2$ и НОД$(a, d) = 1$ имеет место неравенство

$$\left|\pi(x; d, a) - \frac{1}{\varphi(d)}\,\mathrm{li}\,(x)\right| < x^{1/2}(\ln x + 2\ln d) \quad \text{(при условии ERH)}. \tag{1.32}$$

Важным моментом здесь является то, что это неравенство не просто устанавливает точную оценку сверху остаточного члена, но и предоставляет *явное* значение постоянной (которое обычно скрывается за знаком $O$). Именно такие оценки позволяют сочетать теорию и вычисления, а также проверять выдвинутые гипотезы.

Еще одним ERH-условным результатом является следующий. Пусть $d > 2$ и НОД$(d, a) = 1$. Тогда найдется такое простое число $p \equiv a \pmod d$, что $p < 2d^2 \ln^2 d$ (по вопросам, связанным со следствиями из ERH см. [Bach and Shallit 1996]). Как и в случае асимптотического закона распределения простых чисел, безусловные результаты (т. е. независящие от ERH), касающиеся функции $\pi(x; d, a)$, являются менее точными. Например, следующая безусловная теорема имеет большое историческое значение.

**Теорема 1.4.6** (Зигель—Вальфиш). *Для любого* $\eta > 0$ *существует такое число* $C(\eta) > 0$, *что равенство*

$$\pi(x; d, a) = \frac{1}{\varphi(d)}\,\mathrm{li}\,(x) + O\left(x \exp\left(-C(\eta)\sqrt{\ln x}\right)\right)$$

*имеет место для всех взаимно простых чисел* $a, d$, *где* $d < \ln^{\eta} x$, *причем постоянная в знаке* $O$ *является абсолютной.*

Вопросы, связанные с этой и другими близкими ей теоремами, более подробно изложены в книге [Davenport 1980]. Будет небезынтересным отметить, что число $C(\eta)$ из теоремы 1.4.6 не было вычислено ни для одного значения $\eta \geqslant 1$. Более того, это число *не вычислимо* методами, вытекающими из доказательства теоремы. Необходимо отметить, что численные оценки разности

$\pi(x;d,a) - \frac{1}{\varphi(d)}\operatorname{li}(x)$ можно получить и при $1 \leqslant \eta < 2$, пусть и не с таким точным остатком, как в теореме 1.4.6. При $\eta \geqslant 2$ никаких численных оценок остаточного члена вида $o$-малое от главного члена не известно. Несмотря на то, что оценка остаточного члена, предоставляемая теоремой Зигеля—Вальфиша, существенно уступает следствиям из ERH, подобные результаты приобретают огромное значение в сочетании с другими аналитическими методами (см. разд 1.4.4).

В заключение темы, связанной с оценками функции $\pi(x;d,a)$, приведем результат совсем иного характера. Зачастую получение нетривиальной оценки снизу является глубокой и тонкой задачей. Но что если ограничиться поиском только оценки сверху? Для получения таких результатов хорошо подходит семейство методов аналитической теории чисел, называемых *методами решета*. Как и решето, применяемое в вычислительной теории чисел (см., например, разд. 3.2), эти методы восходят к решету Эратосфена, но сам подход не одинаков. В частности, именно методом решета Брун доказал оценку (1.8). Иногда методы решета позволяют получить и очень красивые неравенства с вычисленной явно константой. Одним из изящных примеров того служит следующая формулировка неравенства Бруна—Титчмарша (см. [Montgomery and Vaughan 1973][8]):

**Теорема 1.4.7** (неравенство Бруна—Титчмарша). *Пусть натуральные числа $a$ и $d$ таковы, что* $\operatorname{НОД}(a,d) = 1$. *Тогда для всех $x > d$ выполнено неравенство*

$$\pi(x;d,a) < \frac{2x}{\varphi(d)\ln(x/d)}.$$

## 1.4.4. Тригонометрические суммы

Помимо дзета-функции Римана и арифметических функций, в аналитической теории чисел имеются и другие не менее важные объекты, такие, как тригонометрические суммы. В этих суммах содержится, можно сказать, «спектральная» информация о некоторых функциях и числовых множествах. Таким образом, тригонометрические суммы играют роль мощного моста, соединяющего теорию чисел и теорию комплексных рядов Фурье. Пусть $f$ — действительнозначная функция, $t$ — вещественное число и $a < b$ — целые числа. Положим

$$E(f;a,b,t) = \sum_{a<n\leqslant b} e^{2\pi i t f(n)}. \tag{1.33}$$

Модуль каждого слагаемого такой тригонометрической суммы равен 1, но аргументы этих слагаемых, вообще говоря, различны. Поэтому, если эти аргументы в определенном смысле «случайны», или «некоррелированы», то можно ожидать некоторое взаимное уничтожение членов суммы, существенно уменьшающее значение $|E|$ по сравнению с тривиальной оценкой $b-a$. Тем самым, можно сказать, что сумма $E(f;a,b,t)$ некоторым образом измеряет распределение дробных долей последовательности $(tf(n))$, $a < n \leqslant b$. В частности,

[8] Русскоязычному читателю можно рекомендовать книгу [Хооли 1983]. — *Прим. ред.*

знаменитая теорема Вейля (см. [Weyl 1916]) утверждает, что последовательность $(f(n))$, $n = 1, 2, \ldots$, равномерно распределена по модулю 1 тогда и только тогда, когда для любого целого числа $h \neq 0$ справедливо соотношение $E(f; 0, N, h) = o(N)$. Хотя за ней неизменно стоит распределение дробных долей, теория тригонометрических сумм широко применяется во многих разделах теории чисел. Приведем краткий обзор теоретико-числовых исследований с использованием тригонометрических сумм, в заключение которого бегло рассмотрим основные этапы решения И. М. Виноградовым тернарной проблемы Гольдбаха.

Основы теории тригонометрических сумм были заложены еще Гауссом, но сама теория получила существенное развитие благодаря ключевой работе Вейля, в которой были получены точные оценки сверху для некоторых классов тригонометрических сумм. Кроме того, Вейль нашел довольно простой, но мощный метод, заключавшийся в оценке степеней модуля суммы $E$. Основным соотношением здесь явилось равенство

$$|E(f; a, b, t)|^2 = \sum_{n \in (a,b]} \sum_{k \in (a-n, b-n]} e^{2\pi \imath t (f(n+k) - f(n))}. \tag{1.34}$$

Как видно, в показателе возникает что-то похожее на производную функции $f$, а это позволяет получить некоторые оценки величины $|E|$ для *многочлена* $f$, рекурсивно уменьшая степень. Этот прием уменьшения степени в показателе может быть чрезвычайно плодотворным (см., например, упр. 1.66, а также упомянутые в нем упражнения).

Важной аналитической проблемой, для решения которой применяются тригонометрические суммы, является оценка роста дзета-функции Римана. Эта задача для функции $\zeta(\sigma + \imath t)$ при фиксированном действительном $\sigma$ и растущем действительном $t$ сводится к получению оценок для сумм вида

$$\sum_{N < n \leqslant 2N} \frac{1}{n^{\sigma + \imath t}},$$

причем эти оценки, в свою очередь, могут быть получены как следствие оценок тригонометрических сумм

$$E(f; N, 2N, t) = \sum_{N < n \leqslant 2N} e^{-\imath t \ln n},$$

где мы положили $f(n) = -(\ln n)/(2\pi)$. Отталкиваясь от работы Вейля, ван дер Корпут (см. [van der Corput 1922]) этим способом получил для $\zeta(\sigma + \imath t)$ оценку, которая выражалась нетривиальной степенью параметра $t$. В частности, он показал, что дзета-функция Римана на критической прямой $\sigma = 1/2$ допускает оценку

$$\zeta(1/2 + \imath t) = O(t^{1/6}),$$

где $t \geqslant 1$ (см. [Graham and Kolesnik 1991]). Показатель $1/6$ впоследствии с успехом уменьшался. Например, в статье [Bombieri and Iwaniec 1986] получена оценка $O\left(t^{9/56+\epsilon}\right)$, а в работе [Watt 1989] — $O\left(t^{89/560+\epsilon}\right)$. Гипотеза Линделёфа

утверждает, что $\zeta(1/2+it) = O(t^{\varepsilon})$ для всякого $\varepsilon > 0$. Следствия этой гипотезы также связаны с распределением простых чисел. Например, если через $p_n$ обозначить $n$-е простое число, то из гипотезы Линделёфа можно получить (см. [Yu 1996]) равенство

$$\sum_{p_n \leqslant x} (p_{n+1} - p_n)^2 = x^{1+o(1)}.$$

Наилучшей безусловной оценкой этой суммы является результат $O\left(x^{23/18+\varepsilon}\right)$ для любого $\varepsilon > 0$, принадлежащий Хиз-Брауну. Отметим такое следствие условной теоремы Ю [Yu 1996]: для каждого $\varepsilon > 0$ количество таких натуральных чисел $n \leqslant x$, что интервал $(n, n+n^{\varepsilon})$ содержит простое число, асимптотически эквивалентно $x$. Кроме того, существует связь между гипотезами Римана и Линделёфа: первая влечет последнюю.

Не так легко, но все же возможно получать явные численные оценки при помощи тригонометрических сумм. Например, в статье [Ford 2002] показано, что

$$|\zeta(\sigma+it)| \leqslant 76.2 t^{4.45(1-\sigma)^{3/2}} \ln^{2/3} t$$

при $1/2 \leqslant \sigma \leqslant 1$ и $t \geqslant 2$. Такие результаты позволяют получить явные численные оценки для границы нулей дзета-функции Римана и другие оценки в теоретико-числовых задачах.

При решении аддитивных задач с простыми числами удобно рассматривать еще один важный класс тригонометрических сумм вида

$$E_n(t) = \sum_{p \leqslant n} e^{2\pi i t p}, \tag{1.35}$$

где $p$ пробегает множество всех простых чисел. Оказывается, существует глубокая взаимосвязь между интегралами по ограниченным областям, содержащими сумму $E_n(t)$, и свойствами простых чисел. В частности, доказательство И. М. Виноградова того утверждения, что каждое достаточно большое нечетное натуральное число представимо в виде суммы трех простых чисел, начинается, по существу, с того замечательного наблюдения, что количество искомых представлений числа $n$ в точности равно

$$\begin{aligned} R_3(n) &= \int_0^1 \sum_{n \geqslant p,q,r \in \mathcal{P}} e^{2\pi i t(p+q+r-n)}\, dt \\ &= \int_0^1 E_n^3(t) e^{-2\pi i t n}\, dt. \end{aligned} \tag{1.36}$$

Доказательство Виноградова было развитием более раннего «кругового метода» Харди и Литтлвуда (см. монументальный сборник [Hardy 1966]), который явился *tour de force* аналитической теории чисел, соединившим тригонометрические суммы с основными проблемами аддитивной теории чисел, такими, как проблема Гольдбаха.

Настало время рассказать, в чем же заключается метод Виноградова для оценки интеграла (1.36). Основное наблюдение здесь состоит в том, что между

распределением простых чисел и *спектральной* информацией, содержащейся в сумме $E_n(t)$, имеется ярко выраженное соответствие. Пусть у нас имеется общая оценка количества простых чисел, не превосходящих $n$, принадлежащих арифметической прогрессии $\{a, a+d, a+2d, \ldots\}$, где НОД$(a,d)=1$, вида

$$\pi(n;d,a) = \frac{1}{\varphi(d)}\pi(n) + \varepsilon(n;d,a),$$

причем мы будем считать, что эта оценка достаточно «хороша» в том смысле, что остаточный член $\varepsilon$ настолько мал, насколько это необходимо для рассматриваемой проблемы. (Нам уже встречались такие оценки: соотношение (1.32) следует из ERH, а теорема 1.4.6 не так сильна, но зато безусловна.) Тогда для рационального числа $t=a/q$ справедливы следующие преобразования суммы (1.35):

$$\begin{aligned} E_n(a/q) &= \sum_{f=0}^{q-1} \sum_{p\equiv f \pmod q,\ p\leqslant n} e^{2\pi i p a/q} \\ &= \sum_{\text{НОД}(f,q)=1} \pi(n;q,f) e^{2\pi i f a/q} + \sum_{p\mid q,\ p\leqslant n} e^{2\pi i p a/q} \\ &= \sum_{\text{НОД}(f,q)=1} \pi(n;q,f) e^{2\pi i f a/q} + O(q), \end{aligned}$$

где в суммах, содержащих НОД, пробегаются все числа $f\in[1,q-1]$, взаимно простые с числом $q$. Оказывается, что подобные оценки приобретают наибольшее значение именно тогда, когда знаменатель $q$ сравнительно мал. В таком случае можно воспользоваться приведенной выше оценкой количества простых чисел в классе вычетов и получить соотношение

$$E_n(a/q) = \frac{c_q(a)}{\varphi(q)}\pi(n) + O(q + |\varepsilon|\varphi(q)),$$

где через $|\varepsilon|$ обозначен максимум выражения $|\varepsilon(n;q,f)|$ по всем вычетам $f$, взаимно простым с числом $q$, а через $c_q$ — хорошо изученная сумма Рамануджана

$$c_q(a) = \sum_{\text{НОД}(f,q)=1} e^{2\pi i f a/q}. \tag{1.37}$$

Сумма Рамануджана встретится нам далее, когда речь будет идти о методах дискретной свертки (см. уравнение (9.26)), а сейчас мы приведем результат [Hardy and Wright 1979]

$$c_q(a) = \frac{\mu(q/g)\varphi(q)}{\varphi(q/g)}, \qquad g=\text{НОД}(a,q). \tag{1.38}$$

В частности, когда числа $a, q$ взаимно просты, мы получаем красивое равенство вида

$$E_n(a/q) = \sum_{p\leqslant n} e^{2\pi i p a/q} = \frac{\mu(q)}{\varphi(q)}\pi(n) + \varepsilon', \tag{1.39}$$

где совокупный остаточный член $\varepsilon'$ сложным образом зависит от $a$, $q$ и $n$, а также, разумеется, и от выбора остатка в теореме о распределении простых чисел в классах вычетов. Тем самым, установлено фундаментальное спектральное свойство простых чисел: если знаменатель $q$ мал, то величина тригонометрической суммы допускает эффективное понижение с явным множителем $\mu/\varphi$ по сравнению с тривиальной оценкой $\pi(n)$. Это понижение происходит, конечно, за счет взаимного уничтожения осциллирующих слагаемых, что количественно описывает равенство (1.39).

И. М. Виноградов смог воспользоваться результатом для малых знаменателей $q$ следующим образом. Выберем параметр $Q = \ln^B n$, $B$ — достаточно большое число, и будем считать число $q$ «малым», если $1 \leqslant q \leqslant Q$. (Оказывается, что «большие» знаменатели $q$ достаточно рассмотреть лишь в промежутке $Q < q < n/Q$.) Далее, подынтегральное выражение в равенстве (1.36) выражает «резонанс», когда переменная интегрирования $t$ приближается к рациональному числу $a/q$ с малым знаменателем. Эти промежутки изменения переменной $t$ традиционно называются «большими дугами». Оставшееся множество изменения переменной интегрирования разбивается на «малые дуги», на которых $t \approx a/q$, где $q \in (Q, n/Q)$; их можно рассматривать как «шум», требующий оценки сверху. Аккуратные преобразования приводят к оценке интеграла вида

$$R_3(n) = \frac{n^2}{2\ln^3 n} \sum_{q=1}^{Q} \frac{\mu(q) c_q(n)}{\varphi^3(q)} + \varepsilon'', \tag{1.40}$$

где налицо резонансная сумма от больших дуг, а $\varepsilon''$ включает как прежние остатки для арифметических прогрессий, так и шум от малых дуг. Уже из приведенной суммы по $q \in [1, Q]$ можно, проведя некоторые дополнительные алгебраические выкладки, получить для тернарной проблемы Гольдбаха итоговую оценку (1.12), поскольку остаток $\varepsilon''$, а также конечность верхней границы суммирования $Q$, не являются большими препятствиями (см. упр. 1.68).

Центральным достижением Виноградова стала оценка сверху вклада от малых дуг в совокупный остаточный член $\varepsilon''$. Было доказано, что если $\text{НОД}(a, q) = 1$, $q \leqslant n$ и действительное число $t$ близко́ к дроби $a/q$ в смысле неравенства $|t - a/q| \leqslant 1/q^2$, то

$$|E_n(t)| < C \left( \frac{n}{q^{1/2}} + n^{4/5} + n^{1/2} q^{1/2} \right) \ln^3 n, \tag{1.41}$$

где $C$ — абсолютная постоянная. Доказательство этого глубокого результата далеко не тривиально и основано на тонкой технике работы с арифметическими функциями. Тем не менее, впоследствии был получен ряд улучшений, самое значительное из которых принадлежит Р. Вону (см. ссылки ниже). Отметим, что оценка Виноградова сильна тем, что при $q \in (Q, n/Q)$ и действительном числе $t$ из теоремы величина $E_n(t)$ отличается от общего числа слагаемых $\pi(n)$ множителем, равным степени логарифма. Тем самым, оценка шума малых дуг оказалась достаточной для полного решения тернарной проблемы Гольдбаха. (К сожалению, даже этот мощный метод не позволяет решить *бинарную* про-

блему Гольдбаха: соответствующий остаточный член $\varepsilon''$ включает еще более «шумные» компоненты, оценить которые чрезвычайно трудно.)

Подведем итог. Оценка (1.39) применяется для «резонансов» больших дуг, и дает главный член формулы (1.40), а при помощи результата (1.41) можно оценить «шум» малых дуг, а следовательно, и совокупный остаток $\varepsilon''$. Соотношение (1.40) в конце концов приводит к результату (1.12) в тернарной проблеме Гольдбаха. Читатель, желающий познакомиться с полным изложением данного нами наброска, а также найти подробный материал по другим аддитивным задачам, может обратиться к следующим[9] источникам: [Hardy 1966], [Davenport 1980], [Vaughan 1977, 1997], [Ellison and Ellison 1985, Theorem 9.4], [Nathanson 1996, Theorem 8.5], [Vinogradov 1985], [Estermann 1952].

Как мы только что видели, метод тригонометрических сумм может быть чрезвычайно эффективным, причем его применение не ограничивается одной только проблемой Гольдбаха или даже всеми аддитивными задачами. Ниже мы рассмотрим фундаментальный вклад Гаусса в развитие теории квадратичных сумм. В частности, мы увидим, что в определение 2.3.6 входят вариации формы (1.33) с квадратичной функцией $f$. В разд. 9.5.3 мы обратимся к теме дискретной свертки (в противоположность непрерывности в интегралах) и укажем в тексте и упражнениях, каким образом обработка сигналов и особенно дискретный спектральный анализ связаны с аналитической теорией чисел. Более того, тригонометрические суммы положили начало привлекательным и поучительным вычислительным экспериментам и исследовательским задачам. Для удобства читателя приведем список соответствующих упражнений: 1.35, 1.66, 1.68, 1.70, 2.27, 2.28, 9.41, 9.80.

### 1.4.5. Гладкие числа

Гладкие числа крайне важны для наших вычислительных целей, особенно для решения задач разложения на множители. Помимо этого, гладкие числа допускают ряд интересных теоретических приложений, в частности, к проблеме Варинга (см. [Vaughan 1989]). Начнем с фундаментального определения.

**Определение 1.4.8.** Натуральное число называется $y$-гладким, если у него отсутствуют простые делители, превосходящие $y$.

Так в чем же заключается польза гладких чисел? В основном в том, что $y$-гладкие числа при малых значениях $y$ имеют простую мультипликативную структуру и, тем не менее, они удивительно многочисленны. Например, хотя лишь исчезающе малая доля простых чисел из промежутка $[1, x]$ лежит в промежутке $[1, \sqrt{x}]$, более 30% чисел промежутка $[1, x]$ являются $\sqrt{x}$-гладкими (при достаточно больших значениях $x$). Еще одним примером неожиданно высокой частоты гладких чисел является то факт, что количество $(\ln^2 x)$-гладких чисел, не превосходящих $x$, превосходит $\sqrt{x}$ для всех достаточно больших значений $x$.

[9] См. также литературу, добавленную при переводе: [Виноградов 1937], [Давенпорт 1971], [Карацуба 1983]. — *Прим. ред.*

Эти примеры наталкивают на мысль о рассмотрении функции, подсчитывающей гладкие числа. Положим

$$\psi(x,y) = \#\{1 \leqslant n \leqslant x : n \text{ есть } y\text{-гладкое число}\}. \tag{1.42}$$

В 1930 г. для этой функции Дикманом доказана следующая теорема.

**Теорема 1.4.9** (Дикман). *Для каждого фиксированного действительного числа $u > 0$ существует такое действительное число $\rho(u) > 0$, что*

$$\psi(x, x^{1/u}) \sim \rho(u)x.$$

Более того, Дикман показал, что функция $\rho(u)$ является решением некоторого дифференциального уравнения: это единственная непрерывная функция, определенная на луче $[0,\infty)$, удовлетворяющая условиям

(A) $\rho(u) = 1$ при $0 \leqslant u \leqslant 1$;
(B) $\rho'(u) = -\rho(u-1)/u$ при $u > 1$.

В частности, $\rho(u) = 1 - \ln u$ при $1 \leqslant u \leqslant 2$, но решений в аналитическом виде (в элементарных функциях) при $u > 2$ нет. Для функции $\rho(u)$ получены численные приближения (ср. упр. 3.5), из которых становится ясно, что эта функция быстро стремится к нулю. Более того, она убывает быстрее функции $u^{-u}$, хотя это простое выражение может выступать в качестве разумной оценки функции $\rho(u)$ в различных исследованиях сложности. В частности, справедливо соотношение

$$\ln \rho(u) \sim -u \ln u. \tag{1.43}$$

Теорема 1.4.9 прекрасно подходит для оценки функции $\psi(x,y)$ в том случае, когда числа $x$ и $y$ стремятся к бесконечности так, что отношение $u = \ln x / \ln y$ остается постоянным или ограниченным. Но как оценить $\psi\left(x, x^{1/\ln\ln x}\right)$, или $\psi\left(x, e^{\sqrt{\ln x}}\right)$, или $\psi\left(x, \ln^2 x\right)$? Оценки этих и других аналогичных выражений приобрели особое значение приблизительно в 1980 г., когда начались теоретические исследования субэкспоненциальных алгоритмов разложения на множители (см. гл. 6). Заполняя этот пробел, Э. Кэнфилд и др. [Canfield et al. 1983] показали, что

$$\psi\left(x, x^{1/u}\right) = xu^{-u+o(u)} \tag{1.44}$$

равномерно при $u \to \infty$ и $u < (1-\varepsilon)\ln x / \ln\ln x$. Отметим, что этот результат являлся ожидаемым, поскольку согласно (1.43) справедливо равенство $\rho(u) = u^{-u+o(u)}$. Таким образом, у нас есть приемлемая оценка функции $\psi(x,y)$ при $y > \ln^{1+\varepsilon} x$ и большом $x$. (На самом деле имеются неплохие оценки и при меньших значениях $y$, но в этой книге они нам не понадобятся.)

Для функции $\psi(x,y)$ можно получить и явные неравенства. Например, в статье [Konyagin and Pomerance 1997] показано, что при всех $x \geqslant 4$ и $2 \leqslant x^{1/u} \leqslant x$ справедливо неравенство

$$\psi\left(x, x^{1/u}\right) \geqslant \frac{x}{\ln^u x}. \tag{1.45}$$

Содержащийся здесь результат приемлем тогда, когда $x^{1/u} = \ln^c x$, где число $c > 1$ фиксировано (см. упр. 1.72, 3.19 и 4.28).

Как отмечалось выше, гладкие числа встречаются в многочисленных алгоритмах разложения на множители и именно в этом контексте возникают в нашей книге далее. Вычислительная проблема распознавания гладких чисел из заданного множества рассмотрена в гл. 3. Довольно полный обзор сведений о гладких числах содержится в статье [Granville 2004b].

## 1.5. Упражнения

**1.1.** Найдите наибольшее натуральное число $N$, обладающее следующим свойством: все числа из промежутка $[2, \ldots, N-1]$, взаимно простые с числом $N$, сами являются простыми. Найдите наибольшее натуральное число $N$, которое делится на все числа, меньшие $\sqrt{N}$.

**1.2.** Докажите «первую теорему» Евклида: произведение двух чисел кратно простому числу $p$ тогда и только тогда, когда хотя бы одно из них кратно $p$. Выведите из нее теорему 1.1.1 как следствие.

**1.3.** Докажите, что натуральное число $n$ является простым тогда и только тогда, когда

$$\sum_{m=1}^{\infty} \left( \left\lfloor \frac{n}{m} \right\rfloor - \left\lfloor \frac{n-1}{m} \right\rfloor \right) = 2.$$

**1.4.** Докажите, что для всех целых значений $x \geqslant 2$ справедлива формула

$$\pi(x) = \sum_{n=2}^{x} \left\lfloor \frac{1}{\sum_{k=2}^{n} \lfloor \lfloor n/k \rfloor k/n \rfloor} \right\rfloor.$$

**1.5.** Иногда формула для простых чисел хотя и неэффективна с вычислительной точки зрения, но имеет реальный методический смысл. Докажите формулу Ганди для $n$-го простого числа:

$$p_n = \left\lfloor 1 - \log_2 \left( -\frac{1}{2} + \sum_{d \mid p_n - 1,} \frac{\mu(d)}{2^d - 1} \right) \right\rfloor.$$

(Указание: поучительный вариант доказательства заключается в применении (символически) решета Эратосфена (см. гл. 3) для двоичного представления $1 = (0.11111\ldots)_2$.)

**1.6.** Усовершенствовав метод доказательства теоремы 1.1.2, можно получить (пусть и относительно слабые) оценки снизу для функции $\pi(x)$. Рассмотрим «приориал» числа $p$, определенный равенством

$$p\# = \prod_{q \leqslant p} q = 2 \cdot 3 \cdots p,$$

где символ $q$ пробегает простые числа. Проведите рассуждения, следуя доказательству Евклида, показывающие, что $n$-е простое число $p_n$ при $n \geqslant 3$

удовлетворяет неравенству

$$p_n < p_{n-1}\#.$$

Затем по индукции докажите, что

$$p_n \leqslant 2^{2^{n-1}},$$

и получите оценку

$$\pi(x) > \frac{1}{\ln 2} \ln \ln x$$

для $x \geqslant 2$.

Отметим, что численное изучение примориальных простых чисел вида $p\# + 1$ представляет отдельный интерес. Современным примером большого примориального простого числа является найденное в 1999 г. Колдвеллом число $42209\# + 1$, состоящее более чем из восемнадцати тысяч цифр.

**1.7.** Рассматривая числа вида

$$n = 2^2 \cdot 3 \cdot 5 \cdot \ldots \cdot p - 1,$$

докажите существование бесконечного множества простых чисел, сравнимых с 3 по модулю 4. Найдите аналогичное доказательство для простых чисел, сравнимых с 2 по модулю 3 (ср. упр. 5.22).

**1.8.** Рассматривая числа вида

$$(2 \cdot 3 \cdot \ldots \cdot p)^2 + 1,$$

докажите существование бесконечного множества простых чисел, сравнимых с 1 по модулю 4. Найдите аналогичное доказательство для простых чисел, сравнимых с 1 по модулю 3.

**1.9.** Пусть $a, n$ — натуральные числа, причем $a \geqslant 2$. Положим $N = a^n - 1$. Покажите, что порядок элемента $a \pmod N$ мультипликативной группы $\mathbf{Z}_N^*$ равен $n$, и выведите отсюда, что $n \mid \varphi(N)$. Применяя полученные результаты, докажите, что если $n$ — простое число, то существует бесконечно много простых чисел, сравнимых с 1 по модулю $n$.

**1.10.** Пусть $\mathcal{S}$ — непустое множество простых чисел, сумма обратных величин к которым равна $S < \infty$. Пусть, далее, $\mathcal{A}$ — множество всех натуральных чисел, не кратных ни одному элементу множества $\mathcal{S}$. Покажите, что асимптотическая плотность множества $\mathcal{A}$ меньше, чем $e^{-S}$. Выведите отсюда, что если сумма обратных величин к элементам множества $\mathcal{S}$ бесконечна, то асимптотическая плотность множества $\mathcal{A}$ равна нулю. На основании того факта, что ряд, составленный из всех чисел, обратных к простым числам вида $4k+3$, расходится, докажите, что множество натуральных чисел, представимых в виде суммы двух взаимно простых квадратов, имеет нулевую асимптотическую плотность (см. упр. 1.91 и 5.16.)

**1.11.** Отталкиваясь от того факта, что ряд, составленный из всех чисел, обратных к простым, расходится, используйте упр. 1.10 для доказательства того, что множество простых чисел имеет нулевую асимптотическую плотность, т. е. справедливо соотношение $\pi(x) = o(x)$.

**1.12.** Как утверждается в тексте, «вероятность» того, что случайное натуральное число $x$ окажется простым, «примерно равна» $1/\ln x$. Воспользуйтесь теоремой о простых числах, чтобы перевести это вероятностное соображение на строгий математический язык.

**1.13.** Отталкиваясь от определения

$$\phi(x,y) = \#\{1 \leqslant n \leqslant x : \text{ каждое простое число, делящее } n, \text{ превосходит } y\}$$

(которое появится позже, в разд. 3.7.1, в связи с подсчетом количества простых чисел), покажите, что

$$\phi(x, \sqrt{x}) = \pi(x) - \pi(\sqrt{x}) + 1.$$

Затем докажите классическую формулу Лежандра

$$\pi(x) = \pi(\sqrt{x}) - 1 + \sum_{d \mid Q} \mu(d) \left\lfloor \frac{x}{d} \right\rfloor, \tag{1.46}$$

где $Q$ — некоторое произведение простых чисел, а именно

$$Q = \prod_{p \leqslant \sqrt{x}} p.$$

Подобные комбинаторные рассуждения помогли в свое время Лежандру показать, что $\pi(x) = o(x)$. Для этого докажите равенство

$$\phi(x,y) = x \prod_{p \leqslant y} \left(1 - \frac{1}{p}\right) + E,$$

где для остаточного члена $E$ имеет место оценка $E = O(2^{\pi(y)})$. Наконец, из этого равенства и того факта, что ряд, составленный из всех чисел, обратных к простым, расходится, выведите, что $\pi(x)/x \to 0$ при $x \to \infty$ (ср. упр. 1.11).

**1.14.** Выведите из основной теоремы арифметики 1.1.1, что для любого фиксированного $\varepsilon > 0$ количество $d(n)$ делителей числа $n$ (включая 1 и $n$) допускает оценку

$$d(n) = O(n^{\varepsilon}).$$

Как зависит постоянная в знаке $O$ от выбора числа $\varepsilon$? Исследование этого вопроса можно начать с доказательства того, что при фиксированном $\varepsilon$ существует лишь конечное число степеней $q$ простых чисел, таких, что $d(q) > q^{\varepsilon}$.

**1.15.** Рассмотрим сумму обратных величин ко всем числам Мерсенна $M_n = 2^n - 1$ (для натуральных показателей $n$), а именно,

$$E = \sum_{n=1}^{\infty} \frac{1}{M_n}.$$

Докажите, что справедливо еще одно представление этой суммы, содержащее функцию делителей $d$ (определенную в упр. 1.14):

$$E = \sum_{k=1}^{\infty} \frac{d(k)}{2^k}.$$

Кроме того, можно придать этой сумме более чем линейную скорость сходимости. Для этого докажите, что имеет место и такое равенство:

$$E = \sum_{m=1}^{\infty} \frac{1}{2^{m^2}} \frac{2^m + 1}{2^m - 1}.$$

Отметим, что некоторые свойства числа $E$ хорошо изучены. Например, известно (см. [Erdős 1948], [Borwein 1991]), что число $E$ иррационально, однако его не удалось выразить через фундаментальные постоянные. Возможные подходы к дальнейшему изучению числа $E$ описаны в работе [Bailey and Crandall 2002].

Пусть теперь суммирование ведется по всем *простым* числам Мерсенна. Будем полагать, что в табл. 1.2 не пропущено ни одно такое простое число (напомним, что это еще не доказано) до наибольшего из известных. На основании этих данных найдите количество точных знаков числа

$$\sum_{M_q \in \mathcal{P}} \frac{1}{M_q}.$$

**1.16.** Проверьте, что многочлен Эйлера $x^2+x+41$ принимает простые значения для всех целых чисел $x$, таких, что $-40 \leqslant x \leqslant 39$. Покажите, что если целочисленный многочлен $f(x)$ отличен от постоянной, то существует бесконечное множество целых значений переменной $x$, для которых число $f(x)$ составное.

**1.17.** Может случиться так, что хотя данный многочлен *не всегда* принимает простые значения, тем не менее, он делает это на определенных промежутках довольно часто. Покажите при помощи вычислений, что многочлен Дресса и Оливье [Dress and Olivier 1999]

$$f(x) = x^2 + x - 1354363$$

обладает тем потрясающим свойством, что для случайного целого числа $x \in [1, 10^4]$ вероятность простоты величины $|f(x)|$ превосходит $1/2$. Занимательное следствие таково: если Вам удастся запомнить семизначный телефонный номер 135–43–63, то тем самым Вы запомните и способ получения нескольких тысяч простых чисел.

**1.18.** Рассмотрим последовательность простых чисел $2, 3, 5, 11, 23, 47$, каждое из которых (кроме первого) получается прибавлением единицы к удвоенному предыдущему. Покажите, что вне зависимости от выбора первого простого числа такая цепочка всегда будет конечной.

**1.19.** Как отмечено в тексте, равенство

$$\frac{1}{\zeta(s)} = \sum_{n=1}^{\infty} \frac{\mu(n)}{n^s}$$

имеет место при $\mathrm{Re}(s) > 1$, причем ряд сходится абсолютно. Докажите это. Но доказать эквивалентное теореме о простых числах равенство

$$\sum_{n=1}^{\infty} \frac{\mu(n)}{n} = 0,$$

получающееся при $s \to 1$, не так просто. На этом пути можно указать два полезных шага. Сначала посредством численных экспериментов получите оценку порядка роста частичной суммы

$$\sum_{n \leqslant x} \frac{\mu(n)}{n}$$

как функции от $x$, а затем проведите хотя бы эвристические рассуждения в пользу того, что эта сумма стремится к нулю при $x \to \infty$. На первом шаге Вас ожидает интересная вычислительная задача об эффективной реализации самой функции $\mu$, а на втором Вы можете рассмотреть несколько первых членов ряда, записанных в виде

$$1 - \sum_{p \leqslant x} \frac{1}{p} + \sum_{pq \leqslant x} \frac{1}{pq} - \ldots,$$

что позволит увидеть, почему сумма стремится к нулю с ростом $x$. Интересно, что даже не опираясь на теорему о простых числах, можно доказать ограниченность рассматриваемой суммы при $x \to \infty$, как это сделал Дж. Грэм в 1884 г. (см. [Ribenboim 1996]).

**1.20.** Докажите, что при всех $x > 1$ выполняется неравенство

$$\sum_{p \leqslant x} \frac{1}{p} > \ln \ln x - 1,$$

где $p$ пробегает простые числа. Выведите отсюда, что простых чисел бесконечно много. Доказательство можно разбить на два следующих промежуточных шага.

(1) Покажите, что $\sum_{n=1}^{\lfloor x \rfloor} \frac{1}{n} > \ln x$.
(2) Покажите, что $\sum \frac{1}{n} = \prod_{p \leqslant x} (1 - \frac{1}{p})^{-1}$, где суммирование ведется по всем натуральным числам $n$, не кратным ни одному простому числу, превосходящему $x$.

**1.21.** Воспользуйтесь мультиномиальной[10] теоремой (обобщением биномиальной) для доказательства неравенства

$$\frac{1}{u!} \left( \sum_{p \leqslant x} \frac{1}{p} \right)^u \leqslant \sum_{n \leqslant x^u} \frac{1}{n},$$

где натуральное число $u$ и действительное число $x > 0$ произвольны, $p$ пробегает простые, а $n$ — натуральные числа. Положив в этом неравенстве $u = \lfloor \ln \ln x \rfloor$, покажите, что при $x \geqslant 3$ справедлива оценка

$$\sum_{p \leqslant x} \frac{1}{p} \leqslant \ln \ln x + O(\ln \ln \ln x).$$

[10] См. [Архипов и др. 2004, гл. 2, § 1]. — *Прим. ред.*

**1.22.** Рассмотрите наибольшую степень данного простого числа, которая делит факториал, и выведите формулу

$$N! = \prod_{p \leqslant N} p^{\sum_{k=1}^{\infty} \lfloor N/p^k \rfloor},$$

где произведение берется по простым числам $p$. Затем при помощи неравенства

$$N! > \left(\frac{N}{e}\right)^N$$

(которое следует из соотношения $e^N = \sum_{k=0}^{\infty} N^k/k! > N^N/N!$) докажите, что

$$\sum_{p \leqslant N} \frac{\ln p}{p-1} > \ln N - 1.$$

Выведите отсюда, что простых чисел бесконечно много.

**1.23.** Воспользуйтесь асимптотической формулой Стирлинга

$$N! \sim \left(\frac{N}{e}\right)^N \sqrt{2\pi N},$$

а также методом упр. 1.22 для доказательства соотношения

$$\sum_{p \leqslant N} \frac{\ln p}{p} = \ln N + O(1).$$

Выведите отсюда, что для функции $\pi(x)$ справедлива оценка $\pi(x) = O(x/\ln x)$, причем если $\pi(x) \sim cx/\ln x$ для некоторой постоянной $c$, то $c = 1$.

**1.24.** Выведите из теоремы Чебышёва 1.1.3 следующие оценки для $n$-го простого числа $p_n$ при $n \geqslant 2$:

$$Cn \ln n < p_n < Dn \ln n,$$

где $C, D$ — абсолютные постоянные.

**1.25.** Будучи совсем юным, П. Эрдёш доказал неравенство типа Чебышёва:

$$\prod_{p \leqslant x} p < 4^x$$

при всех $x > 0$. Попытайтесь доказать это неравенство. Начать можно, например, с того замечания, что достаточно провести рассуждения лишь для нечетных чисел $x$. Затем можно воспользоваться индукцией и неравенствами

$$\prod_{n+1 < p \leqslant 2n+1} p \leqslant \binom{2n+1}{n} \leqslant 4^n.$$

**1.26.** Используя результат из упр. 1.25, докажите оценку $\pi(x) = O(x/\ln x)$ (ср. упр. 1.23).

**1.27.** Докажите следующую теорему Чебышёва, известную как постулат Бертрана: для любого натурального числа $N$ на промежутке $(N, 2N]$ найдется не

менее одного простого числа. Благодаря следующей известной песенке, постулат Бертрана относится к общеизвестным знаниям американских студентов в области теории чисел:

*Чебышёв объявил,*
*что найдется всегда*
*простое число*
*между $A$ и $2A$.*

Наметим один из возможных вариантов доказательства. Пусть $P$ — произведение всех простых чисел $p$ из промежутка $N < p \leqslant 2N$. Нам требуется доказать, что $P > 1$. Покажите, что $P$ делит $\binom{2N}{N}$. Пусть, далее, число $Q$ таково, что $\binom{2N}{N} = PQ$. Докажите, что если $q^a$ есть точная степень простого числа $q$, делящая $Q$, то $a \leqslant \ln(2N)/\ln q$. Покажите, далее, что наибольший простой делитель числа $Q$ не превосходит $2N/3$. При помощи упр. 1.25 покажите, что

$$Q < 4^{\frac{2}{3}N} 4^{(2N)^{1/2}} 4^{(2N)^{1/3}} \dots 4^{(2N)^{1/k}},$$

где $k = \lfloor \lg(2N) \rfloor$. Выведите отсюда, что

$$P > \binom{2N}{N} 4^{-\frac{2}{3}N - (2N)^{1/2} - (2N)^{1/3} \lg(N/2)}.$$

Далее, докажите неравенство $\binom{2N}{N} > 4^N/N$ при $N \geqslant 4$ (по индукции) и заключите, что $P > 1$ при $N \geqslant 250$. Завершите доказательство, проведя прямые вычисления для оставшихся значений $N$.

**1.28.** Согласно упр. 1.25, $\prod_{p \leqslant x} p < 4^x$ для всех $x > 0$. Здесь мы предлагаем доказать явную нижнюю оценку этого произведения простых чисел:

$$\prod_{p \leqslant x} p > 2^x \quad \text{для всех} \quad x \geqslant 31.$$

В то время как теорема о простых числах равносильна тому, что произведение всех простых чисел из промежутка $[1, x]$ есть $e^{(1+o(1))x}$ при $x \to +\infty$, получение таких явных оценок этого произведения, как эти, представляет немалый интерес.

Пусть

$$C(N) = \frac{(6N)!N!}{(3N)!(2N)!(2N)!},$$

где $N$ — натуральное число.

(1) Докажите, что число $C(N)$ целое.
(2) Докажите, что если простое число $p$ таково, что $p > (6N)^{1/k}$, то $p^k$ не делит $C(N)$.
(3) При помощи упр. 1.25 и идеи упр. 1.27 докажите неравенство

$$\prod_{p \leqslant 6N} p > C(N)/4^{(6N)^{1/2} + (6N)^{1/3} \lg(1\,5N)}.$$

(4) Воспользуйтесь формулой Стирлинга (или методом математической индукции) и докажите оценку $C(N) > 108^N/(4\sqrt{N})$ для всех $N$.

(5) Докажите, что $\prod_{p\leqslant x} p > 2^x$ при $x \geqslant 2^{12}$.
(6) Завершите доказательство прямыми вычислениями для $31 \leqslant x < 2^{12}$.

**1.29.** При помощи упр. 1.28 докажите неравенство $\pi(x) > x/\lg x$ для всех $x \geqslant 5$. Поскольку здесь значится двоичный, а не натуральный логарифм, эту оценку функции $\pi(x)$ можно в шутку назвать «законом распределения простых чисел для программистов». Из упр. 1.25 выведите неравенство $\pi(x) < 2x/\ln x$ для всех $x > 0$. При этом может оказаться полезным сначала доказать равенство

$$\pi(x) = \frac{\theta(x)}{\ln x} + \int_2^x \frac{\theta(t)}{t\ln^2 t}\,dt,$$

где $\theta(x) \coloneqq \sum_{p\leqslant x} \ln p$. Заметим, что две части этого упражнения доказывают теорему 1.1.3.

**1.30.** Это упражнение содержит любопытное сочетание вычислений и теории. Обозначим сумму всех делителей числа $n$ через $\sigma(n)$. Напомним, что непосредственно перед формулировкой теоремы 1.3.3 мы говорили о том, что $n$ — совершенное число тогда и только тогда, когда $\sigma(n) = 2n$. Пусть $n = p_1^{t_1}\dots p_k^{t_k}$ — каноническое разложение числа $n$. Мы предлагаем читателю следующую последовательность действий.

(1) Получите условие на числа $p_i, t_i$, эквивалентное равенству $\sigma(n) = 2n$.
(2) При помощи результата пункта (1) получите (не прибегая к значительным машинным вычислениям) какую-либо нижнюю оценку для наименьшего нечетного совершенного числа. Например, покажите, что любое нечетное совершенное число должно, по меньшей мере, превышать $10^6$.
(3) Число $n$, для которого справедливо неравенство $\sigma(n) > 2n$, называется «избыточным» (например, $\sigma(12) = 28$). Найдите (при помощи ручки и бумаги или же небольшого вычислительного перебора) *нечетное* избыточное число. Всегда ли нечетное избыточное число делится на 3?
(4) Для нечетных значений $n$ исследуйте вероятность получить «почти» совершенное число. Например, покажите (возможно, при помощи ЭВМ), что каждое нечетное число $n$ из интервала $10 < n < 10^6$ удовлетворяет неравенству $|\sigma(n) - 2n| > 5$.
(5) Объясните, почему функция $\sigma(n)$ почти всегда принимает четные значения. Далее, покажите, что количество таких чисел $n \leqslant x$, для которых $\sigma(n)$ нечетно, равно $\lfloor\sqrt{x}\rfloor + \lfloor\sqrt{x/2}\rfloor$.
(6) Покажите, что для любого фиксированного целого числа $k > 1$ множество таких натуральных чисел $n$, что $k \mid \sigma(n)$, имеет асимптотическую плотность 1. (Указание: воспользуйтесь теоремой Дирихле 1.1.5.) Случай $k = 4$ является более простым, по сравнению с общим. Рассмотрев этот случай, покажите, что множество нечетных совершенных чисел имеет асимптотическую плотность 0.
(7) Положим $s(n) = \sigma(n) - n$ для натуральных чисел $n$ и $s(0) = 0$. Тогда условие избыточности числа $n$ перепишется в виде $s(n) > n$. Пусть $s^{(k)}(n)$ обозначает результат применения $k$ раз функции $s$ к числу $n$. Восполь-

зуйтесь теоремой Дирихле 1.1.5 для доказательства следующего утверждения, принадлежащего Ленстре мл.: для каждого натурального числа $k$ существует такое число $n$, что справедлива цепочка неравенств

$$n < s^{(1)}(n) < s^{(2)}(n) < \ldots < s^{(k)}(n). \tag{1.47}$$

На сегодняшний день остается загадкой, существует ли такое число $n$, для которого эта цепочка имеет место для *всех* значений $k$, а также существует ли такое число $n$, для которого последовательность $\left(s^{(k)}(n)\right)$ неограничена. Наименьшим числом $n$, которое может обладать последним из этих свойств, является 276. П. Эрдёш доказал, что для каждого фиксированного числа $k$ множество таких чисел $n$, для которых $n < s(n)$, но соотношение (1.47) не выполняется, имеет асимптотическую плотность 0.

**1.31.** Докажите следующий результат Р. Вона. Пусть $c_q(n)$ обозначает сумму Рамануджана, определенную равенством (1.37). Тогда для того чтобы число $n$ было совершенным, необходимо и достаточно, чтобы выполнялось равенство

$$\sum_{q=1}^{\infty} \frac{c_q(n)}{q^2} = \frac{12}{\pi^2}.$$

**1.32.** Известно (см. [Copeland and Erdős 1946]), что число

$$0.235711131719\ldots,$$

в котором подряд выписаны все простые числа, является «нормальным по основанию 10», т. е. каждый конечный набор из $k$ последовательных цифр встречается в десятичной записи этого числа со «справедливой» асимптотической частотой $10^{-k}$. Докажите более слабое утверждение: каждый такой набор встречается бесконечно часто.

Кроме того, докажите, что какими бы ни были два набора десятичных цифр, последний из которых заканчивается цифрой 1, 3, 7 или 9, найдется бесконечно много простых чисел, десятичная запись которых начинается первым из этих двух наборов, а заканчивается последним (между ними, конечно, может быть сколько угодно промежуточных цифр).

Относительные плотности множеств простых чисел, заканчивающихся цифрами 1, 3, 7 и 9, равны $1/4$, что следует из соотношения (1.5). Будет ли множество простых чисел с фиксированной *не* последней цифрой обладать аналогичным свойством?

**1.33.** В этом упражнении нам потребуется понятие нормальности числа по данному основанию, введенное в упр. 1.32, а также понятие равномерного распределения, данное в упр. 1.35. Рассмотрим натуральные логарифмы последовательных чисел Ферма как псевдослучайную последовательность действительных чисел. Докажите, что если эта последовательность равномерно распределена по модулю 1, то число $\ln 2$ нормально по основанию 2. Справедливо ли обратное утверждение?

Отметим, что на сегодняшний день остается неизвестным, является ли число $\ln 2$ нормальным хотя бы по какому-то целому основанию. К сожалению, то

же самое можно сказать и про любую другую фундаментальную постоянную, такую, как $\pi$, $e$ и т. д. Таким образом, за исключением случаев искусственно построенных чисел, таких, как в упр. 1.32, доказательства нормальности остаются недостижимыми. Классической книгой, в которой строго рассматривается нормальность и равномерное распределение, является [Kuipers and Niederreiter 1974]. Свойства нормальности некоторых фундаментальных констант, таких, как $\ln 2$, затронуты в работе [Bailey and Crandall 2001].

**1.34.** Примените теорему о простых числах (или просто теорему Чебышёва 1.1.3) и докажите, что множество рациональных чисел вида $p/q$, где числа $p, q$ простые, всюду плотно во множестве положительных вещественных чисел.

**1.35.** Согласно теореме Виноградова, для любого иррационального числа $\alpha$ последовательность $(\alpha p_n)$, где через $p_n$ обозначено $n$-е простое число, равномерно распределена по модулю 1. Равномерность этого распределения означает, что если через $\#(a, b, N)$ обозначить количество попаданий дробных долей первых $N$ элементов этой последовательности в полуинтервал $[a, b) \subset [0, 1)$, то $\#(a, b, N)/N \sim (b - a)$ при $N \to \infty$. Как следствие теоремы Виноградова получите следующий результат. Пусть $\alpha > 1$ — иррациональное число. Рассмотрим множество

$$S(\alpha) = \{\lfloor k\alpha \rfloor : k = 1, 2, 3, \ldots\}.$$

Докажите, что для функции

$$\pi_\alpha(x) = \#\{p \leqslant x \ : \ p \in \mathcal{P} \cap \mathcal{S}(\alpha)\}$$

справедливо асимптотическое соотношение

$$\pi_\alpha(x) \sim \frac{1}{\alpha} \frac{x}{\ln x}.$$

Каким будет поведение функции $\pi_\alpha$ для рациональных чисел $\alpha$?

В продолжение настоящего упражнения заметим, что теорему Виноградова о равномерном распределении можно доказать при помощи рассуждений разд. 1.4.4, связанных с тригонометрическими суммами. Если воспользоваться знаменитой теоремой Вейля о спектральных свойствах равномерно распределенных последовательностей (см. с. 60 и [Kuipers and Niederreiter 1974, Theorem 2.1]), то задача сведется к доказательству того, что для иррационального числа $\alpha$ и любого целого числа $h \neq 0$ сумма

$$E_N(h\alpha) = \sum_{p \leqslant N} e^{2\pi \imath h \alpha p}$$

есть $o(N)$. Это, в свою очередь, можно показать при помощи нахождения надлежащих подходящих дробей к числу $\alpha$ и оценок тригонометрических сумм, при получения которых можно воспользоваться формулой (1.39) для хорошо приближаемых значений $h\alpha$, а для остальных $\alpha$ — оценкой (1.41). По этому вопросу мы рекомендуем и интересный труд [Ellison and Ellison 1985].

Далее, воспользуйтесь тригонометрическими суммами и изучите функцию

$$\pi_c(x) = \#\left\{n \in [1, x] : \lfloor n^c \rfloor \in \mathcal{P}\right\}.$$

Сначала проведите эвристические рассуждения в пользу асимптотической формулы

$$\pi_c(x) \sim \frac{1}{c}\frac{x}{\ln x}.$$

Затем при помощи теоремы о простых числах докажите эту формулу при $c \leqslant 1$. После этого примените метод тригонометрических сумм и получите этот результат и для некоторых $c > 1$. Например, Пятецкий и Шапиро [Graham and Kolesnik 1991] доказали это асимптотическое соотношение для всех $c$ из интервала $1 < c < 12/11$.

**1.36.** Изучение простых чисел приводит к рассмотрению поистине огромных чисел, таких, как числа Скьюза

$$10^{10^{10^{34}}}, \quad 10^{10^{10^{964}}},$$

последнее из которых является установленной верхней границей для наименьшего значения $x$, при котором $\pi(x) > \mathrm{li}_0(x)$, где $\mathrm{li}_0(x) = \int_0^x dt/\ln t$. (Первое число Скьюза есть более ранняя знаменитая оценка, полученная как следствие из гипотезы Римана.) Поскольку подынтегральное выражение имеет особенность в точке $t = 1$, при $x > 1$ указанный интеграл понимается в смысле «главного значения»:

$$\mathrm{li}_0(x) = \lim_{\epsilon \to 0} \left( \int_0^{1-\epsilon} \frac{1}{\ln t}\, dt + \int_{1+\epsilon}^{x} \frac{1}{\ln t}\, dt \right).$$

Справедливо равенство $\mathrm{li}_0(x) = \mathrm{li}\,(x) + c$, где $c \approx 1.0451637801$. Еще до того как Скьюз получил свои оценки, Дж. Литтлвуд показал, что разность $\pi(x) - \mathrm{li}_0(x)$ (а также $\pi(x) - \mathrm{li}\,(x)$) не просто меняет знак, а делает это бесконечно часто.

Любопытной начальной задачей, возникающей при вторжении в «мир Скьюза», является получение десятичной записи *второго* из чисел Скьюза (другими словами, требуется заменить надлежащим образом число $e$ на 10, как это сделано для первого числа Скьюза). Отметим, что свежий подход к этой проблеме описан в статье [Kaczorowski 1984], а современной оценкой наименьшего числа $x$, для которого $\pi(x) > \mathrm{li}_0(x)$, является $x < 1.4 \cdot 10^{316}$ [Bays and Hudson 2000a, 2000b]. Кроме того, последние авторы недавно показали (используя на определенном этапе $10^6$ приближенных значений нулей дзета-функции, предоставленных А. Одлыжко), что $\pi(x) > \mathrm{li}_0(x)$ для некоторого $x \in (1.398201, 1.398244) \cdot 10^{316}$.

Занимательное теоретическое упражнение заключается в приближенной оценке того времени, которое потребуется исследователям для *нахождения и доказательства* явного примера выполнения неравенства $\pi(x) > \mathrm{li}_0(x)$. Кроме того, интересно предположить, насколько далеко смогут продвинуться вычисления значений самой функции $\pi(x)$, скажем, за 30 лет. Алгоритмы, позволяющие подсчитать количество простых чисел, обсуждаются в разд. 3.7 и на сегодняшний день нам доступны результаты, немногим превышающие $\pi\left(10^{21}\right)$ (при этом новые результаты следуют один за другим).

Вот еще одно любопытное направление исследований: постарайтесь представить физические обстоятельства, при которых могут потребоваться такие

огромные числа. Несколько таких примеров описаны в статье [Crandall 1997a]. Несмотря на некоторую абсурдность с физической точки зрения (например, подсчет вероятности лично совершить квантово-механический туннельный переход на Марс и остаться в живых), даже они не приводят к числам, существенно меньшим, чем $A^{-A}$, где $A$ — число Авогадро (моль, приблизительно $6 \cdot 10^{23}$). Придумать статистические обстоятельства, приводящие к простым числам, требующим еще большего «повышения этажности» в показателе, как в числах Скьюза, очень не просто.

Отметим, что по ряду технических причин интегральный логарифм $\mathrm{li}_0$ во многих современных численно-символьных системах удобнее всего вычислять через $\mathrm{Ei}(\ln x)$, где мы прибегли к стандартной функции «экспоненциальный интеграл»

$$\mathrm{Ei}(z) = \int_{-\infty}^{z} t^{-1} e^t \, dt,$$

причем при $t = 0$ интеграл понимается в смысле главного значения. В заключение обращаем внимание читателя на тот факт, что ряд авторов использует обозначение li для той функции, которую мы обозначаем через $\mathrm{li}_0$. Напомним, что отличие заключается в выборе нижнего предела интегрирования (см. равенство (1.3)). Таким образом, для наших функций li и $\mathrm{li}_0$ мы можем записать высказанный совет по вычислению в виде

$$\mathrm{li}\,(x) = \mathrm{li}_0(x) - \mathrm{li}_0(2) = \mathrm{Ei}(\ln x) - \mathrm{Ei}(\ln 2) \approx \mathrm{Ei}(\ln x) - 1.0451637801.$$

**1.37.** Согласно статье [Schoenfeld 1976], из гипотезы Римана при всех $x \geqslant 2657$ следует точное неравенство

$$|\pi(x) - \mathrm{li}_0(x)| < \frac{1}{8\pi}\sqrt{x}\ \ln x,$$

где функция $\mathrm{li}_0(x)$ определена в упр. 1.36. Проверьте при помощи вычислений, что все данные табл. 1.1 согласуются с гипотезой Римана!

С помощью непосредственного вычисления и неравенства

$$\mathrm{li}\,(x) < \mathrm{li}_0(x) < \mathrm{li}\,(x) + 1.05$$

докажите высказанное в тексте утверждение о том, что при условии справедливости гипотезы Римана выполняется неравенство

$$|\pi(x) - \mathrm{li}\,(x)| < \sqrt{x}\ \ln x \quad \text{для всех } x \geqslant 2.01. \tag{1.48}$$

Из рассуждений, проведенных нами касательно формулы (1.25), следует, что (1.48) равносильно гипотезе Римана. Отметим, что поскольку для понимания смысла утверждения (1.48) достаточно знать, что такое простое число, натуральный логарифм и определенный интеграл, его можно рассматривать как формулировку гипотезы Римана, доступную, скажем, тем, кто изучает курс математического анализа.

**1.38.** Пусть функция $\psi(x)$ определена соотношением (1.22). В статье [Schoenfeld 1976] из гипотезы Римана было выведено неравенство

$$|\psi(x) - x| < \frac{1}{8\pi}\sqrt{x}\ \ln^2 x \quad \text{для всех } x \geqslant 73.2.$$

Проведите непосредственные вычисления и покажите, что гипотеза Римана влечет неравенство

$$|\psi(x) - x| < \sqrt{x}\ \ln^2 x \quad \text{для всех } x \geqslant 3.$$

Затем при помощи упр. 1.37 докажите, что гипотеза Римана равносильна элементарному утверждению

$$|L(n) - n| < \sqrt{n}\ \ln^2 n \quad \text{для всех целых } n \geqslant 3, \tag{1.49}$$

где $L(n)$ обозначает натуральный логарифм наименьшего общего кратного чисел $1, 2, \ldots, n$. Если неравенство (1.48) можно назвать версией гипотезы Римана для курса математического анализа, то, наверное, утверждение (1.49) является формулировкой для изучающих начала математического анализа, поскольку оно не содержит более сложных понятий, чем наименьшее общее кратное и натуральный логарифм.

**1.39.** Выведите из гипотетической формы асимптотического закона распределения простых чисел (1.25), что между любыми двумя достаточно большими кубами найдется простое число. Воспользуйтесь неравенством (1.48) и проведите соответствующие вычисления для доказательства того, что (опять таки, если гипотеза Римана верна) существует простое число между *любыми* двумя положительными кубами. Отметим, что существование простого числа между любыми двумя достаточно большими кубами было доказано в 1937 г. Ингамом *безусловно*, а в 1999 г. Чен также безусловно показал справедливость этого утверждения для всех кубов, превосходящих $e^{e^{15}}$.

**1.40.** Докажите соотношение $\sum_{p \leqslant n-2} 1/\ln(n-p) \sim n/\ln^2 n$, где суммирование ведется по простым числам.

**1.41.** При помощи известной теоремы о существовании такого положительного числа $c$, что количество четных чисел, не превосходящих $x$, не представимых в виде суммы двух простых чисел, есть $O(x^{1-c})$, докажите существование бесконечного множества троек простых чисел в арифметической прогрессии. (Иной подход к этому вопросу рассмотрен в упр. 1.42.)

**1.42.** Благодаря теории тригонометрических сумм известно, что

$$\sum_{n \leqslant x} \left(R_2(2n) - \mathcal{R}_2(2n)\right)^2 = O\left(\frac{x^3}{\ln^5 x}\right), \tag{1.50}$$

где через $R_2(2n)$, как и в тексте, обозначено количество представлений числа $2n$ в виде $p + q = 2n$, где $p, q$ — простые числа, а функция $\mathcal{R}_2$ определяется равенством (1.10) (см. [Prachar 1978]). Далее, при помощи метода решета Бруна доказано, что

$$R_2(2n) = O\left(\frac{n \ln\ln n}{\ln^2 n}\right).$$

Покажите, что и для функции $\mathcal{R}_2(2n)$ имеет место такая же оценка. Воспользуйтесь этими результатами и докажите, что множество простых чисел $p$, для которых число $2p$ *не* представимо в виде суммы двух различных простых чисел, имеет относительную асимптотическую плотность нуль во множестве всех

простых чисел, т. е. количество таких чисел $p \leqslant x$ есть $o(\pi(x))$. Положим, далее,

$$A_3(x) = \#\left\{(p,q,r) \in \mathcal{P}^3 \ : \ 0 < q-p = r-q; \ q \leqslant x\right\},$$

т. е. $A_3(x)$ равно количеству арифметических прогрессий, составленных из простых чисел $p < q < r$, для которых $q \leqslant x$. Докажите, что при $x \geqslant 2$ справедливо соотношение

$$A_3(x) = \frac{1}{2} \sum_{p \leqslant x, p \in \mathcal{P}} (R_2(2p) - 1) \sim C_2 \frac{x^2}{\ln^3 x},$$

где через $C_2$ обозначена постоянная простых близнецов, определенная равенством (1.6).

Перейдем к вычислительному аспекту этого вопроса. Разработайте эффективный алгоритм, позволяющий находить точное значение функции $A_3(x)$ для данного числа $x$ и проверьте, что $A_3(3000) = 15482$ (т. е. существуют 15482 тройки из различных простых чисел, образующих арифметические прогрессии, среднее из которых не превосходит 3000), $A_3(10^4) = 109700$ и $A_3(10^6) = 297925965$ (последний из этих результатов получен Р. Томпсоном). Здесь можно предложить два варианта проведения вычислений: применение одного из вариантов решета Эратосфена и использование методов, основанных на преобразовании Фурье (см. упр. 1.67). Приведенная выше асимптотическая формула для $A_3$ дает при $x = 10^6$ примерно на 16% меньший результат по сравнению с истинным. Замена дроби $x^2/\ln^3 x$ на двойной интеграл

$$\int_2^x \int_2^{2t-2} \frac{1}{(\ln t)(\ln s)(\ln(2t-s))}\, ds\, dt$$

позволяет сократить этот разрыв до 0.4% (при $x = 10^6$). Объясните полученный эффект.

**1.43.** В статье [Saouter 1998] описано, как можно воспользоваться вычислениями по бинарной проблеме Гольдбаха для всех четных чисел вплоть до $4 \cdot 10^{11}$ при решении тернарной проблемы Гольдбаха для нечетных чисел от 7 до $10^{20} - 1$. Как мы знаем, бинарная проблема Гольдбаха решена для всех четных чисел вплоть до $4 \cdot 10^{14}$. Разработайте алгоритм, следуя которому можно будет расширить результат Саутера, скажем, до $10^{23}$.

Кстати говоря, исследования, связанные с гипотезой Гольдбаха, могут принести ощутимое вознаграждение. В связи с выходом романа А. Доксиадиса *Uncle Petros and Goldbach's Conjecture*, в 2000 г. издатель учредил приз в \$1,000,000 за решение бинарной проблемы Гольдбаха, но «время выдачи» приза истекло в 2002 г.

**1.44.** Здесь мы предлагаем доказать или, по меньшей мере, завершить доказательство результата Шнирельмана (см. разд. 1.2.3) о том, что множество $S = \{p+q \ : \ p,q \in \mathcal{P}\}$ имеет «положительную нижнюю плотность» (терминология будет объяснена ниже). Как и в тексте, обозначим через $R_2(n)$ количество представлений числа $n$ в виде $n = p + q$, где $p, q$ — простые числа.

(1) Выведите из теоремы Чебышёва 1.1.3, что при всех достаточно больших

значениях $x$ справедливо неравенство

$$\sum_{n\leqslant x} R_2(n) > A_1 \frac{x^2}{\ln^2 x},$$

где $A_1$ — некоторая положительная постоянная.

(2) Будем считать известным неравенство

$$\sum_{n\leqslant x} R_2(n)^2 < A_2 \frac{x^3}{\ln^4 x},$$

при $x > 1$, где $A_2$ — некоторая постоянная (здесь мы избегаем огромного объема тяжелой работы!). Этот результат доказывается такими далеко не тривиальными средствами, как методы решета Сельберга и Бруна (см. [Nathanson 1996]).

(3) Из пунктов (1) и (2), а также неравенства Коши—Буняковского

$$\left(\sum_{n=1}^{x} a_n b_n\right)^2 \leqslant \left(\sum_{n=1}^{x} a_n^2\right)\left(\sum_{n=1}^{x} b_n^2\right)$$

(которое имеет место для произвольных действительных чисел $a_n$, $b_n$) выведите, что существует такая положительная постоянная $A_3$, для которой при всех достаточно больших значений переменной $x$ справедливо неравенство

$$\#\{n \leqslant x \ : \ R_2(n) > 0\} > A_3 x.$$

Именно этот результат и скрывается за словосочетанием «положительная нижняя плотность» множества $S$. (Указание: положите $a_n = R_2(n)$, а в качестве $(b_n)$ выберите подходящую *двоичную* последовательность.)

Как сказано в тексте, Шнирельман показал, что эта нижняя оценка плотности влечет его знаменитый результат: существует такое натуральное число $s$, что каждое натуральное число, начиная с 2, представимо в виде суммы не более, чем $s$ простых чисел. Удивительно, что *верхняя* оценка числа гольдбаховых представлений (см. пункт (2)) является ключом ко всему этому рассуждению! Разумеется, дело здесь в том, что согласно этой верхней оценке число таких представлений держится «под контролем», т. е. ведет себя так, что достаточная часть четных чисел $n$ допускает нужное представление (см. упр. 9.80, в котором содержится дальнейшее применение этого метода.)

**1.45.** Выведите из гипотезы 1.2.1 о $k$-кортежах простых чисел, что для каждого числа $k$ существует конечная арифметическая прогрессия, состоящая из $k$ *последовательных* простых чисел.

**1.46.** Обратите внимание на то, что каждое из простых чисел Мерсенна $2^2 - 1$, $2^3 - 1$, $2^5 - 1$ входит в некоторую пару простых близнецов. Воспользуйтесь табл. 1.2 и исследуйте остальные известные простые числа Мерсенна на обладание этим свойством.

**1.47.** Пусть $q$ — простое число Софи Жермен, т. е. число $s = 2q + 1$ также является простым. Докажите, что если $q \equiv 3 \pmod 4$ и $q > 3$, то число Мерсенна

$M_q = 2^q - 1$ является составным, и, в частности, делится на $s$. Керхнер и Галло нашли огромное простое число Софи Жермен

$$q = 18458709 \cdot 2^{32611} - 1,$$

так что возникающее число Мерсенна $M_q$ имеет поистине гигантские размеры (примерно $10^{10^4}$ цифр) и, как мы знаем, это число составное.

**1.48.** Докажите, что для чисел Мерсенна справедливо равенство

$$\text{НОД}\,(2^a - 1, 2^b - 1) = 2^{\text{НОД}\,(a,b)} - 1.$$

Выведите отсюда, что для различных простых чисел $q$ и $r$ числа Мерсенна $M_q$ и $M_r$ взаимно просты.

**1.49.** Отталкиваясь от неравенства Келлера

$$p > 6 \cdot 10^{19}$$

для наименьшего простого делителя $p$ числа $F_{24}$, оцените при помощи соотношения (1.13) априорную вероятность того, что число $F_{24}$ является простым (теперь мы знаем, что это число *не* простое, но будет небесполезным провести работу вне зависимости от этого результата). На основе данных о простых делителях произвольных чисел Ферма оцените вероятность того, что после $F_4$ простых чисел Ферма *нет* (здесь мы предлагаем воспользоваться специальным видом возможных делителей, а также известными свойствами найденных чисел Ферма).

**1.50.** Выведите теорему 1.2.1 как следствие оценки Бруна (1.8).

**1.51.** При помощи оценки Виноградова для $R_3(n)$ приближенно найдите количество представлений нечетного числа $n = 3 \cdot 5 \ldots 101$ (произведение берется по нечетным простым числам) в виде суммы трех простых чисел (см. упр. 1.68).

**1.52.** Непосредственным вычислением покажите, что $10^8$ *не* представимо в виде суммы двух псевдопростых по основанию 2 (определение см. в разд. 3.4). Попутно покажите, что, тем не менее, если через $p$ обозначить простое число, а через $P_2$ — нечетное псевдопростое по основанию 2, то существует в точности 120 представлений вида

$$10^8 = p + P_2 \quad \text{или} \quad P_2 + p$$

(это хорошая проверка для соответствующей программы). Кстати говоря, одним из таких представлений является замечательное равенство

$$10^8 = 99999439 + 561,$$

где 561 — знаменитое наименьшее число Кармайкла (см. разд. 3.4.2). А можно ли представить $10^8$ в виде суммы двух псевдопростых чисел по некоторому основанию, отличному от 2? Каково математическое ожидание числа различных «псевдопредставлений» вида $p + P_b$ для данного числа $n$?

**1.53.** Докажите, что если в двоичной записи простого числа $p$ все единицы занимают места с номерами, образующими арифметическую прогрессию, то

это число $p$ не является простым Вифериха. Выведите отсюда, что ни простые числа Мерсенна, ни простые числа Ферма не могут быть одновременно и простыми числами Вифериха.

**1.54.** Обозначим через $u^{-1}$ вычет, обратный по умножению к $u$ по модулю $p$. Покажите, что для любого нечетного простого числа $p$ справедливо сравнение

$$\sum_{p/2<u<p} u^{-1} \equiv \frac{2^p - 2}{p} \pmod{p}.$$

**1.55.** При помощи теоремы Вильсона—Лагранжа 1.3.6 докажите, что для всякого простого числа $p \equiv 1 \pmod 4$ разрешимо сравнение $x^2 + 1 \equiv 0 \pmod p$.

**1.56.** Докажите следующий результат, связанный с простыми числами Вильсона: если $g$ — первообразный корень по модулю простого числа $p$, то частное Вильсона

$$w_p \equiv \sum_{j=1}^{p-1} \left\lfloor \frac{g^j}{p} \right\rfloor g^{p-1-j} \pmod{p}.$$

Далее воспользуйтесь этим фактом и составьте алгоритм, определяющий, является ли простое число $p$ с первообразным корнем $g = 2$ простым Вильсона. При этом пользоваться умножением запрещается (разрешены сложение, вычитание и сравнение).

**1.57.** Между простыми близнецами и теоремой Вильсона—Лагранжа существует следующая связь. Пусть $p > 1$ — целое число. Докажите теорему Клемента о том, что числа $p$ и $p + 2$ образуют пару простых близнецов тогда и только тогда, когда справедливо сравнение

$$4(p-1)! \equiv -4 - p \pmod{p(p+2)}.$$

**1.58.** Попробуйте разрешить следующий «парадокс Мерсенна». Пусть $x$ — большое натуральное число. Рассмотрим «вероятность» того, что это число $x$ окажется простым. Как мы знаем, для проверки простоты достаточно исследовать делимость числа $x$ на все простые числа, не превосходящие $\sqrt{x}$. Но может показаться, что из теоремы 1.4.2 вытекает, что когда отсеены все простые числа, меньшие, чем $\sqrt{x}$, процесс заканчивается с итоговой вероятностью

$$\prod_{p \leqslant \sqrt{x}} \left(1 - \frac{1}{p}\right) \sim \frac{2e^{-\gamma}}{\ln x}.$$

Придите к этому же результату, просто убрав целую часть в соотношении (1.46). Однако асимптотический закон распределения простых чисел утверждает, что истинная асимптотическая вероятность того, что $x$ — простое число, равна $1/\ln x$. Отметим, что $2e^{-\gamma} = 1.1229189\ldots$, так что мы сталкиваемся с парадоксом!

Мы уже говорили, что решето Эратосфена «более эффективно», чем случайное тестирование, и это один из способов понимания рассматриваемого «парадокса». Кроме того, существуют и иные пути к его решению. Например, в

интересной статье [Furry 1942] проведен анализ воздействия решета Эратосфена на заданный промежуток $[x, x+d]$ и выявлен ряд удивительных моментов, связанных с количеством вычеркиваемых из него составных чисел (исторический обзор по этой проблематике содержится в книге [Bach and Shallit 1996, p. 365]).

**1.59.** Предположив справедливость соотношения (1.24) на всей области сходимости интеграла, докажите, что оценка $M(x) = O(x^{1/2+\varepsilon})$ влечет гипотезу Римана.

**1.60.** Рассмотрим наглядный пример, иллюстрирующий взаимосвязь асимптотического закона распределения простых чисел с поведением дзета-функции Римана. Воспользуйтесь равенством

$$-\frac{\zeta'(s)}{\zeta(s)} = s\int_1^\infty \psi(x)x^{-s-1}\,dx$$

и покажите, что из оценки

$$\psi(x) = x + O(x^\alpha)$$

следует отсутствие нулей $\zeta(s)$ в полуплоскости $\mathrm{Re}(s) > \alpha$. Отсюда видна связь между остаточным членом в формулировке асимптотического закона распределения простых чисел и распределением нулей функции $\zeta$.

Перейдем к более трудному примеру. Предположим, что $\zeta$ не имеет нулей в полуплоскости $\mathrm{Re}(s) > \alpha$. При помощи соотношения (1.23) докажите оценку

$$\sum_{\mathrm{Im}(\rho) \leqslant T} \frac{x^\rho}{|\rho|} = O(x^\alpha \ln^2 T)$$

(этот результат нетривиален и представляет самостоятельный интерес [Davenport 1980]). Наконец, выведите, что

$$\psi(x) = x + O\left(x^{\alpha+\varepsilon}\right)$$

для любого $\varepsilon > 0$. Рассмотренный пример объясняет, почему предположенное Риманом равенство

$$\pi(x) = \mathrm{li}\,(x) + O(x^{1/2}\ln x)$$

иногда рассматривают как запись гипотезы Римана через асимптотический закон распределения простых чисел.

**1.61.** В этом упражнении мы научимся вычислять значения дзета-функции Римана на критической прямой и применим полученную формулу для проверки приведенных ниже значений высокой точности. Начнем с того, что по возможности кратко рассмотрим знаменитую формулу Римана—Зигеля. Эта на первый взгляд неуклюжая формула оказывается настолько мощной, что связанные с ней трудности становятся лишь небольшой ценой, которую приходится заплатить за ее применение. Отметим, что именно эта формула позволила установить тот факт, что первые 1.5 миллиарда нулей дзета-функции (с положительной мнимой частью) лежат в точности на критической прямой (параллельные варианты позволили также продвинуться гораздо дальше: см. текст и упр. 1.62).

Первый шаг состоит в том, что мы рассмотрим функцию Харди

$$Z(t) = e^{i\vartheta(t)}\zeta(1/2 + \imath t),$$

где

$$\vartheta(t) = \operatorname{Im}\left(\ln\Gamma\left(\frac{1}{4} + \frac{it}{2}\right)\right) - \frac{1}{2}t\ln\pi,$$

получив *действительнозначную* функцию $Z$ на критической прямой (переменная $t$ принимает действительные значения). Более того, существует соответствие между переменами знака функции $Z$ и нулями функции $\zeta$, а именно, если при $a < b$ числа $Z(a)$ и $Z(b)$ имеют разные знаки, то на интервале $(1/2 + ia, 1/2 + ib)$ лежит по меньшей мере один нуль дзета-функции. Кроме того, очень удобно равенство

$$|Z(t)| = |\zeta(1/2 + it)|.$$

Отметим, что исследователь может работать как исключительно с действительной функцией $Z$ (например, при численном изучении гипотезы Римана), так и с самой $\zeta$ на критической прямой при помощи соответствующих вычислений гамма-функции и т. д.

Теперь мы готовы перейти к рассмотрению наиболее эффективной с вычислительной точки зрения формулы Римана—Зигеля [Brent 1979]. Положим $\tau = t/(2\pi)$, $m = \lfloor\sqrt{\tau}\rfloor$, $z = 2(\sqrt{\tau} - m) - 1$. Тогда

$$\begin{aligned} Z(t) &= 2\sum_{n=1}^{m} n^{-1/2}\cos(t\ln n - \vartheta(t)) \\ &\quad + (-1)^{m+1}\tau^{-1/4}\sum_{j=0}^{M}(-1)^j\tau^{-j/2}\Phi_j(z) + R_M(t). \end{aligned}$$

Здесь $M$ — целочисленный параметр, целые функции $\Phi_j$ при $j \geqslant 0$ выражаются через функцию $\Phi_0$ и ее производные, а $R_M(t)$ обозначает остаточный член. На практике часто выбирают $M = 2$, а следовательно, нам потребуются функции

$$\Phi_0(z) = \frac{\cos(\frac{1}{2}\pi z^2 + \frac{3}{8}\pi)}{\cos(\pi z)},$$

$$\Phi_1(z) = \frac{1}{12\pi^2}\Phi_0^{(3)}(z),$$

$$\Phi_2(z) = \frac{1}{16\pi^2}\Phi_0^{(2)}(z) + \frac{1}{288\pi^4}\Phi_0^{(6)}(z).$$

Несмотря на всю ее сложность, необходимо сказать, что формула для $Z(t)$ идеально подходит при проведении реальных вычислений. В частности, для остатка $R_2$ справедлива оценка

$$|R_2(t)| < 0.011t^{-7/4} \quad \text{для всех} \quad t > 200.$$

Известны (в основном из диссертации [Gabcke 1979]) и оценки остатков более высокого порядка ($M > 2$), но именно $R_2$ использовался для вычислений на протяжении двух десятков лет.

Примените формулу Римана—Зигеля при $M = 2$ и проверьте следующие данные:

$$\begin{aligned}\zeta(1/2 + 300\imath) \approx \quad & 0.47745567187848255453606 19 \\ & + 0.60790213327955307265907 49\,\imath, \\ Z(1/2 + 300\imath) \approx \quad & 0.77298701299230422726245 25,\end{aligned}$$

которые являются точными до последнего знака. Далее, вычислите ближайший нуль к точке $1/2 + 300\imath$ (должно получиться $t \approx 299.84035$). Проверьте также, что (снова на уровне аппроксимации $M = 2$) вычисление значения

$$\zeta(1/2 + 10^6\imath) \approx 0.0760890697382 + 2.805102101019\,\imath$$

с соответствующей точностью займет очень мало машинного времени.

Компьютерная реализация формулы Римана—Зигеля открывает дверь в прекрасный мир вычислений, способствующих развитию аналитической теории чисел. Различные пути применения вычислений значений функции $\zeta$ в комплексной области описаны в работах [Brent 1979], [van de Lune et al. 1986], [Odlyzko 1994], [Borwein et al. 2000]. Необходимо отметить, что несмотря на всю мощь формулы Римана—Зигеля, существуют и другие средства для эффективных вычислений значений функции $\zeta$ при больших значениях мнимой части ее аргумента. В частности, можно избежать присущую ряду Римана—Зигеля асимптотичность благодаря хорошо сходящимся представлениям, основанным на неполных значениях гамма-функции или на седловых точках некоторых интегралов. По этому вопросу мы рекомендуем работы [Galway 2000], [Borwein et al. 2000] и [Crandall 1999c].

**1.62.** Для проведения вычислений, основанных на формуле Римана—Зигеля (см. упр. 1.61) и иных аналогичных соотношениях, в случае когда $s = \sigma + \imath t$ не лежит на критической прямой, очевидно, можно воспользоваться суммами вида

$$S_m(s) = \sum_{n=1}^{m} \frac{1}{n^s},$$

где параметр $m$ выбран подходящим образом (обычно $m \sim \sqrt{t}$). Исследуйте вопрос о вычислении $S_m(s)$ для значений $s$, составляющих *арифметическую прогрессию*, воспользовавшись неоднородным алгоритмом быстрого преобразования Фурье (алгоритм 9.5.8). А именно, для

$$s = \sigma + \imath k\tau$$

при, скажем, $k = 0, \ldots, K - 1$ имеем

$$S_m(\sigma + \imath k\tau) = \sum_{n=1}^{m} \frac{1}{n^\sigma} e^{-\imath k\tau \ln n},$$

что, без сомнения, подсказывает стратегию применить $(m/K)$ неоднородных БПФ, каждое из которых имеет длину $K$. Заметим, что этот подход позволяет вычислить $S_m$ для всех $k \in [0, K - 1]$ суммарно за

$$O(m \ln K)$$

операций, причем требуемая точность входит (лишь логарифмически) в скрывающуюся за символом $O$ постоянную. Это является существенным достижением по сравнению с бесхитростным подходом — вычислением $K$ раз сумм длины $m$, что требует $O(mK)$ операций.

Такие ускорения применяются не только для проверки гипотезы Римана, но и наряду с аналитическими методами для подсчета количества простых чисел. Отметим, что этот подход с неоднородным БПФ, по существу, эквивалентен по сложности параллельному методу статьи [Odlyzko and Schönhage 1988]. Однако специалистам, знакомым с БПФ, или располагающим эффективными БПФ-программами (неоднородное БПФ может вызывать их как подпрограммы), метод настоящего упражнения должен показаться привлекательным.

**1.63.** Покажите, что функция $\psi(x)$, определенная равенством (1.22), совпадает с логарифмом наименьшего общего кратного всех натуральных чисел, не превосходящих $x$. Докажите, что теорема о простых числах эквивалентна соотношению $\psi(x) \sim x$. Кстати говоря, в статье [Deléglise and Rivat 1998] показано, что $\psi\left(10^{15}\right) = 999\,999\,997\,476\,930.507683\ldots$. Это замечательная иллюстрация соотношения $\psi(x) \sim x$. Кроме того, отсюда видно, что при $x = 10^{15}$ остаток $|\psi(x) - x|$ примерно равен $\sqrt{x}$, а именно такого порядка остаток и должен быть согласно гипотезе Римана.

**1.64.** Проведите вычисления, связывающие распределение простых чисел с нулями дзета-функции Римана на критической прямой («критическими нулями») посредством функции $\psi$, определенной равенством (1.22). Отталкиваясь от точного классического соотношения (1.23), составьте таблицу численных значений первых $2K$ критических нулей ($K$ из которых имеют положительную мнимую часть) и численно исследуйте полученное приближение для $\psi(x)$, скажем, при нецелых $x \in (2, 1000)$. Для проверки правильности Ваших вычислений воспользуйтесь тем, что если обозначить через $\psi^{(K)}$ приближение, полученное на основе $2K$ нулей, то для $K = 200$ имеет место поразительное неравенство

$$\left|\psi(x) - \psi^{(200)}(x)\right| < 5$$

для описанных значений $x$. Эвристически говоря, это означает, что первые 200 критических нулей и сопряженные к ним определяют положение простых чисел на интервале $(2, 1000)$ с точностью «счета на пальцах одной руки». Более того, при этом приближении остаток колеблется около нуля так, что это приближение оказывается очень удачным в смысле среднего значения. Попробуйте ответить на следующий вопрос. Пусть задан некоторый интервал изменения переменной $x$. Каким должно быть количество задействованных критических нулей для того, чтобы на этом интервале выполнялось неравенство $|\psi - \psi^{(K)}| < 1$? Возникает еще один вычислительный вопрос: насколько численно хорошим является приближение (основанное на гипотезе Римана)

$$\psi(x) = x + 2\sqrt{x} \sum_t \frac{\sin(t \ln x)}{t} + O\left(\sqrt{x}\right),$$

где $t$ пробегает мнимые части критических нулей (см. [Ellison and Ellison 1985])?

Аналогичный аналитический подход к реальному подсчету простых чисел описан в разд. 3.7 и упр. 3.50.

**1.65.** Здесь так же, как и в упр. 1.64, нам потребуются данные о критических нулях дзета-функции Римана. Существуют несколько полезных тестов для любой вычислительной схемы, связанной с критической прямой. Вот один из них. Следствием гипотезы Римана является точное равенство (см. [Bach and Shallit 1996, p. 214])

$$\sum_{\rho} \frac{1}{|\rho|^2} = 2 + \gamma - \ln(4\pi),$$

где $\rho$ пробегает все нули на критической прямой. Проведите численную проверку этого соотношения с наибольшей возможной точностью. Для этого выберите достаточно большое число $T$ и сделайте два шага.

(1) Рассчитайте эту сумму по всем $\rho = 1/2 + it$ при $|t| \leqslant T$.
(2) Получите *оценку* остатка ряда при помощи известных формул о приближенном распределении нулей (см. [Edwards 1974], [Titchmarsh 1986], [Ivić 1985]).

Укажем, что непосредственное вычисление значений дзета-функции рассмотрено в упр. 1.61, а в упр. 8.34 затронуты и другие вычислительные вопросы, связанные с гипотезой Римана.

**1.66.** Простые тригонометрические суммы допускают занимательный анализ. Достаточно часто их оценки (преимущественно верхние) находят интересное применение. Пусть $p$ — нечетное простое число и $a, b, c$ — целые числа. Положим

$$S(a,b,c) = \sum_{x=0}^{p-1} e^{2\pi i(ax^2+bx+c)/p}.$$

При помощи равенства Вейля (1.34) докажите, что

$$|S(a,b,c)| = 0, \ p \ \text{ или } \ \sqrt{p},$$

и покажите, как по данным значениям $a, b, c$ определить, какое именно из трех значений принимает эта сумма. Далее, обобщите полученный результат на тот случай, когда в сумме $S$ вместо простого числа $p$ стоит составное число $N$. Если немного потрудиться, то можно даже справиться и с тем случаем, когда числа $a$ и $N$ не являются взаимно простыми. Отметим, что мы только что описали некоторый подход к вычислению гауссовых сумм (см. упр. 2.27, 2.28).

Далее, примените тот же подход для следующей кубической тригонометрической суммы (где простое число $p$ и целое число $a$ произвольны)

$$T(a) = \sum_{x=0}^{p-1} e^{2\pi i a x^3/p}.$$

Очевидно, что $0 \leqslant |T(a)| \leqslant p$. Опишите, при каких значениях $p, a$ достигается каждое из равенств (нулю и $p$). Затем докажите, что при всех $a \not\equiv 0 \pmod p$

справедливо неравенство

$$|T(a)| < \sqrt{p^{3/2}+p} < 2p^{3/4}.$$

Применив более сильные средства, чем соотношение (1.34), этот результат $O(p^{3/4})$ можно усилить до наилучшего из известных $O\left(p^{1/2}\right)$ [Korobov 1992, Theorem 5], [Vaughan 1997, Lemma 4.3]. Тем не менее, даже оценка с показателем $3/4$ приводит к некоторым интересным результатам. Более того, даже соотношение $T(a) = o(p)$ при $p \to \infty$ влечет асимптотическое равномерное распределение кубов по модулю $p$ (см. упр. 1.35). Отметим также, что при помощи верхних оценок тригонометрических сумм можно получать оценки *снизу* некоторых других сумм. По затронутым вопросам см. упр. 9.41 и 9.80.

**1.67.** Равенство (1.36) является одним из обширного множества интегральных соотношений для числа любопытных представлений, связанных с простыми числами. Напомним, что мы ввели обозначение

$$E_N(t) = \sum_{p \leqslant N} e^{2\pi i t p}.$$

Докажите следующие высказывания об эквивалентности.

(1) Бесконечность множества пар простых близнецов равносильна расходимости при $N \to \infty$ интеграла

$$\int_0^1 e^{4\pi i t} E_N(t) E_N(-t)\, dt.$$

(2) Бесконечность множества троек простых чисел, образующих арифметическую прогрессию (см. упр. 1.41, 1.42), равносильна расходимости при $N \to \infty$ интеграла

$$\int_0^1 E_N^2(t) E_N(-2t)\, dt.$$

(3) Бинарная проблема Гольдбаха эквивалентна соотношению

$$\int_0^1 e^{-2\pi i t N} E_N^2(t)\, dt \neq 0$$

для четных $N > 2$, а тернарная проблема Гольдбаха эквивалентна соотношению

$$\int_0^1 e^{-2\pi i t N} E_N^3(t)\, dt \neq 0$$

для нечетных $N > 5$.

(4) Бесконечность множества простых чисел Софи Жермен (т. е. таких простых чисел $p$, для которых число $2p+1$ также простое) равносильна расходимости при $N \to \infty$ интеграла

$$\int_0^1 e^{2\pi i t} E_N(2t) E_N(-t)\, dt.$$

**1.68.** В разд. 1.4.4 мы отмечали, что существует связь между тригонометрическими суммами и сингулярным рядом $\Theta$, возникшим в решении Виноградова тернарной проблемы Гольдбаха (см. (1.12)). Докажите, что произведение Эйлера для $\Theta(n)$ сходится (а что будет, если $n$ четно?) и совпадает с абсолютно сходящимся рядом

$$\Theta(n) = \sum_{q=1}^{\infty} \frac{\mu(q)}{\varphi^3(q)} c_q(n),$$

где сумма Рамануджана $c_q$ определена равенством (1.37). Полезно заметить, что функции $\mu$, $\varphi$ и $c$ мультипликативны, причем последняя в том смысле, что если НОД$(a,b) = 1$, то $c_a(n)c_b(n) = c_{ab}(n)$. Покажите также, что при достаточно больших значениях $B$ замена ряда (1.40) на его частичную сумму до $Q = \ln^B n$ вносит пренебрежимо малый вклад в общую оценку для тернарной проблемы Гольдбаха.

Пусть, далее, $R_s(n)$ обозначает количество представлений натурального числа $n$ в виде суммы $s$ простых чисел. Известно, что при $s > 2$ и $n \equiv s \pmod 2$ справедливо равенство

$$R_s(n) = \frac{\Theta_s(n)}{(s-1)!} \frac{n^{s-1}}{\ln^s n} \left(1 + O\left(\frac{\ln\ln n}{\ln n}\right)\right),$$

где общий сингулярный ряд согласно теории тригонометрических сумм равен

$$\Theta_s(n) = \sum_{q=1}^{\infty} \frac{\mu^s(q)}{\varphi^s(q)} c_q(n).$$

Представьте этот сингулярный ряд в виде произведения Эйлера (при $s = 3$ результат должен совпасть с формулой, полученной в тексте). Проверьте, что существуют такие положительные постоянные $C_1, C_2$, что при всех $s > 2$ и $n \equiv s \pmod 2$ выполнено двойное неравенство

$$C_1 < \Theta_s(n) < C_2.$$

А Вы смогли бы получить (предполагаемый, неокончательный) особый ряд формулы (1.9) для случая $s = 2$? Конечно, на протяжении столетий трудность в этом вопросе представляет не получение особого ряда, а оценка остаточного члена. Анализ таких остаточных членов привлекал внимание исследователей на протяжении большей части XX века, причем новые оценки получались, можно сказать, одна за другой. В частности, статья [Languasco 2000] содержит исторический обзор последовательных результатов, связанных с точными оценками сверху остатка для всех $s \geqslant 3$, полученных при условии справедливости гипотезы Римана.

Перейдем к вычислительной стороне вопроса. Получите хорошее численное значение сингулярного ряда из (1.12), скажем для $n = 10^8 - 1$, и сравните истинное количество представлений $R_3(n)$ с оценкой Виноградова (1.12). Можно ли повысить точность приближения, заменив $n^2/\ln^3 n$ на интеграл? (Обратите внимание на рассуждение в тексте, касающееся точного значения $R_2(10^8)$.)

**1.69.** Положим

$$S = \{n\lfloor \ln n \rfloor : \ n = 1, 2, 3 \ldots\}.$$

Докажите, что каждое достаточно большое натуральное число принадлежит множеству $S + S$, т. е. представимо в виде суммы двух элементов множества $S$. (Доказательство может быть как комбинаторным, опирающимся на китайскую теорему об остатках (см. разд. 2.1.3), так и основанным на методах свертки, обсуждаемых далее в этой книге.) Верно ли, что все натуральные числа, начиная с 221, лежат во множестве $S + S$? Положим, далее,

$$T = \{\lfloor n \ln n \rfloor : \ n = 1, 2, 3, \ldots\}.$$

Верно ли, что все натуральные числа, превосходящие 25, принадлежат множеству $T + T$?

Поскольку $n$-е простое число асимптотически эквивалентно $n \ln n$, эти результаты показывают, что проблеме Гольдбаха не угрожает относительная редкость простых чисел. В связи с этим возникает ряд интересных вопросов. Например, можно ли предъявить такое множество $U$, состоящее из натуральных чисел, что его $n$-й элемент $U$ асимптотически эквивалентен $n \ln n$, но, тем не менее, множество $U + U$ имеет асимптотическую плотность 0?

**1.70.** В этом упражнении собраны теоретические и вычислительные задания, связанные с тригонометрическими суммами, а именно, с рассмотренной в тексте суммой $E_N$, для которой мы обсуждали оценку

$$E_N(a/q) = \sum_{p \leqslant N} e^{2\pi i p a/q} \ \approx \ \frac{\mu(q/g)}{\varphi(q/g)} \pi(N),$$

где $g = \text{НОД}(a, q)$. Напомним, что это приближение является наиболее удачным главным образом тогда, когда $g = 1$ и знаменатель $q$ мал. Начнем с теоретических заданий.

(1) Покажите, что при $q = 2$ и $a = 0$ или 1 наша оценка суммы $E_N$ тривиальна.

(2) При помощи векторов на комплексной плоскости проиллюстрируйте, как происходит приближение в случае $q = 3$ и $a = 1$ или 2.

(3) Заметим, что при $q = 4$ для некоторых значений $a$ правая часть рассматриваемой оценки равна нулю. Примените оценку остатка (такую, как условный результат (1.32)) и получите для таких случаев точные ненулевые оценки суммы $E_N(a/4)$ при $a = 1$ или 3.

Приведенные теоретические задачи раскрывают основы поведения тригонометрических сумм при малых значениях $q$.

Перейдем к вычислительным аспектам исследования суммы $E_N$. Мы предлагаем такую последовательность действий. Положим $N = 10^5$, $q = 31$.

(1) Непосредственным суммированием по простым числам $p \leqslant N$ создайте таблицу значений суммы $E_N$ при $a \in [0, q-1]$ (в таблице будет $q$ комплексных чисел).

(2) Создайте еще одну таблицу, состоящую из значений $\pi(N)\frac{\mu(q/g)}{\varphi(q/g)}$ также для всех $a \in [0, q-1]$.
(3) Сравните (визуально) полученные таблицы. Несмотря на больший разброс в первой таблице, должно наблюдаться очень хорошее согласование в среднем. Насколько согласуется разница данных в таблицах с теорией?
(4) Объясните гладкость данных второй таблицы (за исключением $(a=0)$-го элемента). Наконец, покажите, как можно построить первую таблицу при помощи быстрого преобразования Фурье (БПФ) двоичного сигнала (т. е. потока нулей и единиц).

Кроме того, полезно провести численное интегрирование для проверки (при малых значениях $N$) некоторых гипотетических эквивалентностей из упр. 1.67.

**1.71.** Проверьте, что существуют ровно 35084 числа, меньших $10^{100}$, которые являются 4-гладкими. Докажите существование такой постоянной $c$, что для функции $\psi(x,4)$, равной количеству 4-гладких чисел, не превосходящих $x$, справедливо асимптотическое соотношение

$$\psi(x,4) \sim c\ln^2 x.$$

Получите численное значение этой постоянной $c$, а также постарайтесь наиболее точно оценить остаток. Обобщите проведенные рассуждения и покажите, что для каждого $y \geqslant 2$ существует такая положительная постоянная $c_y$, что

$$\psi(x,y) \sim c_y \ln^{\pi(y)} x, \text{ где число } y \text{ фиксировано и } x \to \infty.$$

**1.72.** При помощи численных экспериментов подтвердите высказанное после равенства (1.45) утверждение о том, когда неявная оценка снизу «приемлема».

**1.73.** Воспользуйтесь ЭВМ для расчета примерной вероятности того, что все простые делители случайного целого числа, состоящего из 100 цифр, окажутся меньше, чем $10^{10}$ (можно воспользоваться методом работы [Bernstein 1998]). Отметим, что для $\rho$ из теоремы 1.4.9 должно получиться $\rho(10) \approx 2.77 \times 10^{-11}$.

**1.74.** Какова приблизительная вероятность того, что все, кроме одного, простые делители случайного целого числа (заданного порядка, скажем, $x$) не превосходят $B$, а оставшееся простое число лежит в промежутке $(B, C]$? Эта проблема возникает при изучении методов разложения на множители, включающих «вторую стадию», на которой, после того как на первой стадии достигнута (в определенном, обусловленном алгоритмом смысле) первая граница $B$, происходит поиск оставшегося простого числа на промежутке $(B, C]$. Для большинства реализаций различных методов разложения на множители характерен выбор числа $C$ существенно больший, чем $B$, поскольку обычно операции на второй стадии существенно «дешевле». По этому вопросу см. упр. 3.5.

**1.75.** Здесь мы затронем вопрос, приводящий к любопытным вычислительным задачам. Рассмотрим число

$$c = 1/3 + \cfrac{1}{1/5 + \cfrac{1}{1/7 + \ldots}},$$

где элементы так называемой непрерывной (или цепной) дроби являются обратными ко всем последовательным нечетным простым числам. Нетрудно показать, что число $c$ определено корректно. (Вообще, непрерывная дробь,

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \dots}}$$

сходится, если сумма ее элементов $a_0 + a_1 + a_2 + \dots$ расходится.) Сначала получите приближенное численное значение постоянной $c$. Затем проведите численное (но строгое) доказательство того, что $c \neq 1$. Далее, исследуйте следующий тонкий вопрос: почему если в числе $c$ использовать все простые числа, т. е. начать с двойки, то получится примерно тот же численный результат? Докажите, что если бы эти две непрерывные дроби оказались равны, то было бы справедливо равенство $c = \left(1 + \sqrt{17}\right)/4$. При помощи более тонких численных экспериментов постарайтесь разрешить вопрос, верно это равенство или нет.

**1.76.** Как следует из занимательной теоремы Бредихина [Bredihin 1963], если число $n$ равно степени двойки, то для функции

$$N(n) = \#\{(x,y,p) \ : \ n = p + xy;\ p \in \mathcal{P};\ x,y \in \mathbf{Z}^+\}$$

справедливо асимптотическое соотношение

$$\frac{N(n)}{n} \sim \frac{105}{2\pi^4}\zeta(3) \approx 0.648\dots.$$

Здесь можно предложить такие вычислительные задания. Во-первых, проверьте это асимптотическое соотношение путем непосредственного подсчета числа решений. Во-вторых, снимите ограничение, что число $n$ равно степени двойки, и постарайтесь (как экспериментально, так и теоретически) показать, что в общем случае постоянная 105 должна быть заменена на произведение

$$315 \prod_{p \in \mathcal{P},\ p \mid n} \frac{(p-1)^2}{p^2 - p + 1}.$$

## 1.6. Проблемы для исследования

**1.77.** В подтверждение теоремы Миллса (первая часть теоремы 1.2.2) постарайтесь найти такое явное значение $\theta$, чтобы число $\left\lfloor \theta^{3^j} \right\rfloor$ было простым при всех $j = 1, 2, \dots, n$ для достаточно большого числа $n$. Рассмотрите пример рационального выбора $\theta = 165/92$, при котором числа

$$\left\lfloor \theta^{3^1} \right\rfloor, \left\lfloor \theta^{3^2} \right\rfloor, \left\lfloor \theta^{3^3} \right\rfloor, \left\lfloor \theta^{3^4} \right\rfloor$$

оказываются простыми, но уже число $\left\lfloor \theta^{3^5} \right\rfloor$, к сожалению, является составным. Можете ли Вы найти по возможности более простое рациональное значение $\theta$, при котором длина цепочки соответствующих простых чисел будет больше четырех?

Назовем (конечную или бесконечную) последовательность простых чисел $q_1 < q_2 < \ldots$ «последовательностью Миллса», если существует такое число $\theta$, для которого $q_j = \left\lfloor \theta^{3^j} \right\rfloor$ при $j = 1, 2, \ldots$. Верно ли, что любую конечную последовательность Миллса можно продолжить до бесконечности, быть может, выбрав другое значение $\theta$ (но с сохранением первых членов последовательности)? Если это так, то отсюда вытекает, что для каждого простого числа $p$ существует бесконечная последовательность Миллса, начинающаяся с числа $p$. Быть может, этот более общий вопрос можно разрешить для достаточно больших $q_1$ при помощи оригинального метода статьи [Mills 1947] (см. также книгу [Ellison and Ellison 1985, p. 31])). Конечно, если более сильное утверждение ложно, то можно попробовать найти соответствующий контрпример. В работе [Weisstein 2005] сообщается, что теорема Миллса имеет место для некоторого $\theta$, немного большего, чем 1.3. Строгое доказательство пока не получено, так что перед нами стоит настоящая проблема для исследований.

**1.78.** Существует ли такое действительное число $\theta > 1$, что последовательность $(\lfloor \theta^n \rfloor)$ состоит исключительно из простых чисел? Существование такого числа $\theta$ кажется маловероятным, однако авторам неизвестны результаты в этом направлении. Целые части первых восьми степеней числа $\theta = 1287/545$ равны $2, 5, 13, 31, 73, 173, 409, 967$ (это простые числа). Найдите более длинную цепь. Если бы такая бесконечная последовательность существовала, то это означало бы существование бесконечного множества троек простых чисел $p, q, r$, для которых найдется такое число $\alpha$, что $p = \lfloor \alpha \rfloor$, $q = \lfloor \alpha^2 \rfloor$, $r = \lfloor \alpha^3 \rfloor$. Вероятно, таких троек и в самом деле бесконечно много, и быть может, это даже нетрудно доказать, однако снова авторам неизвестны эти результаты. Нам известно лишь существование бесконечного множества пар простых чисел $p, q$ вида $p = \lfloor \alpha \rfloor$, $q = \lfloor \alpha^2 \rfloor$ (см. [Balog 1989]).

**1.79.** Для последовательности $\mathcal{A} = (a_n)$ определим последовательность $D(\mathcal{A})$ с общим членом $(|a_{n+1} - a_n|)$. Пусть $\mathcal{P}$ — последовательность всех простых чисел. Рассмотрим последовательности $D(\mathcal{P})$, $D(D(\mathcal{P}))$ и т. д. Верно ли, что все эти последовательности начинаются с единицы? Одлыжко проверил это утверждение для первых $3 \cdot 10^{11}$ последовательностей [Ribenboim 1996], однако общее доказательство до сих пор неизвестно.

**1.80.** Найдите большие простые числа вида $(2^n + 1)/3$, отталкиваясь от известных теорем о допустимых малых делителях и т. п. Свежие примеры Р. Макинтоша таковы:

$$p = (2^{42737} + 1)/3, \quad q = (2^{83339} + 1)/3, \quad r = (2^{95369} + 1)/3.$$

Эти числа относятся к вероятно простым числам (см. гл. 3). На современном уровне развития вычислительной техники получить строгие доказательства их простоты не представляется возможным (и, видимо, в обозримом будущем ситуация не изменится!).

**1.81.** На практике кандидаты $M_p = 2^p - 1$ в простые числа Мерсенна обычно отбраковываются путем предъявления некоторого собственного простого делителя. Разработайте программное обеспечение для поиска делителей чисел

Мерсенна, позволяющее получить наибольшие известные на сегодняшний день простые числа Мерсенна. Для этого воспользуйтесь известным видом произвольных делителей числа $M_p$ и реализуйте последовательный отбор кандидатов. Ваша программа должна уметь устанавливать такие факты, как, например,

$$460401322803353 \mid 2^{20295923} - 1.$$

По вопросам, связанным с такими большими числами Мерсенна см. упр. 1.82.

**1.82.** В доступной с вычислительной точки зрения области до $2^{20000000}$ была предпринята как минимум одна попытка доказательства составности числа посредством не поиска делителей, а теста Люка—Лемера на простоту. Результат Дж. Спенса — составность числа $2^{20295631} - 1$ — до сих пор не проверен. Так что, на настоящий момент, можно сказать, что это «чисто» составное число, поскольку ни один из его собственных делителей не выписан явно. Заметим, что это гигантское число Мерсенна превосходит даже $F_{24}$, составность которого была доказана недавно. Однако результат, касающийся $F_{24}$, был тщательно проверен независимыми исследованиями, поэтому именно это число Мерсенна можно считать наибольшим «чисто» составным числом на сегодняшний день.

Проведенные рассуждения приводят к постановке исследовательской проблемы. Для начала отметим любопытную дилемму, состоящую в том, что эта «игра в чистую составность» может привести к тривиальным утверждениям, как указал Л. Вашингтон (см. [Lenstra 1991]). В самом деле, если доказана составность числа $C$, то числа $2^C - 1$, $2^{2^C-1} - 1$ и т. д. автоматически являются составными. Поэтому при отсутствии новых знаний о делителях чисел этой цепочки смысл поиска «наибольшего чисто составного числа» оказывается под сомнением. Кроме того, заметим, что если $C \equiv 3 \pmod 4$ и число $2C + 1$ окажется простым, то это число будет делить $2^C - 1$. По-видимому, такое число $C$ может быть чисто составным, однако, следующий элемент цепочки, а именно $2^C - 1$, будет иметь явный делитель. Итак, вот поле для исследовательской деятельности: найдите и докажите составность некоторого числа $C \equiv 3 \pmod 4$, так чтобы никто не знал никаких собственных делителей числа $C$ (т. е. найти их должно быть нелегко). Вы также знаете, что число $2C + 1$ будет простым, поэтому Вам *также* известен и явный делитель числа $2^C - 1$. Отметим, что доказать простоту числа $2C + 1$ без ссылки на разложение на множители числа $C$ совсем не просто. Для этих целей применимы методы, основанные на разложении на множители числа $C + 1$, которые мы рассмотрим в гл. 4.

**1.83.** Несмотря на то, что нам не известно, существует ли бесконечно много простых чисел Мерсенна или Ферма, для некоторых специальных классов чисел уже получены соответствующие результаты. Обозначим $n$-е число Каллена через $C_n = n2^n + 1$. Эти и другие аналогичные числа послужили плодородной почвой для возникновения разнообразных исследовательских направлений.

Одним из таких направлений является численный поиск простых чисел Каллена, начинающийся, быть может, с построения строгого теста на простоту для этих чисел. Схожие задачи возникают и в связи с числами Серпинского, к рассмотрению которых мы перейдем ниже.

Хорошее нетрудное упражнение заключается в доказательстве существования бесконечного множества составных чисел Каллена. Для этого, например, можно рассмотреть свойства, скажем, $C_{p-1}$ при нечетных простых значениях $p$. Другой способ заключается в том, чтобы доказать, что $C_n$ кратно 3 при $n \equiv 1, 2 \pmod 6$ и $C_n$ кратно 5 при $n \equiv 3, 4, 6, 17 \pmod{20}$. Вообще говоря, можно показать, что существуют $p-1$ классов вычетов по модулю $p(p-1)$ для индекса $n$, при которых $C_n$ делится на простое число $p$. При помощи методов решета можно показать, что множество индексов $n$, для которых число $C_n$ составное, имеет асимптотическую плотность 1 (см. [Hooley 1976]).

Еще одним классом чисел, о которых, по меньшей мере, хоть что-то известно, являются числа Серпинского, т. е. такие числа $k$, для которых число $k \cdot 2^n + 1$ является составным для всех натуральных значений $n$. Серпинский доказал бесконечность множества таких чисел $k$. Докажите этот результат и, кроме того, вслед за Серпинским покажите, что существует бесконечная арифметическая прогрессия из натуральных чисел $k$, для которых число $k \cdot 2^n + 1$ является составным для всех натуральных значений $n$. Каждое известное число Серпинского является некоторым членом такой бесконечной арифметической прогрессии. Например, наименьшее известное число Серпинского $k = 78557$ входит в бесконечную арифметическую прогрессию чисел Серпинского. Быть может, поиск этой прогрессии доставит Вам удовольствие. Интересной открытой проблемой вычислительной теории чисел является вопрос минимальности числа 78557. С другой стороны, Эрдёш и Одлыжко доказали, что существует множество положительной асимптотической плотности, состоящее из нечетных чисел $k$, для каждого из которых существует по меньшей мере одно число $n$, для которого $k \cdot 2^n + 1$ — простое число (см. [Guy 1994]).

**1.84.** Реализуйте (попеременно для $+$ и $-$) машинный поиск больших простых чисел вида $n = k \cdot 2^q \pm 1$ (получится пара простых близнецов). При этом считайте, что показатель $q$ фиксирован, а $k$ принимает небольшие значения. Наша цель — отбросить те значения $k$, для которых число $n$ явно составное. Сначала опишите строго, как можно отбросить такие значения $k$ путем просеивания через множество простых нечетных чисел $p \leqslant B$, где граница $B$ фиксирована. Затем ответьте на следующий, важный с практической точки зрения вопрос: если число $k$ не было отсеено, то какова условная эвристическая «вероятность» простоты числа $n$?

Отметим, что в гл. 3 содержится материал, важный для практической оптимизации описанного поиска простых чисел. Проблема состоит в поиске оптимального соотношения между просеиванием значений $k$ и непосредственным применением теста на простоту для оставшихся кандидатов $k \cdot 2^q \pm 1$. Здесь будет уместно сказать, что при некоторых ограничениях на числа $q$ и $k$ известны относительно быстрые детерминистические методы установления простоты (см. гл. 4).

**1.85.** Изучение $n$-ок простых чисел может оказаться столь же интересным, сколь и полным загадок. Сначала докажите несложный факт, что существует лишь одна тройка простых чисел $\{p, p+2, p+4\}$. Затем рассмотрите шаблон

вида $\{p, p+a, p+b\}$, где параметры $a, b$ фиксированы. Какими должны быть значения этих параметров для того, чтобы существовало бесконечное множество таких троек простых чисел? Опишите эффективный алгоритм для поиска этих троек. Например, в случае $(a = 2, b = 6)$ начальное простое число

$$p = 2^{3456} + 5661177712051$$

дает соответствующую тройку простых чисел (результат получен Т. Форбсом в 1995 г., а простота доказана в 1998 г. Ф. Морейном [Forbes 1999]).

Перейдем, далее, к четверкам простых чисел. Проведите эвристические рассуждения в пользу существования бесконечного множества четверок $\{p, p+2, p+6, p+8\}$, составленных из простых чисел. Наибольшая из известных на сегодняшний день такая четверка состоит из простых чисел колоссальной длины — более 1000 цифр (см. [Forbes 1999]).

Далее докажите, что существует ровно одна шестерка простых чисел $\{p, p+2, p+6, p+8, p+12, p+14\}$. Затем заметьте, что при $p = 11$ существует семерка простых чисел $\{p, p+2, p+6, p+8, p+12, p+18, p+20\}$. Найдите другую семерку с тем же шаблоном. Отметим, что наибольшая найденная семерка с этим шаблоном, полученная в 1997 г. Аткином, начинается с числа

$$p = 4269551436942131978484635747263286365530029980299077\backslash$$
$$5938011114100367923769 1$$

**1.86.** Исследуйте числа Смарандаша—Веллена

$$w_n = (p_1)(p_2)\ldots(p_n),$$

полученные путем последовательной записи простых чисел друг за другом (т. е. конкатенации). Например, начальными элементами последовательности $w_n$ являются числа 2, 23, 235, 2357, 235711, ....

Сначала докажите известный результат о том, что последовательность $w_n$ содержит бесконечно много составных чисел. (Подсказка: воспользуйтесь доказанным Литлвудом свойством разности $\pi(x; 3, 1) - \pi(x; 3, 2)$: она принимает сколь угодно большие по модулю положительные и отрицательные значения.) Затем найдите (быть может, эвристическую, недоказанную) асимптотическую оценку для количества простых чисел Смарандаша—Веллена, не превосходящих $x$. В частности, примером такого простого числа служит

$$w_{128} = 23571113171923\ldots719.$$

Сколько цифр содержится в его десятичной записи? Несмотря на всю грандиозность размеров этого числа, известны и еще большие экземпляры (по меньшей мере, вероятно) простых чисел (см. [Wellin 1998] и [Weisstein 2005]).

**1.87.** Докажите несложное утверждение о том, что если $k$ простых чисел, превосходящих $k$, образуют арифметическую прогрессию, то ее разность $d$ делится на каждое простое число, не превосходящее $k$. Найдите как можно более длинную арифметическую прогрессию, состоящую из простых чисел. Отметим, что $k = 22$ было рекордом, установленным в 1995 г. (см. [Pritchard et al. 1995]), а в

2004 г., согласно работе [Frind et al. 2004], была установлена простота чисел

$$56211383760397 + k \cdot 44546738095860$$

для каждого целого $k \in [0, 22]$, и, тем самым, поставлен новый рекорд — 23 простых числа. Разложите на множители разность прогрессии $d = 44546738095860$ и получите пример, подтверждающий наше утверждение.

Найдите арифметическую прогрессию, состоящую из $j$ *последовательных* простых чисел. Текущий рекорд $j = 10$ принадлежит М. Топлику [Dubner et al. 1998], причем соответствующая прогрессия имеет вид $\{P+210m : m = 0, \ldots, 9\}$, где

$$P = 10099697246971424763778665558796984032950932468919004\backslash$$
$$18036034177589043417033488821590672297 19.$$

В связи с этим примером в книге [Dubner et al. 1998] сделано интересное утверждение (мы приводим цитату):

> «Несмотря на то, что многие указывают нам на равенство $10 + 1 = 11$, мы убеждены в том, что поиск арифметической прогрессии из одиннадцати последовательных простых чисел является гораздо более трудной задачей. Наименьшее расстояние между такими простыми числами должно превосходить 2310 (а не 210), причем числа, с которыми придется иметь дело при поиске, будут состоять из сотен цифр. Значит, нам нужна или новая идея, или увеличение вычислительных мощностей в триллион раз. Так что, по нашему мнению, рекорд из десяти простых чисел продержится еще долго.»

**1.88.** Заметим, что 61 делит $67 \cdot 71 + 1$. Существуют ли три бóльших последовательных простых числа $p < q < r$, для которых $p \mid qr + 1$ (см. [Honaker 1998])? Гаццони в электронном письме приводит следующие эвристические рассуждения в пользу того, что число таких троек, по видимому, не бесконечно. Существует гипотеза, что для разности между последовательными простыми числами $p$ и $p'$ справедлива оценка $p' - p = O(\ln^2 p)$. Мы предположим, что справедливо даже более слабое (но также недоказанное) соотношение $p' - p = O(p^c)$, где $c < 1/2$. Тогда если $p, q, r$ — последовательные простые числа, $q = p + s$ и $r = p + t$, то $st = O(p^{2c})$. Но $qr + 1 = (p + s)(p + t) + 1 \equiv st + 1 \pmod{p}$, поэтому для достаточно больших $p$ мы получаем $qr + 1 \not\equiv 0 \pmod{p}$.

**1.89.** Несмотря на то, что утверждение, обратное к теореме 1.3.1, неверно, можно поставить следующий вопрос. Пусть $q$ — простое число Мерсенна. Обязательно ли $2^q - 1$ также будет простым числом Мерсенна? Обоснуйте отрицательный ответ на этот вопрос, приведя пример такого простого числа Мерсенна $q$, что число $2^q - 1$ является составным. (Для этого можно воспользоваться табл. 1.2, считая, что она содержит все простые числа Мерсенна вплоть до наибольшего из известных.) Родственный предыдущему вопрос о том, являются ли числа

$$c_1 = 2^2 - 1 = 3, \quad c_2 = 2^{c_1} - 1 = 7, \quad c_3 = 2^{c_2} - 1 = 127,$$

и т. д. простыми, остается открытым. Поскольку члены этой последовательности растут чрезвычайно быстро (уже $c_5$ содержит более $10^{37}$ цифр), метод пробных делений может оказаться единственным применимым к ним методом разложения на множители, хотя и этот простой алгоритм может не пройти на стандартных ЭВМ. (Для усиления этого скептицизма покажите, например, что наименьший простой делитель числа $c_5$ превосходит $c_4$.)

В духе таких изящных гипотез и в связи с описанной в тексте «обновленной гипотезой Мерсенна» Селфридж назначил премии (по \$1000 каждая) за исследование характера (простое/составное) чисел

$$2^{B(31)}-1,\quad 2^{B(61)}-1,\quad 2^{B(127)}-1,$$

где $B(p)=(2^p+1)/3$. Перед непосредственным написанием программы для исследования этих чисел Мерсенна мы рекомендуем сначала оценить всю грандиозность их размеров.

**1.90.** Здесь мы обсудим пути получения численного значения постоянной Мертенса $B$ (см. теорему 1.4.2). Сначала докажите формулу

$$B=1-\ln 2+\sum_{n=2}^{\infty}\frac{\mu(n)\ln\zeta(n)+(-1)^n(\zeta(n)-1)}{n}$$

(см. [Bach 1997b]). Затем заметьте, что часть ряда совпадает с постоянной Эйлера, поскольку

$$\gamma+\ln 2-1=\sum_{n=2}^{\infty}(-1)^n\frac{\zeta(n)-1}{n}.$$

Воспользуйтесь известными методами для хорошего приближения $\zeta(n)$ (см. [Borwein et al. 2000]) и с помощью сходящегося как геометрическая прогрессия ряда получите численное значение вида

$$B\approx 0.261497212847642783755426838608695859051566648261 20\ldots.$$

Оцените количество простых чисел, которые потребовались бы для вычисления постоянной $B$ с той же точностью непосредственно по определению. Кроме того, существуют и другие постоянные, допускающие быстро сходящиеся разложения, не требующие явного использования простых чисел. Одной из таких «легких» констант является постоянная простых близнецов $C_2$ (см. оценку (1.6)). Еще один пример — постоянная Артина

$$A=\prod_p\left(1-\frac{1}{p(p-1)}\right)\approx 0.3739558136\ldots,$$

равная предполагаемой относительной асимптотической плотности множества простых чисел с первообразным корнем 2 (более подробно по этому вопросу см. [Bach and Shallit 1996]). Попробуйте вычислить постоянные $C_2$, $A$ и другие интересные постоянные, такие, как сингулярный ряд в соотношении (1.12), с некоторой представляющей интерес точностью, но без обращения к явным значениям простых чисел, как было сделано выше в случае постоянной Мертенса. Известным исключением здесь является постоянная Бруна, для которой до сих

пор не известен алгоритм вычисления с полиномиальным временем (см. статью [Borwein et al. 2000], в которой содержится обширное описание подобных применений численных значений дзета-функции Римана). Любопытный способ ускорения сходимости ряда для постоянной Мертенса предложен в статье [Lindqvist and Peetre 1997].

**1.91.** Согласно теореме Ландау (независимо доказанной и Рамануджаном), для асимптотической плотности чисел $n$, представимых в виде $a^2+b^2$, справедлива асимптотическая формула

$$\#\{1 \leqslant n \leqslant x:\ r_2(n) > 0\} \sim L\frac{x}{\sqrt{\ln x}},$$

где постоянная Ландау—Рамануджана

$$L = \frac{1}{\sqrt{2}} \prod_{p\equiv 3 \pmod 4} \left(1-\frac{1}{p^2}\right)^{-1/2} = 0.764223653\ldots$$

Вычислительным аспектом здесь является вопрос о создании быстрого алгоритма, позволяющего вычислять постоянную $L$ с высокой точностью, например, методами упр. 1.90 (по этому вопросу см. [Shanks and Schmid 1966] и [Flajolet and Vardi 1996]). Любопытная связь между значением $L$ и вероятной трансцендентностью иррационального числа $z = \sum_{n\geqslant 0} 1/2^{n^2}$ подмечена в работе [Bailey et al. 2003].

**1.92.** Посредством соответствующих вычислений покажите, что гипотеза выпуклости 1.2.3 не совместима с гипотезой о $k$-кортежах простых чисел 1.2.1 (см. [Hensley and Richards 1973]). Отметим, что эти авторы показали, что если эта гипотеза о $k$-кортежах верна, то должно существовать такое целое $y$, что

$$\pi(y+20000) - \pi(y) > \pi(20000).$$

Таким образом, для доказательства указанной несовместности достаточно показать, что промежуток $(0, 20000]$ содержит «допустимое» множество из более чем $\pi(20000)$ элементов (множество целых чисел является допустимым, если для каждого простого числа $p$ существует по меньшей мере один класс вычетов по модулю $p$, не представленный в этом множестве). Если конечное множество $S$ допустимо, то из гипотезы о $k$-кортежах простых чисел следует существование бесконечного множества таких натуральных чисел $n$, что число $n+s$ является простым для всех $s \in S$. Следовательно, результат Хенсли и Ричардса будет доказан, если установить, что для каждого простого числа $p \leqslant 20000$ существует класс вычетов $a_p$, для которого при исключении из промежутка $(0, 20000]$ всех чисел, сравнимых с $a_p$ по модулю $p$, оставшееся (допустимое) множество будет состоять более чем из $\pi(20000)$ элементов. В статье [Vehka 1979] указан более современный пример, а именно, допустимое множество из 1412 элементов промежутка $(0, 11763]$ (в то время как $\pi(11763) = 1409$). В магистерской диссертации, защищенной в Brigham Young University в 1996 г., Н. Ярвису удалось уменьшить исходное значение 20000 из расчетов Хенсли и Ричардса до 4930. Наименьшее значение $y$, для которого промежуток $(0, y]$ содержит допустимое множество из более чем $\pi(y)$ элементов, до сих пор не

известно, однако в работе [Gordon and Rodemich 1998] показано, что это число $y$ не может быть меньше, чем 1731. Интересный анализ особенно плотных допустимых множеств, основанный на реальных вычислениях, описан в книге [Bressoud and Wagon 2000]. Вэгон уменьшил результат Ярвиса 4930 до 4893. Текущий рекорд таков: $2077 < y \leqslant 3159$ (актуальная информация размещена на сайте [Engelsma 2004]).

Задача обращения такого большого допустимого множества в искомый контрпример к гипотезе выпуклости представляется очень не простой. Если и есть надежда на опровержение этой гипотезы, не связанное с непосредственным доказательством гипотезы о $k$-кортежах простых чисел, то ключ к разгадке может таиться в поиске длинных и плотных скоплений простых чисел. И здесь нельзя недооценивать возможности вычислительной теории чисел. Например, как указано в разд. 3.7.2, мы можем получать оценки $\pi(x)$ для очень больших значений $x$. Возможно, настанет день, когда посредством вычислений можно будет получить оценку снизу подходящей разности $\pi(x+y)-\pi(x)$, скажем, *не* зная все подразумеваемые простые числа, и, тем самым, решить этот удивительный вопрос о совместности.

**1.93.** Можно предложить следующий простой способ проверки, является ли число $p$ простым Вильсона: достаточно произвести непосредственное умножение чисел $1,\ldots,p-1$, каждый раз беря остаток от деления на $p^2$. Однако этот подход требует много избыточных усилий. В самом деле, если число $N$ четно, то можно воспользоваться равенством

$$N! = 2^{N/2}\,(N/2)!\,N!!,$$

где через $N!!$ обозначено произведение всех нечетных чисел из отрезка $[1, N-1]$. Докажите, что при помощи этого соотношения можно сократить число описанных выше операций (умножение + взятие остатка), примерно равное $p$, до $3p/4$.

Попробуйте еще сильнее уменьшить это значение, применяя более тонкое факториальное соотношение, скажем, рассмотрев большее количество классов эквивалентности для чисел, меньших, чем $N$ (т. е. использовав не только четность).

**1.94.** Исследуйте, каким образом тождество Грэнвилля

$$\prod_{j=1}^{m-1} \binom{p-1}{\lfloor jp/m \rfloor} \equiv (-1)^{(p-1)(m-1)/2}(m^p - m + 1) \pmod{p^2},$$

где $1 < m < p$ и $p$ простое число, позволяет ускорить проверку того, является ли $p$ простым числом Вильсона. Это и другие ускоряющие тождества рассмотрены в статье [Crandall et al. 1997].

**1.95.** Исследуйте функцию $\omega(n)$, равную количеству различных простых делителей числа $n$, со статистической точки зрения. Статистические свойства этой функции следуют из ряда замечательных элементарных утверждений. Например, согласно знаменитой теореме Эрдёша—Каца, величина

$$\frac{\omega(n) - \ln\ln n}{\sqrt{\ln\ln n}}$$

имеет нормальное гауссовское распределение с нулевым средним и единичной дисперсией. Таким образом, множество натуральных чисел $n$, для которых введенная статистика не превосходит $u$, имеет асимптотическую плотность $\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{u} e^{-t^2/2}\, dt$. История этого вопроса изложена в книге [Ruzsa 1999].

Несмотря на свою значимость, эти факты основаны на элементарных рассуждениях. Исследуйте возможность аналитического подхода при помощи красивого формального тождества

$$\sum_{n=1}^{\infty} \frac{2^{\omega(n)}}{n^s} = \frac{\zeta^2(s)}{\zeta(2s)}.$$

Можно предложить такое полезное и увлекательное упражнение аналитического толка: перейдя в этом равенстве к пределу при $s \to 1^+$, докажите существование бесконечного множества простых чисел. Какую еще информацию о функции $\omega$ можно получить при таком аналитическом подходе?

Помимо прочего, попробуйте исследовать удивительную гипотезу Дж. Селфриджа о том, что количество различных простых делителей числа Ферма, т. е. $\omega(F_n)$, *не* является монотонной (неубывающей) функцией от $n$. Отметим, что согласно табл. 1.3 эта гипотеза вполне актуальна. (Селфридж склонен полагать, что разложение на множители числа $F_{14}$ может позволить решить эту проблему благодаря малому количеству делителей.) Но пока эта гипотеза остается для нас недостижимой в том смысле, что у нас нет достаточного количества разложенных на множители чисел Ферма. С другой стороны, можно провести эвристические рассуждения, показывающие, в некотором смысле, «вероятность» истинности гипотезы Селфриджа. Более того, можно даже сказать, что эта вероятность равна нулю на том основании, что каждое число Ферма примерно равно *квадрату* предыдущего. В самом деле, в силу теоремы Эрдёша—Каца выполнение неравенства $\omega(b) \geqslant \omega(a)$ для двух случайных натуральных чисел $a, b$, для которых $b \approx a^2$, является крайне маловероятным.

Глава 2

# АППАРАТ ТЕОРИИ ЧИСЕЛ

В этой главе мы уделим особое внимание набору тех фундаментальных средств и отвечающих им вычислительных алгоритмов, которые применяются для исследования простых чисел и разложения на множители. Более совершенные целочисленные алгоритмы, в том числе современные модификации классических алгоритмов настоящей главы, рассматриваются более подробно в гл. 9. У читателя, возможно, появится желание время от времени заглядывать в эту главу, особенно, когда возникает вопрос о сложности вычислений и оптимизации.

## 2.1. Модулярная арифметика

Через всю теорию простых чисел и факторизации проходит понятие модулярной арифметики, постоянно напоминая, что одно из великих изобретений математики состояло в том, чтобы рассматривать числа по модулю $N$, тем самым эффективно сжимая бесконечное множество целых чисел в конечное множество вычетов. Многие теоремы о простых числах используют редукцию по модулю $p$, а большинство методов факторизации оперируют с вычетами по модулю $N$, где $N$ — число, которое требуется разложить на множители.

Несколько слов о терминологии. Здесь и на протяжении всей книги через $x \bmod N$ будем обозначать наименьший неотрицательный вычет $x \pmod N$. Употребление mod без скобок удобно, когда под этим подразумевается шаг алгоритма или машинная операция (модуль в операторном смысле будет рассматриваться подробнее в разд. 9.1.3). Так, запись $x^y \bmod N$ означает, что $y$-я степень $x$ отображается на отрезок $[0, N-1]$. Если $x$ взаимно просто с $N$, то допускаются отрицательные значения показателя $y$, так что операция $x^{-1} \bmod N$ возвращает приведенное значение обратного элемента и т. д.

### 2.1.1. Наибольший общий делитель и обратный элемент

В настоящем разделе представлены алгоритмы решения одной из самых старых задач вычислительной теории чисел — нахождения наибольшего общего делителя НОД $(x, y)$. С ней тесно связана задача инвертирования, т. е. вычисление $x^{-1} \bmod N$. Эта операция (когда она возможна) возвращает единственное целое число $y \in [1, N-1]$, такое, что $xy \equiv 1 \pmod N$. Связь между вычислением НОД и инвертированием сразу очевидна из следующего фундаментального результата.

**Теорема 2.1.1** (линейное соотношение для НОД). *Если $x, y$ — не равные одновременно 0 целые числа, то найдутся такие целые $a, b$, что*

$$ax + by = \text{НОД}(x, y). \tag{2.1}$$

Доказательство. Пусть $g$ — наименьшее положительное целое вида $ax + yb$, где $a, b$ — целые числа. (По крайней мере одно число такого вида существует, например, $x^2 + y^2$.) Утверждается, что $g = \text{НОД}(x, y)$. Понятно, что любой общий делитель $x$ и $y$ делит $g = ax + by$. Следовательно, $\text{НОД}(x, y)$ делит $g$. Предположим, что $g$ не делит $x$. Тогда $x = tg + r$ для некоторого целого числа $r$, $0 < r < g$. Получаем, что $r = (1 - ta)x - tby$, а это противоречит минимальности $g$. Значит, $g$ делит $x$. Аналогично доказывается, что $g$ делит $y$. Таким образом, $g = \text{НОД}(x, y)$. □

Из (2.1) немедленно получаем следствие для инвертирования: если $x, y$ — положительные целые числа, и $\text{НОД}(x, y) = 1$, то уравнение $ax + by = 1$ разрешимо, а

$$b \bmod x, \ a \bmod y$$

являются обратными к $y \pmod{x}$ и $x \pmod{y}$ соответственно.

Однако с вычислительной точки зрения доказательство теоремы 2.1.1 имеет тот недостаток, что не указывает на способ нахождения решения $a, b$ уравнения (2.1). Поэтому далее мы рассмотрим основополагающие, классические методы вычислений, и начнем со знаменитого алгоритма Евклида, стоящего в центре классического подхода. Восходящая к 300 г. до н. э., это, вероятно, одна из старейших вычислительных схем, если не самая старая. В этом и в последующих алгоритмах шаг, переопределяющий значения пары переменных $x, y$, записывается как

$$(x, y) = (f(x, y), g(x, y)).$$

Это означает, что вместо пары $(x, y)$ должна быть подставлена пара $(f, g)$, вычисленная по первоначальной паре $(x, y)$. Аналогично, векторное соотношение $(a, b, c, \ldots) = \ldots$ большей размерности означает преобразование компонент в левой части, состоящее в подстановке соответствующих компонент из правой части уравнения, вычисляемых по первоначальному набору. (Это правило преобразования компонент вектора обсуждается в приложении.)

**Алгоритм 2.1.2** (алгоритм Евклида для нахождения наибольшего общего делителя). **Для целых чисел $x, y$, таких, что $x \geqslant y \geqslant 0$ и $x > 0$, алгоритм вычисляет $\text{НОД}(x, y)$.**

1. **[Цикл алгоритма Евклида]**
   **while**$(y > 0)$ $(x, y) = (y, x \bmod y)$;
   **return** $x$;

Трудно поверить, но этот алгоритм, настолько простой и элегантный, насколько это вообще возможно, совсем не прост для описания в категориях сложности. Хотя касательно деталей поведения алгоритма остается до сих пор открытым ряд интересных вопросов, в основном его сложность описывается следующей теоремой.

**Теорема 2.1.3** (Ламе, Диксон, Хайльбронн). *Пусть $x > y$ — целые числа из отрезка $[1, N]$. Тогда число шагов в цикле алгоритма Евклида* 2.1.2 *не превосходит*

$$\left\lceil \ln\left(N\sqrt{5}\right) / \ln\left(\left(1+\sqrt{5}\right)/2\right)\right\rceil - 2,$$

*а среднее (по всем выборам $x, y$) число шагов в цикле асимптотически эквивалентно*

$$\frac{12\ln 2}{\pi^2}\ln N.$$

Первое утверждение теоремы следует из интересной взаимосвязи алгоритма Евклида и теории простых цепных дробей (см. упр. 2.4). При выводе второго утверждения используется теория меры цепных дробей.

Полагая, что $x, y$ — величины порядка $N$, и используя в алгоритме Евклида стандартный метод приведения по модулю, можно показать, что общая сложность вычисления НОД составляет

$$O\left(\ln^2 N\right)$$

битовых операций, или порядка квадрата от количества цифр в операнде (см. упр. 2.6). Эта оценка, безусловно, может быть улучшена при помощи современных методов, и не только за счет ускорения операции приведения по модулю, как будет показано в последней главе.

Алгоритм Евклида можно распространить на вычисление обратного элемента. На самом деле, такой расширенный алгоритм Евклида будет находить все решения уравнения (2.1).

**Алгоритм 2.1.4** (расширенный алгоритм Евклида для вычисления НОД и инвертирования). **По заданным целым числам $x, y$, где $x \geqslant y \geqslant 0$ и $x > 0$, алгоритм возвращает тройку целых чисел $(a, b, g)$, таких, что $ax + by = g =$ НОД$(x, y)$. (Таким образом, если $g = 1$ и $y > 0$, то вычеты $b \pmod x$, $a \pmod y$ являются обратными к $y \pmod x$, $x \pmod y$ соответственно.)**

1. **[Инициализация]**
   $(a, b, g, u, v, w) = (1, 0, x, 0, 1, y)$;
2. **[Цикл расширенного алгоритма Евклида]**
   **while** $(w > 0)$ {
   $q = \lfloor g/w \rfloor$;
   $(a, b, g, u, v, w) = (u, v, w, a - qu, b - qv, g - qw)$;
   }
   **return** $(a, b, g)$;

Поскольку этот алгоритм одновременно вычисляет как НОД, так и оба обратных элемента (если числа на входах взаимно просты и положительны), он встроен во многие популярные вычислительные пакеты. Любопытные детали, касающиеся организации вычислений в этом и родственных ему алгоритмах, можно найти в книгах [Cohen 2000], [Knuth 1981]. В гл. 9 рассматриваются современные усовершенствования, в том числе асимптотически более быстрые алгоритмы для НОД, более быстрое инвертирование, инвертирование по спе-

циальным модулям и т. д. В заключение укажем, что в разд. 2.1.2 будет изложен простой и перспективный для практической реализации метод инвертирования, основанный на тождестве (2.3).

## 2.1.2. Степени

Знаменитая теорема Эйлера гласит, что

$$a^{\varphi(m)} \equiv 1 \pmod{m} \tag{2.2}$$

для всех положительных целых чисел $m$, если $a, m$ взаимно просты. В частности, для простого $p$ выполняется сравнение

$$a^{p-1} \equiv 1 \pmod{p},$$

которое часто используется в качестве простого первичного (но не абсолютного) теста на простоту. Для нас важно то, что возведение в степень является фундаментальной операцией теории простых чисел, и особый интерес представляет возведение в степень с приведением по модулю. Среди многих примеров использования операции возведения в степень выделим следующий. Очевидный метод инвертирования основывается на наблюдении, что если существует $a^{-1} \pmod{m}$, то всегда выполняется

$$a^{-1} \bmod m = a^{\varphi(m)-1} \bmod m, \tag{2.3}$$

и этот способ инвертирования может соперничать с алгоритмом 2.1.4 в эффективности машинной реализации.

Одним из важнейших результатов вычислительной теории является то, что для возведения некоторого $x$ в $n$-ю степень обычно не требуется по очереди перемножать все $n$ символов $x * x * \ldots * x$, как записано. Мы приведем принципиально более эффективный (для больших степеней) рекурсивный алгоритм возведения в степень, который легко записывается и прост для понимания. В качестве объектов, возводимых в степень, могут выступать целые числа, элементы конечного поля, многочлены или что-нибудь еще. Для алгоритма существенно лишь то, чтобы $x$ принадлежал некоторой полугруппе, т. е. множеству, на котором определено произведение $x * x * \ldots * x$.

**Алгоритм 2.1.5** (рекурсивный алгоритм для возведения в степень). **Алгоритм вычисляет $x^n$, где $x$ принадлежит некоторой полугруппе, а $n$ — положительное целое число.**

1. [Рекурсивная функция $pow$]

```
pow(x, n) {
    if (n == 1) return x;
    if (n — четно) return pow(x, n/2)^2;          // Четный случай.
    return x * pow(x, (n − 1)/2)^2;               // Нечетный случай.
}
```

Этот рекурсивный алгоритм компактен, однако при практической реализации рекомендуется использовать лестничные методы из разд. 9.3.1, по сути не

отличающиеся от данного, но лучше приспособленные для работы с большими массивами данных. Рекурсивную структуру алгоритма 2.1.5 проиллюстрируем на примере вычисления $3^{13} \pmod{15}$. Поскольку $n = 13$, легко видеть, что порядок операций будет следующий

$$\begin{aligned} 3 * pow(3,6)^2 &= 3 * \left(pow(3,3)^2\right)^2 \\ &= 3 * \left(\left(3 * pow(3,1)^2\right)^2\right)^2. \end{aligned}$$

Когда вычисляется $x^n \bmod m$, на каждом шаге алгоритма (четном или нечетном) требуется выполнять приведения по модулю. Например, глядя на полученную выше цепочку степеней с модулем $m = 15$, легко получить ответ $3^{13} \bmod 15 = 3 \cdot \left((-3)^2\right)^2 \bmod 15 = 3 \cdot 6 \bmod 15 = 3$. Обратим внимание на то, что здесь использовалось три возведения в квадрат и два умножения, а в общем случае число таких операций зависит от двоичного разложения показателя $n$ и, как правило, существенно меньше чем $n$. Говоря точнее, если $x$, $n$ — целые числа порядка $m$, то, используя для вычисления $x^n \bmod m$ стандартные методы умножения/сложения и алгоритм 2.1.5, для возведения в степень достаточно $O(\ln^3 m)$ битовых операций (см. упр. 2.17 и разд. 9.3.1).

## 2.1.3. Китайская теорема об остатках

Китайская теорема об остатках (Chinese remainder theorem — CRT) — это замечательная и очень древняя идея. Она объясняет, как найти целое число, зная его вычеты на некотором подходящем множестве сравнительно небольших модулей. Эта теорема была известна еще Сунь Цю, жившему в 1 веке н.э. [Hardy and Wright 1979], [Ding et al. 1996]. Достоверно известно, что в древности китайской теоремой пользовались при подсчете численности войска. Если $n$ солдат построить в шеренги по 7 человек в каждой, то в последней шеренге находится $n \bmod 7$ солдат; выстроив их в шеренги по 11, можно узнать $n \bmod 11$, и т. д. Проделав «достаточное» число подобных операций с небольшими модулями, получим точное значение $n$. На самом деле, не обязательно, чтобы модули были простыми числами — достаточно, чтобы они были попарно взаимно просты.

**Теорема 2.1.6** (китайская теорема об остатках (CRT)). *Пусть $m_0, \ldots, m_{r-1}$ — положительные попарно взаимно простые модули и $M = \prod_{i=0}^{r-1} m_i$. Пусть также заданы $r$ соответствующих им вычетов $n_i$. Тогда система из $r$ уравнений и неравенства*

$$n \equiv n_i \pmod{m_i}, \quad 0 \leqslant n < M$$

*имеет единственное решение. Более того, это решение является в точности наименьшим неотрицательным вычетом по модулю $M$ числа*

$$\sum_{i=0}^{r-1} n_i v_i M_i,$$

*где $M_i = M/m_i$, а $v_i$ — обратные элементы, определяемые из соотношений $v_i M_i \equiv 1 \pmod{m_i}$.*

Поясним этот принцип на простейшем примере. Положим $m_0 = 3$, $m_1 = 5$, $m_2 = 7$, тогда произведение составит $M = 105$, пусть также $n_0 = 2$, $n_1 = 2$, $n_2 = 6$. Мы ищем решение $n < 105$ системы

$$n \equiv 2 \pmod 3,\ n \equiv 2 \pmod 5,\ n \equiv 6 \pmod 7.$$

Сначала определим $M_i$,

$$M_0 = 35,\ M_1 = 21,\ M_2 = 15$$

Затем найдем обратные

$$v_0 = 2 = 35^{-1} \bmod 3, \quad v_1 = 1 = 21^{-1} \bmod 5, \quad v_2 = 1 = 15^{-1} \bmod 7.$$

Наконец, вычисляем

$$\begin{aligned} n &= (n_0 v_0 M_0 + n_1 v_1 M_1 + n_2 v_2 M_2) \bmod M \\ &= (140 + 42 + 90) \bmod 105 \\ &= 62. \end{aligned}$$

Действительно, число 62 обладает заданным набором вычетов $2, 2, 6$ по модулям $3, 5, 7$ соответственно.

Несмотря на свою древность, CRT-алгоритм по сей день находит себе множество приложений. Некоторые из них рассматриваются в гл. 9 и упражнениях к ней. Пока же заметим, что в китайской теореме заложен определенный «параллелизм». Вычисления могут выполняться на независимых машинах, каждая из которых работает со своим небольшим модулем $m_i$, после чего может быть восстановлено окончательное значение. Например, если каждое из чисел $x, y$ записывается не более чем 100 цифрами, то для их умножения можно выбрать набор простых модулей $\{m_i\}$ с произведением $M > 10^{200}$. Машина с номером $i$ вычисляет $((x \bmod m_i) * (y \bmod m_i)) \bmod m_i$, а окончательный результат $x * y$ находится при помощи CRT. Подобным образом, на одной компьютерной микросхеме реализуются независимые умножители, выполняющие операции с небольшими модулями.

Все сказанное означает, что центральной становится задача о восстановлении числа по его остаткам. Действительно, восстановление $n$ оказывается наиболее трудоемким шагом CRT-алгоритма. Заметим, однако, что в случае, когда один и тот же набор модулей используется многократно, некоторая часть вычислений может быть выполнена заранее и только один раз. Из теоремы 2.1.6 видно, что числа $M_i$ и обратные к ним $v_i$ достаточно вычислить однажды, чтобы в дальнейшем применять многократно к различным наборам вычетов $\{n_i\}$. Фактически могут быть предвычислены произведения $v_i M_i$. После этого значение $\sum n_i v_i M_i$ может быть восстановлено за $O(\ln r)$ шагов на компьютере с $r$ параллельными процессорами.

Есть и другие способы организации вычислений в CRT-алгоритме, например, с нахождением остатка по «частичному» модулю на каждом шаге. К таким методам относится алгоритм Гарнера (см. [Menezes et al. 1997]), также допускающий предвычисления.

**Алгоритм 2.1.7** (CRT-алгоритм Гарнера восстановления числа по остаткам с предвычислениями). Сохраняем обозначения теоремы 2.1.6. Пусть $r \geqslant 2$, $m_0, \ldots, m_{r-1}$ — фиксированные попарно взаимно простые модули, произведение которых равно $M$, и задан набор вычетов $\{n_i \pmod{m_i}\}$. Алгоритм возвращает единственное число $n \in [0, M-1]$, имеющее заданный набор вычетов. По выполнении предвычислений алгоритм может применяться многократно для нахождения таких $n$ (если набор $\{m_i\}$ не изменяется).

1. [Предвычисления]
   for($1 \leqslant i < r$) {
   $\mu_i = \prod_{j=0}^{i-1} m_j$;
   $c_i = \mu_i^{-1} \bmod m_i$;
   }
   $M = \mu_{r-1} m_{r-1}$;
2. [Точка повторного входа для задания входного набора $\{n_i\}$]
   $n = n_0$;
   for($1 \leqslant i < r$) {
   $u = ((n_i - n)c_i) \bmod m_i$;
   $n = n + u\mu_i$; // Теперь $n \equiv n_j \pmod{m_j}$ для $0 \leqslant j \leqslant i$;
   }
   $n = n \bmod M$;
   return $n$;

Можно показать, что этот алгоритм более эффективен, чем тот, который получается непосредственным применением теоремы 2.1.6 (см. упр. 2.8). Более того, в случае, когда CRT-алгоритм неоднократно используется с одним и тем же модулем $M$, однократное выполнение шага [Предвычислений] алгоритма 2.1.7 с запоминанием подходящего набора из $r-1$ чисел обеспечивает эффективный механизм повторного входа в программу.

В разд. 9.5.9 мы опишем CRT-алгоритм восстановления числа по набору остатков, эффективность которого основывается не только на предвычислениях, но и на методах быстрого умножения целых чисел.

## 2.2. Полиномиальная арифметика

Многие алгоритмы модулярной арифметики могут быть практически без изменений перенесены на операции с многочленами.

### 2.2.1. Наибольший общий делитель многочленов

Далее мы опишем алгоритмы, аналогичные целочисленным алгоритмам Евклида нахождения НОД и инвертирования из разд. 2.1.1, только для многочленов. Когда говорится о многочленах, первый вопрос, который возникает — это откуда берутся коэффициенты. Мы можем иметь дело с $\mathbf{Q}[x]$, многочленами с рациональными коэффициентами, или же с $\mathbf{Z}_p[x]$, многочленами с коэффициентами из конечного поля $\mathbf{Z}_p$ или еще из какого-нибудь поля. Можно

также рассматривать многочлены с коэффициентами из некоторого кольца, не являющегося полем, как в случае $\mathbf{Z}[x]$ или $\mathbf{Z}_n[x]$, где $n$ не является простым.

Поскольку область, в которой мы должны работать, представляется не очень определенной, возможно, лучше возвратиться к исходным определениям и начать с самого первичного понятия деления с остатком. Для многочленов из $F[x]$, где $F$ — некоторое поле, выполняется теорема о делении с остатком, абсолютно аналогичная теореме для обычных целых чисел. А именно, если $f(x), g(x)$ — многочлены из $F[x]$, и многочлен $f$ отличен от нуля, то существуют (единственные) многочлены $q(x), r(x)$ из $F[x]$, такие, что

$$g(x) = q(x)f(x) + r(x) \text{ и либо } r(x) = 0, \text{ либо } \deg r(x) < \deg f(x). \qquad (2.4)$$

Более того, для нахождения частного $q(x)$ можно воспользоваться известным «школьным» алгоритмом деления, последовательно вычисляя коэффициенты многочленов $q(x)$ и $r(x)$. Из анализа этого метода можно увидеть, что единственным свойством поля, которое используется в алгоритме и которому не удовлетворяет произвольное коммутативное кольцо, является обратимость старшего коэффициента многочлена $f(x)$, на который производится деление. Следовательно, в более общем случае, когда многочлены выбираются из $R[x]$, где $R$ — коммутативное кольцо с единицей, можно выполнять деление с остатком при условии, что старший коэффициент многочлена-делителя унитарен, т.е. имеет мультипликативный обратный в данном кольце.

Например, пусть требуется разделить $x^2$ на $3x + 2$ в кольце многочленов $\mathbf{Z}_{10}[x]$. Обратный к 3 элемент кольца $\mathbf{Z}_{10}$ — это 7 (его можно найти при помощи алгоритма 2.1.4). Находим частное $7x + 2$ и остаток 6.

Окончательно, пусть многочлены $f(x), g(x)$ принадлежат $R[x]$, где $R$ — коммутативное кольцо с единицей, и старший коэффициент многочлена $f$ унитарен в $R$, тогда многочлены $q(x), r(x)$ из $R[x]$, такие, что выполняется (2.4), существуют и определены единственным образом. Обозначаем $r(x) = g(x) \bmod f(x)$. Более подробно о делении многочленов с остатком см. разд. 9.6.2.

Хотя НОД двух многочленов можно иногда определить в более общей алгебраической структуре $R[x]$, однако далее мы ограничим наше рассмотрение более простой $F[x]$, где $F$ — поле. В таком случае алгоритмы и теория останутся почти в точности такими же, как и для целых чисел. (Понятие НОД для случая, когда $R$ — не обязательно поле, обсуждается в разд. 4.3.) Определим НОД двух многочленов, одновременно не равных 0, как многочлен наибольшей степени, делящий оба этих многочлена. Домножая произвольный многочлен, удовлетворяющий такому определению НОД, на ненулевой элемент поля $F$, получаем вновь многочлен, удовлетворяющий этому определению. Чтобы внести однозначность, из всех таких многочленов выберем нормированный многочлен, у которого старший коэффициент равен 1 — именно такой многочлен и обозначим через НОД $(f(x), g(x))$. Окончательно, НОД $(f(x), g(x))$ — это нормированный общий делитель многочленов $f(x)$ и $g(x)$ наибольшей степени. Чтобы нормировать произвольный ненулевой многочлен, нужно просто домножить его на обратный к старшему коэффициенту элемент.

**Алгоритм 2.2.1** (НОД для многочленов). **Для двух заданных многочленов**

$f(x), g(x)$ из $F[x]$, одновременно не равных нулю, алгоритм возвращает $d(x) = \text{НОД}(f(x), g(x))$.

1. [Инициализация]
   Пусть $u(x)$, $v(x)$ — это многочлены $f(x), g(x)$ в таком порядке, что либо $\deg u(x) \geqslant \deg v(x)$, либо $v(x)$ равен 0;
2. [Цикл алгоритма Евклида]
   while$(v(x) \neq 0)$ $(u(x), v(x)) = (v(x), u(x) \bmod v(x))$;
3. [Нормирование многочлена]
   Пусть $c$ — старший коэффициент многочлена $u(x)$;
   $d(x) = c^{-1}u(x)$;
   return $d(x)$;

Так, например, если выбраны многочлены

$$\begin{aligned} f(x) &= 7x^{11} + x^9 + 7x^2 + 1, \\ g(x) &= -7x^7 - x^5 + 7x^2 + 1 \end{aligned}$$

из $\mathbf{Q}[x]$, то последовательность вычислений в алгоритме Евклида будет следующей:

$$\begin{aligned} &(7x^{11} + x^9 + 7x^2 + 1,\ -7x^7 - x^5 + 7x^2 + 1) \\ &\quad\to\ (-7x^7 - x^5 + 7x^2 + 1,\ 7x^6 + x^4 + 7x^2 + 1) \\ &\quad\to\ (7x^6 + x^4 + 7x^2 + 1,\ 7x^3 + 7x^2 + x + 1) \\ &\quad\to\ (7x^3 + 7x^2 + x + 1,\ 14x^2 + 2) \\ &\quad\to\ (14x^2 + 2,\ 0). \end{aligned}$$

Таким образом, окончательно $u(x)$ принимает значение $14x^2+2$, следовательно, наибольший общий делитель $d(x)$ равен $x^2 + \frac{1}{7}$. Разумеется, все вычисления в данном алгоритме должны производиться в кольце многочленов $F[x]$. Поэтому, если в приведенном примере $F = \mathbf{Z}_{13}$, то $d(x) = x^2+2$, если $F = \mathbf{Z}_7$, то $d(x) = 1$, а в случае $F = \mathbf{Z}_2$ алгоритм закончит работу на шаг раньше с результатом $d(x) = x^3 + x^2 + x + 1$.

Помимо НОД двух многочленов, может понадобиться вычисление обратного многочлена. Как и в случае целочисленного инвертирования, решается уравнение

$$s(x)f(x) + t(x)g(x) = d(x)$$

с заданными многочленами $f, g$, где $d(x) = \text{НОД}(f(x), g(x))$.

**Алгоритм 2.2.2** (расширенный алгоритм нахождения НОД многочленов). Пусть $F$ — некоторое поле. По заданным многочленам $f(x), g(x)$ из $F[x]$, одновременно не равных нулю, где $\deg f(x) \geqslant \deg g(x)$ или $g(x) = 0$, алгоритмом вычисляются многочлены $(s(x), t(x), d(x))$ из $F[x]$, такие, что $d = \text{НОД}(f, g)$ и $sg + th = d$. (Для удобства обозначений аргумент $x$ далее будет опускаться.)

1. [Инициализация]
   $(s, t, d, u, v, w) = (1, 0, f, 0, 1, g)$;

2. [Цикл расширенного алгоритма Евклида]
   while($w \neq 0$) {
   $q = (d - (d \bmod w))/w$; // $q$ — это частное от деления $d \div w$.
   $(s, t, d, u, v, w) = (u, v, w, s - qu, t - qv, d - qw)$;
   }
3. [Нормирование многочлена]
   Пусть $c$ — старший коэффициент многочлена $d$;
   $(s, t, d) = (c^{-1}s, c^{-1}t, c^{-1}d)$;
   return $(s, t, d)$;

Если $d(x) = 1$, и никакой из многочленов $f(x), g(x)$ не равен 0, то $s(x)$ — обратный к многочлену $f(x) \pmod{g(x)}$, а $t(x)$ — обратный к $g(x) \pmod{f(x)}$. Очевидно, что используя стандартный метод деления многочленов с остатком, описанный выше, получаем алгоритм сложности $O(D^2)$ операций в поле, где $D$ — наибольшая из степеней входных многочленов (см. [Menezes et al. 1997]).

## 2.2.2. Конечные поля

Примерами бесконечных полей являются поля рациональных чисел $\mathbf{Q}$, действительных чисел $\mathbf{R}$ и комплексных чисел $\mathbf{C}$. Однако в этой книге мы, в основном, будем иметь дело с конечными полями, причем наиболее часто будет встречаться поле

$$\mathbf{F}_p = \mathbf{Z}_p,$$

которое состоит из всех вычетов $0, 1, \ldots, p-1$ по модулю простого числа $p$, а операции выполняются согласно обычным правилам модулярной арифметики.

Пусть задано некоторое поле $F$, и в $F[x]$ выбран многочлен $f(x)$ положительной степени. Рассмотрим фактор-кольцо $F[x]/(f(x))$. Элементами $F[x]/(f(x))$ являются подмножества кольца $F[x]$ вида $\{g(x) + f(x)h(x) : h(x) \in F[x]\}$. Такое подмножество будем обозначать через $g(x) + (f(x))$. Это смежный класс идеала $(f(x))$ с представителем $g(x)$. (В действительности смежный класс может быть представлен *произвольным* своим многочленом, поскольку $g(x) + (f(x)) = G(x) + (f(x))$ равносильно $G(x) \in g(x) + (f(x))$, что равносильно $G(x) - g(x) = f(x)h(x)$ для некоторого $h(x) \in F[x]$, а это равносильно $G(x) \equiv g(x) \pmod{f(x)}$. Таким образом, смежные классы могут рассматриваться как особый вид конгруэнций.) Для каждого смежного класса естественным и однозначным образом определяется канонический представитель — это либо 0, либо многочлен степени, меньшей, чем $\deg f(x)$.

Сложение и умножение смежных классов соответствуют аналогичным операциям для их представителей:

$$\begin{aligned} \big(g_1(x) + (f(x))\big) \;+\; \big(g_2(x) + (f(x))\big) \quad &= \quad g_1(x) + g_2(x) \;+\; (f(x)), \\ \big(g_1(x) + (f(x))\big) \;\cdot\; \big(g_2(x) + (f(x))\big) \quad &= \quad g_1(x)g_2(x) \;+\; (f(x)). \end{aligned}$$

С определенными таким образом операциями сложения и умножения $F[x]/(f(x))$ является кольцом, содержащим изоморфную копию поля $F$: элемент $a \in F$ ассоциируется со смежным классом $a + (f(x))$.

**Теорема 2.2.3.** *Если $F$ — поле, и многочлен $f(x) \in F[x]$ имеет положительную степень, то $F[x]/(f(x))$ является полем тогда и только тогда, когда $f(x)$ — неприводим в $F[x]$.*

Эта теорема позволяет строить новые поля из уже построенных. Пусть, например, задано поле рациональных чисел $\mathbf{Q}$. Рассмотрим неприводимый в поле $\mathbf{Q}[x]$ многочлен $x^2-2$. Смежный класс $a+bx+(f(x))$, где $a, b \in \mathbf{Q}$, будем обозначать просто как $a+bx$. Выпишем правила для сложения и умножения,

$$\begin{aligned}(a_1+b_1x)+(a_2+b_2x) &= (a_1+a_2)+(b_1+b_2)x,\\ (a_1+b_1x)\cdot(a_2+b_2x) &= (a_1a_2+2b_1b_2)+(a_1b_2+a_2b_1)x.\end{aligned}$$

Видно, что многочлены складываются и перемножаются обычным образом, а соотношение $x^2=2$ используется для редукции. Мы только что «построили» поле

$$\mathbf{Q}\left[\sqrt{2}\right] = \left\{a+b\sqrt{2} : a, b \in \mathbf{Q}\right\}.$$

Воспользуемся той же идеей, выбирая конечное поле $\mathbf{F}_7$ в качестве стартового. Пусть $f(x)=x^2+1$. Многочлен степени 2 неприводим над полем $F$ в том и только том случае, когда он не имеет корней в $F$. Нетрудно проверить, что у многочлена $x^2+1$ нет корней в $\mathbf{F}_7$, поэтому он неприводим над этим полем. Следовательно, по теореме 2.2.3, $\mathbf{F}_7[x]/(x^2+1)$ — поле. Элементы этого поля можно обозначать через $a+bi$, где $a, b \in \mathbf{F}_7$ и $i^2=-1$. Построенное нами поле состоит из 49 элементов.

В более общем случае, когда $p$ — простое, и многочлен $f(x) \in \mathbf{F}_p[x]$ степени $d \geqslant 1$ неприводим, $\mathbf{F}_p[x]/(f(x))$ также является конечным полем. Оно содержит $p^d$ элементов. Любопытно, что *все* конечные поля с точностью до изоморфизмов могут быть построены этим способом.

Важным отличием конечных полей от таких полей, как $\mathbf{Q}$ и $\mathbf{C}$, является то, что последовательное прибавление единицы конечного поля к самой себе в конечном счете приводит к 0. На самом деле, количество единиц в первой нулевой сумме должно быть простым числом, в противном случае получим, что произведение двух ненулевых элементов поля равно 0.

**Определение 2.2.4.** Характеристика поля — это аддитивный порядок 1. В случае, когда порядок бесконечен, характеристика полагается равной 0.

Как отмечалось выше, характеристика поля, если она положительна, является простым числом. Особую роль в приложениях играют поля характеристики 2, в основном, из-за простоты выполнения арифметических операций в этих полях.

Приведем некоторые фундаментальные факты классической теории конечных полей.

**Теорема 2.2.5** (основные свойства конечных полей).

*(1) Характеристика конечного поля $F$ отлична от нуля и является простым числом.*

*(2) Для любых двух элементов $a, b$ конечного поля $F$ характеристики $p$ справедливо равенство $(a+b)^p = a^p + b^p$.*

*(3) Любое конечное поле содержит $p^k$ элементов, где $k$ — некоторое положительное целое, а $p$ — характеристика поля.*

*(4) Для заданного простого числа $p$ и показателя $k$ существует единственное (с точностью до изоморфизма) поле из $p^k$ элементов; будем обозначать его через $\mathbf{F}_{p^k}$.*

*(5) Поле $\mathbf{F}_{p^k}$ для каждого $j \mid k$ содержит единственное подполе, изоморфное $\mathbf{F}_{p^j}$, и не содержит никаких других подполей.*

*(6) Мультипликативная группа $\mathbf{F}_{p^k}^*$ ненулевых элементов поля $\mathbf{F}_{p^k}$ является циклической, т. е. она целиком порождается степенями некоторого элемента поля.*

Изучение мультипликативной группы $\mathbf{F}_{p^k}^*$ имеет значение для задач возведения в степень и извлечения корней, а также в криптографии.

**Определение 2.2.6.** Элемент, степени которого порождают всю группу $\mathbf{F}_{p^k}^*$, называется примитивным корнем поля $\mathbf{F}_{p^k}$. Другими словами, это генератор циклической группы $\mathbf{F}_{p^k}^*$.

Так, в примере, приведенном выше, где было построено поле из 49 элементов, а именно $\mathbf{F}_{7^2}$, элемент $3+i$ является примитивным корнем.

В циклической группе из $n$ элементов содержится всего $\varphi(n)$ генераторов, где $\varphi$ — функция Эйлера. Следовательно, в поле $\mathbf{F}_{p^k}$ имеется $\varphi(p^k-1)$ примитивных корней.

Один из способов нахождения примитивных корней опирается на следующий результат.

**Теорема 2.2.7** (тест на примитивный корень). *Элемент $g$ из $\mathbf{F}_{p^k}^*$ является примитивным корнем тогда и только тогда, когда*

$$g^{(p^k-1)/q} \neq 1$$

*для любого простого $q$ — делителя $p^k - 1$.*

Если число $p^k - 1$ может быть разложено на множители, то этот тест предоставляет эффективное средство для поиска примитивного корня. Рассмотрим простой алгоритм нахождения примитивного корня. Выберем $g \in \mathbf{F}_{p^k}^*$ случайным образом, для последовательных простых делителей $q$ числа $p^k - 1$ вычисляем степени $g^{(p^k-1)/q} \bmod p$, и в случае, когда одна из этих степеней равна 1, переходим к другому $g$. Элемент $g$, прошедший полную цепочку проверок, является примитивным корнем, согласно теореме 2.2.7.

В большей части этой книги используется только арифметика в $\mathbf{F}_p$, однако порой нам придется иметь дело и с полями степенного порядка. Хотя в общем случае арифметика в поле $\mathbf{F}_{p^k}$ может осуществляться достаточно сложным образом, примечательно, что эффективность ряда алгоритмов в действительности может повышаться за счет привлечения таких полей. Как мы убедились выше, конечное поле $\mathbf{F}_{p^k}$ можно «построить», найдя предварительно какой-либо неприводимый многочлен $f(x)$ из $\mathbf{F}_p[x]$ степени $k$. Поэтому несколько слов о том, как это можно сделать.

Для каждого элемента $a$ поля $\mathbf{F}_{p^k}$ выполняется свойство $a^{p^k} = a$, т. е. $a$ является корнем многочлена $x^{p^k} - x$. На самом деле этот многочлен разлагается на попарно различные линейные множители над $\mathbf{F}_{p^k}$. Из этого факта следует, что многочлен $x^{p^k} - x$ является произведением *всех* нормированных неприводимых многочленов из $\mathbf{F}_p[x]$, степени которых делят $k$. Отсюда можно вывести формулу для числа $N_k(p)$ нормированных неприводимых многочленов степени $k$ в $\mathbf{F}_p[x]$. Применяя к тождеству

$$\sum_{d|k} dN_d(p) = p^k$$

формулу обращения Мёбиуса, получим

$$N_k(p) = \frac{1}{k}\sum_{d|k} p^d \mu(k/d). \tag{2.5}$$

Здесь $\mu$ — функция Мёбиуса, которая более подробно рассматривается в разд. 1.4.1. Легко видеть, что в последней сумме доминирует член с $d = k$, поэтому $N_k(p)$ приблизительно равно $p^k/k$. Таким образом, примерно 1 из $k$ произвольных нормированных многочленов степени $k$ в $\mathbf{F}_p[x]$ является неприводимым. Поэтому случайный поиск такого многочлена должен привести к успеху за $O(k)$ попыток. Однако, как убедиться в том, что многочлен является неприводимым? Ответ дает следующая теорема.

**Теорема 2.2.8.** *Пусть $f(x)$ — многочлен из $\mathbf{F}_p[x]$ положительной степени $k$. Тогда следующие утверждения равносильны:*

*(1) $f(x)$ неприводим;*
*(2)* НОД$(f(x), x^{p^j} - x) = 1$ *для всех* $j = 1, 2, \ldots, \lfloor k/2 \rfloor$;
*(3)* $x^{p^k} \equiv x \pmod{f(x)}$ *и* НОД$(f(x), x^{p^{k/q}} - x) = 1$ *для каждого простого* $q|k$.

Эта теорема, доказательство которой мы оставляем в качестве упр. 2.15, является обоснованием следующих двух тестов на неприводимость.

**Алгоритм 2.2.9** (тест на неприводимость 1). Для заданного простого числа $p$ и многочлена $f(x) \in \mathbf{F}_p[x]$ степени $k \geqslant 2$ алгоритм определяет, является ли многочлен $f(x)$ неприводимым над $\mathbf{F}_p$.

1. [Инициализация]
   $g(x) = x$;
2. [Цикл теста]
   for($1 \leqslant i \leqslant \lfloor k/2 \rfloor$) {
   $g(x) = g(x)^p \bmod f(x)$; // Возведение в степень (алгоритм 2.1.5).
   $d(x) =$ НОД$(f(x), g(x) - x)$; // Вычисление НОД (алгоритм 2.2.1).
   if($d(x) \neq 1$) return НЕТ;
   }
   return ДА; // Многочлен $f$ неприводим.

**Алгоритм 2.2.10** (тест на неприводимость 2). Для заданного простого числа $p$ и многочлена $f(x) \in \mathbf{F}_p[x]$ степени $k \geqslant 2$, а также различных простых чисел

$q_1 > q_2 > \ldots > q_l$, делящих $k$, алгоритм определяет, является ли многочлен $f(x)$ неприводимым над $\mathbf{F}_p$.

1. [Инициализация]
   $q_{l+1} = 1$;
   $g(x) = x^{p^{k/q_1}} \bmod f(x)$; // Возведение в степень (алгоритм 2.1.5).
2. [Цикл теста]
   for($1 \leqslant i \leqslant l$) {
   $d(x) = \text{НОД}(f(x), g(x) - x)$; // Вычисление НОД (алгоритм 2.2.1).
   if($d(x) \neq 1$) return НЕТ;
   $g(x) = g(x)^{p^{k/q_{i+1}} - p^{k/q_i}} \bmod f(x)$; // Возведение в степень (алгоритм 2.1.5).
   }
3. [Финальное сравнение]
   if($g(x) \neq x$) return НЕТ;
   return ДА; // Многочлен $f$ неприводим..

Алгоритм 2.2.9 основан на простых арифметических процедурах, рассмотренных в настоящей главе, и при больших значениях $k$ уступает алгоритму 2.2.10 по скорости, поскольку первому приходится вычислять гораздо больше значений функции наибольшего общего делителя. Однако привлечение более искусного метода нахождения НОДа многочленов (см. [von zur Gathen and Gerhard 1999, Sec. 11.1]) позволяет почти устранить эту разницу во времени работы.

Кратко сформулируем принципы выполнения операций в конечных полях. Пусть выбран подходящий неприводимый многочлен $f$ степени $k$ над $\mathbf{F}_p$. Каждый элемент $a \in \mathbf{F}_{p^k}$ записывается в виде

$$a = a_0 + a_1 x + a_2 x^2 + \ldots + a_{k-1} x^{k-1},$$

где все $a_i \in \{0, \ldots, p-1\}$. Таким образом, элемент $a$ представляется вектором в пространстве $\mathbf{F}_p^k$. Заметим, что существует всего $p^k$ таких векторов. Здесь сложение совпадает с обычным векторным сложением, но, естественно, арифметические операции по каждой из координат выполняются по модулю $p$. Умножение чуть сложнее: можно рассматривать его просто как умножение многочленов, но помимо выполнения координатных операций по модулю $p$, дополнительно требуется многочлены высоких степеней приводить по модулю $f(x)$. Это означает, что для умножения $a * b$ в $\mathbf{F}_{p^k}$ мы вычисляем произведение многочленов $a(x)b(x)$ в обычном смысле, приводя по модулю $p$ коэффициенты, которые при этом окажутся больше, чем $p-1$. Затем это произведение приводится по модулю $f(x)$ (имеется в виду модулярная операция над многочленами), и снова коэффициенты должны быть приведены по модулю $p$, и так всякий раз, когда в этом возникает необходимость. Можно, в принципе, вычислить произведение $a(x)b(x)$ безо всяких ограничений, затем привести результат по модулю $f$, и окончательно выполнить редукцию по модулю $p$. Результат в этом случае будет тем же самым, однако величина перемножаемых в процессе целых чисел может выйти за рамки, особенно, если перемножается сразу

несколько многочленов. Поэтому рекомендуется приводить коэффициенты по модулю $p$ после каждого значимого действия.

Рассмотрим пример точного построения поля характеристики 2, а именно $\mathbf{F}_{16}$. Согласно нашей формуле (2.5), в $\mathbf{F}_2[x]$ существует ровно 3 неприводимых многочлена степени 4, и легко убедиться, что это многочлены $x^4+x+1$, $x^4+x^3+1$ и $x^4+x^3+x^2+x+1$. Хотя для построения поля $\mathbf{F}_{16}$ может быть выбран любой из них, но первый удобен тем, что выражение высоких степеней $x$ через более низкие выполняется особенно просто. Приведение по модулю $f(x)$ реализуется при помощи простого правила $x^4 = x+1$ (напомним, что мы работаем с полем характеристики 2, поэтому $1 = -1$). Для обозначения элемента поля $a_0+a_1x+a_2x^2+a_3x^3$, где все $a_i \in \{0,1\}$, будем использовать сокращенную запись в виде бинарной строки $(a_0a_1a_2a_3)$. Сложение выполняется покомпонентно по модулю 2, что реализуется элементарными операциями «исключающего ИЛИ», например,

$$(0111) + (1011) = (1100).$$

Чтобы умножить $a * b = (0111) * (1011)$, мы можем смоделировать умножение многочленов, выполнив координатную свертку, что приводит вначале к строке (0110001) длины 7. (Обозначив ее через $(c_0c_1c_2c_3c_4c_5c_6)$, имеем $c_j = \sum_{i_1+i_2=j} a_{i_1}b_{i_2}$, где суммирование производится по парам $i_1, i_2$ целых чисел из $\{0,1,2,3\}$ с суммой $j$.) Чтобы получить окончательный ответ, нужно все единицы на местах 6, 5, 4 (именно в таком порядке) последовательно заменять на нули в соответствии с правилом приведения по модулю $f(x)$. В нашем примере единица на 6-м месте приводит к появлению единиц на местах 2 и 3. После выполнения операции «исключающего ИЛИ», получим (0101000). В старших разрядах единиц больше не осталось, и результат равен (0101). Таким образом,

$$(0111) * (1011) = (0101).$$

Даже такой небольшой пример иллюстрирует все основные принципы реализации арифметических операций в полиномиальном представлении конечных полей.

## 2.3. Квадраты и корни

### 2.3.1. Квадратичные вычеты

Вначале приведем несколько определений.

**Определение 2.3.1.** Пусть целые числа $m$, $a$ взаимно просты, и $m$ положительно. Число $a$ называется квадратичным вычетом $(\mathrm{mod}\ m)$ в том и только том случае, когда сравнение

$$x^2 \equiv a \ (\mathrm{mod}\ m)$$

имеет целочисленное решение для $x$. Если оно не имеет решений, то $a$ называется квадратичным невычетом $(\mathrm{mod}\ m)$.

Отметим, что квадратичные вычеты или невычеты определяются только тогда, когда НОД$(a, m) = 1$. Поэтому, например, вычет 0 (mod $m$) хоть и является всегда квадратом, но не является ни квадратичным вычетом, ни невычетом. Другой пример 3 (mod 9). Этот вычет не является квадратом, однако мы *не* рассматриваем его в качестве квадратичного невычета ввиду того, что числа 3 и 9 не взаимно просты. Для простого модуля единственным не взаимно простым случаем является вычет 0, который, однако, учитывается в следующем определении.

**Определение 2.3.2.** Если $p$ — нечетное простое число, то символ Лежандра $\left(\frac{a}{p}\right)$ определяется как

$$\left(\frac{a}{p}\right) = \begin{cases} 0, & \text{если } a \equiv 0 \pmod{p}, \\ 1, & \text{если } a \text{ — квадратичный вычет } \pmod{p}, \\ -1, & \text{если } a \text{ — квадратичный невычет } \pmod{p}. \end{cases}$$

Таким образом, символ Лежандра показывает, является ли вычет $a \not\equiv 0 \pmod{p}$ квадратом (mod $p$). С символом Лежандра тесно связан, хоть и имеет некоторые существенные отличия, символ Якоби.

**Определение 2.3.3.** Пусть $m$ — нечетное натуральное число (не обязательно простое), тогда для любого целого числа $a$ символ Якоби $\left(\frac{a}{m}\right)$ определяется в терминах (единственного) разложения числа $m$ на простые множители

$$m = \prod p_i^{t_i}$$

как

$$\left(\frac{a}{m}\right) = \prod \left(\frac{a}{p_i}\right)^{t_i},$$

где $\left(\frac{a}{p_i}\right)$ — символы Лежандра. Подразумевается, что $\left(\frac{a}{1}\right) = 1$.

Заметим теперь, что функция $\chi(a) = \left(\frac{a}{m}\right)$, определенная для всех целых $a$, является характером по модулю $m$, см. разд. 1.4.3. Сразу необходимо оговорить, что для составных нечетных чисел $m$ символ Якоби $\left(\frac{a}{m}\right)$ может иногда принимать значение $+1$ и в том случае, когда сравнение $x^2 \equiv a \pmod{m}$ не имеет решений. Например,

$$\left(\frac{2}{15}\right) = \left(\frac{2}{3}\right)\left(\frac{2}{5}\right) = (-1)(-1) = 1,$$

хотя число 2 в действительности не является квадратом по модулю 15. Однако, если $\left(\frac{a}{m}\right) = -1$, то $a$ и $m$ взаимно просты, а сравнение $x^2 \equiv a \pmod{m}$ не имеет решений. Наконец, $\left(\frac{a}{m}\right) = 0$ тогда и только тогда, когда НОД$(a, m) > 1$.

Понятно, что символ $\left(\frac{a}{m}\right)$ можно вычислить. Разлагаем число $m$ на простые множители, затем вычисляем соответствующие символы Лежандра, определяя разрешимость сравнения $x^2 \equiv a \pmod{p}$ путем полного перебора. Однако символы Лежандра и Якоби не применялись бы столь успешно, если бы их нельзя было вычислять совсем просто, без факторизации и тестов на простоту, и, конечно, без полного перебора. В следующей теореме собраны некоторые

замечательные свойства символов Лежандра и Якоби — свойства, превращающие их вычисление в элементарную задачу, сопоставимую по сложности с нахождением НОД.

**Теорема 2.3.4** (соотношения для символов Лежандра и Якоби). *Пусть $p$ обозначает нечетное простое число, $m, n$ — произвольные натуральные нечетные числа (допускаются и простые), а $a, b$ — произвольные целые числа. Тогда справедлив критерий Эйлера квадратичного вычета по простому модулю*

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}, \tag{2.6}$$

*а также мультипликативные законы*

$$\left(\frac{ab}{m}\right) = \left(\frac{a}{m}\right)\left(\frac{b}{m}\right), \tag{2.7}$$

$$\left(\frac{a}{mn}\right) = \left(\frac{a}{m}\right)\left(\frac{a}{n}\right) \tag{2.8}$$

*и специальные соотношения*

$$\left(\frac{-1}{m}\right) = (-1)^{(m-1)/2}, \tag{2.9}$$

$$\left(\frac{2}{m}\right) = (-1)^{(m^2-1)/8}. \tag{2.10}$$

*Кроме того, для взаимно простых $m, n$ выполняется закон квадратичной взаимности:*

$$\left(\frac{m}{n}\right)\left(\frac{n}{m}\right) = (-1)^{(m-1)(n-1)/4}. \tag{2.11}$$

Уже из формулы (2.6) видно, что если $|a| < p$, то для вычисления символа Лежандра $\left(\frac{a}{p}\right)$ достаточно $O\left(\ln^3 p\right)$ битовых операций с использованием стандартной арифметики и алгоритма 2.1.5, см. упр. 2.17. Однако можно получить лучшую оценку, и при этом даже не обязательно производить проверку на простоту.

**Алгоритм 2.3.5** (вычисление символа Лежандра/Якоби). По заданному нечетному целому числу $m$ и целому $a$ алгоритм возвращает символ Якоби $\left(\frac{a}{m}\right)$, который для простых нечетных $m$ совпадает также с символом Лежандра.

1. [Циклы редукции]
   $a = a \bmod m$;
   $t = 1$;
   while($a \neq 0$) {
      while($a$ — четно) {
         $a = a/2$;
         if($m \bmod 8 \in \{3, 5\}$) $t = -t$;
      }
      $(a, m) = (m, a)$; // Перестановка аргументов.
      if($a \equiv m \equiv 3 \pmod 4$) $t = -t$;

$a = a \bmod m;$
}
2. [Завершение работы]
if($m == 1$) return $t$;
return 0;

Очевидно, что этот алгоритм принципиально не более трудоемкий, чем алгоритм 2.1.2 нахождения НОД$(a, m)$, и, как следствие, он требует $O\left(\ln^2 m\right)$ битовых операций в случае $|a| < m$.

В других разделах этой книги мы часто будем опираться на хорошо известную связь между символами Лежандра и тригонометрическими суммами. Этот вопрос изучен достаточно глубоко, но прямо сейчас мы сформулируем только один основной результат, который предварим следующим определением.

**Определение 2.3.6.** Пусть $a, N$ — целые числа, и $N$ положительно. Тогда квадратичная сумма Гаусса $G(a; m)$ определяется как

$$G(a; N) = \sum_{j=0}^{N-1} e^{2\pi\imath a j^2/N}.$$

Эта сумма с точностью до сопряжения совпадает с дискретным преобразованием Фурье (ДПФ), различные модификации которого рассматриваются в гл. 9. Суммы более общего вида — суммы с характерами — применяются для доказательства простоты (см. разд. 4.4). Главный результат, который мы здесь приводим, устанавливает важную взаимосвязь с символами Лежандра.

**Теорема 2.3.7** (Гаусс). *Для нечетного простого $p$ и целого $a \not\equiv 0 \pmod p$ выполняется*

$$G(a; p) = \left(\frac{a}{p}\right) G(1; p),$$

*и вообще, для положительного целого числа $m$ справедливо*

$$G(1; m) = \frac{1}{2}\sqrt{m}(1 + i)(1 + (-\imath)^m).$$

Доказательство первого утверждения совсем несложно. Читатель может проделать его, не обращаясь к ссылкам. Оба утверждения теоремы в совокупности позволяют обратить преобразование Фурье, что позволяет получить выражение для символа Лежандра при $a \not\equiv 0 \pmod p$ в виде

$$\left(\frac{a}{p}\right) = \frac{c}{\sqrt{p}} \sum_{j=0}^{p-1} e^{2\pi\imath a j^2/p} = \frac{c}{\sqrt{p}} \sum_{j=0}^{p-1} \left(\frac{j}{p}\right) e^{2\pi\imath a j/p}, \tag{2.12}$$

где $c = 1, -\imath$ при $p \equiv 1, 3 \pmod 4$ соответственно. Это показывает, что символ Лежандра, по существу, является своим собственным дискретным преобразованием Фурье (ДПФ). Попрактиковаться в работе с гауссовыми суммами помогут упр. 1.66, 2.27, 2.28 и 9.41.

### 2.3.2. Квадратные корни

Вооруженные алгоритмом для вычисления НОД, обратного элемента (т. е. $-1$-й степени) и возведения в целые положительные степени, обратимся к вопросу извлечения квадратного корня по модулю простого числа. Как мы увидим, метод на самом деле потребует возведения остатков в высокие целые степени, и потому эта задача совершенно не похожа на извлечение квадратного корня для действительных чисел.

Мы видели, что для нечетного простого $p$ разрешимость сравнения

$$x^2 \equiv a \not\equiv 0 \pmod{p}$$

определяется значением символа Лежандра $\left(\frac{a}{p}\right)$. Если $\left(\frac{a}{p}\right) = 1$, важной задачей является поиск «квадратных корней» из $x$, которых будет два, каждый из них будет равен другому со знаком «минус» (по модулю $p$). Мы представим два алгоритма извлечения таких квадратных корней; оба алгоритма в вычислительном отношении оптимальны, но при этом они создают различные проблемы при реализации.

Первый алгоритм начинается с критерия Эйлера (2.6). Если простое число $p$ сравнимо с $3 \pmod 4$, и $\left(\frac{a}{p}\right) = 1$, то критерий Эйлера утверждает, что $a^t \equiv 1 \pmod{p}$, где $t = (p-1)/2$. Тогда $a^{t+1} \equiv a \pmod{p}$, и, поскольку $t+1$ в этом случае четно, мы можем положить наш квадратный корень $x \equiv a^{(t+1)/2} \pmod{p}$. Разумеется, это восхитительно простое решение проблемы квадратного корня можно обобщить! Да, но это оказывается не так просто. В общем случае мы можем написать $p-1 = 2^s t$, где $t$ нечетно. Критерий Эйлера (2.6) гарантирует нам, что $a^{2^{s-1}t} \equiv 1 \pmod{p}$, однако не похоже, чтобы он говорил что-либо о величине $A = a^t \pmod{p}$.

Однако на самом деле он кое-что говорит; он утверждает, что мультипликативный порядок элемента $A$ по модулю $p$ является делителем $2^{s-1}$. Допустим, что $d$ является квадратичным *невычетом* по модулю $p$, и пусть $D = d^t \bmod p$. Тогда критерий Эйлера (2.6) утверждает, что мультипликативный порядок элемента $D$ по модулю $p$ в точности равен $2^s$, поскольку $D^{2^{s-1}} \equiv -1 \pmod{p}$. То же самое верно для $D^{-1} \pmod{p}$, а именно, мультипликативный порядок этого элемента равен $2^s$. Поскольку мультипликативная группа $\mathbf{Z}_p^*$ является циклической, то $A$ лежит в циклической подгруппе, порожденной $D^{-1}$, и более того, $A$ является четной степенью $D^{-1}$, т. е. $A \equiv D^{-2\mu} \pmod{p}$ для некоторого целого числа $\mu$, $0 \leqslant \mu < 2^{s-1}$. Подставляя это значение в качестве $A$, получаем $a^t D^{2\mu} \equiv 1 \pmod{p}$. Тогда, если умножить это сравнение на $a$, показатели всех степеней в левой части сравнения окажутся четными, и можно будет получить квадратный корень из $a$ по модулю $p$ в виде $a^{(t+1)/2} D^{\mu} \pmod{p}$.

Чтобы представить эту идею в виде алгоритма, нужно решить две проблемы:

(1) найти квадратичный невычет $d \pmod{p}$;
(2) найти целое число $\mu$, такое что $A \equiv D^{-2\mu} \pmod{p}$.

Может показаться, что задача (1) простая, а (2) — сложная, поскольку квадратичных невычетов по модулю $p$ много, а нам нужен только один из них,

причем любой; в то время как для задачи (2) необходимо найти определенное число $\mu$. В некотором смысле это рассуждение верно. Однако нам неизвестно ни одного строгого, *детерминированного* метода для быстрого поиска квадратного невычета. Мы обойдем эту проблему, используя *случайный* алгоритм. И хотя задача (2) является примером известной сложной задачи дискретного логарифмирования (см. гл. 5), именно в нашей постановке она оказывается простой.

Следующий алгоритм создан в 1891 г. А. Тонелли на основе более ранней работы Гаусса.

**Алгоритм 2.3.8** (квадратный корень по модулю $p$). Пусть заданы нечетное простое число $p$ и целое число $a$, такое что $\left(\frac{a}{p}\right) = 1$. Этот алгоритм возвращает решение $x$ сравнения $x^2 \equiv a \pmod p$.

1. [Простейшие случаи: $p \equiv 3, 5, 7 \pmod 8$]
   $a = a \bmod p$;
   if($p \equiv 3, 7 \pmod 8$) {
      $x = a^{(p+1)/4} \bmod p$;
      return $x$;
   }
   if($p \equiv 5 \pmod 8$) {
      $x = a^{(p+3)/8} \bmod p$;
      $c = x^2 \bmod p$; // Тогда $c \equiv \pm a \pmod p$.
      if($c \neq a \bmod p$) $x = x2^{(p-1)/4} \bmod p$;
      return $x$,
   }
2. [Случай $p \equiv 1 \pmod 8$]
   Найти случайный квадратичный невычет $d \pmod p$), используя алгоритм 2.3.5;
   Найти случайное целое $d \in [2, p-1]$, $\left(\frac{d}{p}\right) = -1$;
   // Вычислить символ Якоби (алгоритм 2.3.5).
   Представить $p - 1 = 2^s t$, где $t$ нечетно;
   $A = a^t \bmod p$;
   $D = d^t \bmod p$;
   $m = 0$; // $m$ представляет собой $2\mu$.
   for($0 \leqslant i < s$){ // Можно начать с $i = 1$ (см. текст).
      if($(AD^m)^{2^{s-1-i}} \equiv -1 \pmod p$) $m = m + 2^i$;
   } // Таким образом, $AD^m \equiv 1 \pmod p$.
   $x = a^{(t+1)/2} D^{m/2} \bmod p$;
   return $x$;

Отметим следующие интересные свойства алгоритма. Во-первых, оказывается, что случай $p \equiv 1 \pmod 8$ — самый сложный случай — на самом деле включает в себя все остальные. (Мы существенно использовали в случае $p \equiv 5 \pmod 8$, что можно выбрать $d = 2$. Для случаев $p \equiv 3 \pmod 4$ показатель $m$

равен 0, так что величина $d$ не нужна.) Во-вторых, обратите внимание: в алгоритм встроена проверка $A^{2^{s-1}} \equiv 1 \pmod{p}$, которая и гарантирует нам, что $m$ четно. Если бы это не было выполнено, то было бы не верно, что $\left(\frac{a}{p}\right) = 1$; поэтому алгоритм можно исправить, чтобы он не содержал этого требования, с прекращением цикла, если случай $i = 0$ в цикле приводит к остатку $-1$. Если требуется извлечь много квадратных корней из вычетов $a$, о которых неизвестно, являются ли они квадратичными вычетами или невычетами, то можно было бы попытаться дать алгоритму 2.3.8 решить этот вопрос за нас. Однако, если квадратичные невычеты появляются хоть сколько-нибудь часто, в среднем было бы быстрее сначала выполнить алгоритм 2.3.5, чтобы проверить квадратичную природу вычета $a$, и тем самым избежать выполнения более дорогого алгоритма 2.3.8 для невычетов.

Как мы упомянули, не известен ни один детерминистический алгоритм для поиска квадратичного невычета $d$ для заданного простого числа $p$, работающий за полиномиальное время. Однако, если предположить, что верна расширенная гипотеза Римана (extended Riemann hypothesis, ERH), то можно доказать существование квадратичного невычета $d < 2\ln^2 p$, см. теорему 1.4.5; и потому полный перебор до этой границы позволяет найти квадратичный невычет за полиномиальное время. Таким образом, если ERH предполагается верной, существует детерминированный способ вычисления квадратного корня по модулю простого числа $p$ за полиномиальное время. С теоретической точки зрения любопытно, что для *фиксированного* числа $a$ имеется строго доказанный детерминированный алгоритм Шуфа [Schoof 1985]. (Битовая сложность полиномиальна относительно длины $p$, но экспоненциальна относительно длины $a$, поэтому для фиксированных значений $a$ допустимо говорить, что алгоритм работает за полиномиальное время.) Тем не менее, несмотря на такое необычное состояние дел, тот факт, что половина всех ненулевых невычетов $d \pmod{p}$ удовлетворяет равенству $\left(\frac{d}{p}\right) = -1$, наталкивает на мысль, что хватит малого количества случайных попыток, чтобы найти подходящее $d$. В самом деле, ожидаемое число случайных попыток равно 2.

Сложность алгоритма 2.3.8 определяется большим количеством необходимых возведений в степень, и потому составляет $O(s^2 + \ln t)$ модулярных операций. В случае наивной реализации арифметических подпрограмм оказывается, что эта оценка составляет в наихудшем случае (когда $s$ велико) $O\left(\ln^4 p\right)$ битовых операций. Однако, если алгоритм 2.3.8 применяется ко многим простым модулям $p$, видимо, лучше рассмотреть усредненный случай, что означает лишь $O\left(\ln^3 p\right)$ битовых операций. Это так, поскольку имеется очень мало простых чисел $p$, у которых $p - 1$ делится на большую степень 2.

Следующий алгоритм работает асимптотически быстрее, чем алгоритм 2.3.8 в наихудшем случае. Этот метод, созданный в 1907 г. М. Чиполлой, является красивым применением арифметики конечных полей $\mathbf{F}_{p^2}$.

**Алгоритм 2.3.9** (вычисление квадратного корня по модулю $p$ при помощи поля $\mathbf{F}_{p^2}$). Пусть заданы простое нечетное число $p$ и квадратичный вычет $a$ по модулю $p$. Этот алгоритм возвращает решение $x$ сравнения $x^2 \equiv a \pmod{p}$.

1. [Найти квадратичный невычет]
   Найти случайное целое $t \in [0, p-1]$, такое, что $\left(\frac{t^2-a}{p}\right) = -1$;
   // Вычислить символ Якоби (алгоритм 2.3.5).
2. [Найти квадратный корень в поле $\mathbf{F}_{p^2} = \mathbf{F}_p(\sqrt{t^2-a})$ ]
   $x = (t + \sqrt{t^2-a})^{(p+1)/2}$; // В поле $\mathbf{F}_{p^2}$.
   return $x$;

Вероятность того, что случайно выбранная в шаге [Найти квадратичный невычет] величина $t$ будет правильной, равна $(p-1)/2p$. Несложно показать, что элемент $x \in \mathbf{F}_{p^2}$ является на самом деле элементом подполя $\mathbf{F}_p$ поля $\mathbf{F}_{p^2}$, и что $x^2 \equiv a \pmod p$. (В самом деле, из второго утверждения следует, что $x$ лежит в $\mathbf{F}_p$, поскольку $a$ имеет те же квадратные корни в $\mathbf{F}_p$, что и в большем поле $\mathbf{F}_{p^2}$.)

Следует поговорить об арифметике этого поля, которая для случая $\mathbf{F}_{p^2}$ существенно проста, как можно ожидать согласно разд. 2.2.2. Пусть $\omega = \sqrt{t^2-a}$. Используя представление этого поля в виде

$$\mathbf{F}_{p^2} = \{x + \omega y : x, y \in \mathbf{F}_p\} = \{(x, y)\},$$

можно далее проводить все вычисления с помощью правила

$$\begin{aligned}(x,y) * (u,v) &= (x + y\omega)(u + v\omega) \\ &= xu + yv\omega^2 + (xv + yu)\omega \\ &= (xu + yv(t^2-a), xv + yu),\end{aligned}$$

заметив, что $\omega^2 = t^2 - a$ лежит в $\mathbf{F}_p$. Конечно, мы рассматриваем $x, y, u, v, t, a$ как остатки по модулю $p$, и потому все выражения выше всегда приводятся по этому модулю. Любая двоичная схема возведения в степень в этой книге годится для вычисления $x$ в этапе [Найти квадратный корень ...]. Равносильный алгоритм для вычисления квадратного корня приводится в [Menezes et al. 1997], где находится квадратичный невычет $b^2 - 4a$, определяется многочлен $f(x) = x^2 - bx + a$ из $\mathbf{F}_p[x]$, и затем просто находится требуемый корень $r = x^{(p+1)/2} \bmod f$ (используя модулярные операции над многочленами). Заметим, в заключение, что для увеличения средней производительности алгоритмов в любом из них можно исключить особые случаи $p \equiv 3, 5, 7 \pmod 8$, как это было сделано в алгоритме 2.3.8

Сложность алгоритма 2.3.9 составляет $O(\ln^3 p)$ битовых операций (в предположении о наивной реализации операций), что асимптотически лучше, чем сложность алгоритма 2.3.8 в наихудшем случае. Однако, если не желательно реализовывать модифицированную схему возведения в степень для вычислений в поле $\mathbf{F}_{p^2}$, то асимптотически более медленного метода обычно хватает. К счастью, имеется другой, равносильный, метод извлечения корней с использованием последовательностей Люка (см. упр. 2.31).

Очень интересно отметить, что не известно ни одного быстрого метода извлечения квадратного корня из квадратичного вычета для произвольного составного модуля. В самом деле, как мы увидим далее, такое извлечение по сути равносильно факторизации модуля (см. упр. 6.5).

### 2.3.3. Поиск корней многочленов

Обсудив вопросы существования и вычисления квадратных корней, мы теперь рассматриваем задачу вычисления корней многочлена произвольной степени над конечным полем. Под полем мы понимаем $\mathbf{F}_p$, но многое из того, что мы излагаем, обобщается на случай произвольного конечного поля.

Пусть $g \in \mathbf{F}_p[x]$ — многочлен, т. е. многочлен с целыми коэффициентами, приведенными по модулю $p$. Мы ищем корни $g$ в поле $\mathbf{F}_p$, и потому мы могли бы начать с замены $g(x)$ на НОД многочленов $g(x)$ и $x^p - x$, поскольку, как мы видели, последний многочлен является произведением одночленов $x - a$, когда $a$ пробегает значения всех элементов поля $\mathbf{F}_p$. Если $p > \deg g$, следует сначала вычислить $x^p \bmod g(x)$ с помощью алгоритма 2.1.5. Если степень НОД не превосходит 2, то ранее изученные нами методы решают задачу. Если же его степень больше 2, то мы переходим к следующему НОД с многочленом $(x+a)^{(p-1)/2}-1$ для случайного $a \in \mathbf{F}_p$. Любое конкретное $b \neq 0$ из $\mathbf{F}_p$ является корнем многочлена $(x+a)^{(p-1)/2} - 1$ с вероятностью $1/2$, так что у нас имеется положительная вероятность расщепления многочлена $g(x)$ на два многочлена меньшей степени. Это приводит к следующему рекурсивному алгоритму.

**Алгоритм 2.3.10** (корни многочлена над полем $\mathbf{F}_p$). Пусть задан ненулевой многочлен $g \in \mathbf{F}_p[x]$, где $p$ — нечетное простое число. Этот алгоритм возвращает множество $r$ корней (без учета кратности) многочлена $g$ в поле $\mathbf{F}_p$. Предполагается, что множество $r$ — глобальное, изменяемое как указано ниже в течение всех рекурсивных вызовов.

1. [Начальные установки]
   $r = \{\ \}$; // Корневой список изначально пуст.
   $g(x) = \text{НОД}(x^p - x, g(x))$; // Используя алгоритм 2.2.1.
   if($g(0) == 0$) { // Проверить, корень ли 0.
   $r = r \cup \{0\}$;
   $g(x) = g(x)/x$;
   }
2. [Рекурсивный вызов и возврат]
   $r = r \cup roots(g)$;
   return $r$;
3. [Рекурсивная функция $roots()$]
   $roots(g)$ {
   Если $\deg(g) \leqslant 2$, использовать квадратичную формулу (или более низкого порядка), согласно алгоритму 2.3.8 или 2.3.9, чтобы добавить ко множеству $r$ все корни многочлена $g$, и вернуться;
   while($h == 1$ or $h == g$) { // Случайное разбиение.
   Выбрать случайное $a \in [0, p-1]$;
   $h(x) = \text{НОД}((x+a)^{(p-1)/2} - 1, g(x))$;
   }
   $r = r \cup roots(h) \cup roots(g/h)$;
   return;
   }

Вычисление $h(x)$ в цикле случайного разделения можно сделать проще, используя алгоритм 2.1.5 для первоначального вычисления $(x+a)^{(p-1)/2} \bmod g(x)$ (коэффициенты, конечно, всегда приводятся по модулю $p$). Можно показать, что вероятность того, что случайное $a$ успешно расщепит $g(x)$ (где $\deg(g) \geqslant 3$) больше примерно $3/4$ при больших $p$, и всегда оценивается числом, большим 0. Заметим, что мы можем использовать идею случайного разделения для многочленов степени 2, и потому у нас уже три алгоритма извлечения квадратного корня! (Если $g(x)$ имеет степень 2, то вероятность того, что случайный выбор $a$ в шаге [Рекурсивная функция *roots*()] разделит многочлен $g(x)$, не менее $(p-1)/(2p)$.) Различные подробности реализации этого алгоритма обсуждаются в книге Коэна [Cohen 2000]. Отметим, что этот алгоритм не разлагает на самом деле многочлен на сомножители; например, многочлен $f$ может быть произведением двух несократимых многочленов без корней в поле $\mathbf{F}_p$. Для собственно разложения на множители имеется алгоритм Берлекэмпа (см. [Menezes et al. 1997], [Cohen 2000]), однако многие важные алгоритмы требуют только того поиска корней, который мы осуществили.

Теперь обсудим задачу поиска корней многочлена по составному модулю. Предположим, что модуль $n = ab$, где числа $a$ и $b$ взаимно просты. Если существует целое число $r$, удовлетворяющее $f(r) \equiv 0 \pmod{a}$, и целое число $s$ такое, что $f(s) \equiv 0 \pmod{b}$, то мы можем найти решение сравнения $f(x) \equiv 0 \pmod{ab}$, «соответствующее» числам $r$ и $s$. А именно, если целое число $t$ одновременно удовлетворяет сравнениям $t \equiv r \pmod{a}$ и $t \equiv s \pmod{b}$, то $f(t) \equiv 0 \pmod{ab}$. Такое число $t$ можно найти с помощью китайской теоремы об остатках; см. теорему 2.1.6. Таким образом, если модуль $n$ можно разложить на произведение взаимно простых сомножителей, и если удастся решить задачу для модулей, являющихся степенями простых чисел, то мы также сможем решить задачу в общем случае.

С этой целью мы остановимся на решении сравнений многочленов по модулю степеней простых чисел. Заметим, что для любого многочлена $f(x) \in \mathbf{Z}[x]$ и любого целого числа $r$ имеется многочлен $g_r(x) \in \mathbf{Z}[x]$ такой, что

$$f(x+r) = f(r) + xf'(r) + x^2 g_r(x). \tag{2.13}$$

Это можно увидеть либо рассматривая разложение Тейлора для многочлена $f(x+r)$, либо воспользовавшись биномом Ньютона в виде

$$(x+r)^d = r^d + dr^{d-1}x + x^2 \sum_{j=2}^{d} \binom{d}{j} r^{d-j} x^{j-2}.$$

Мы можем использовать алгоритм 2.3.10 для поиска решений сравнения $f(x) \equiv 0 \pmod{p}$, если они существуют. Вопрос заключается в том, как перейти к решению для модуля $p^k$ при различных значениях $k$. Допустим, мы успешно нашли решение сравнения по модулю $p^i$, т. е. $f(r) \equiv 0 \pmod{p^i}$, и хотим найти решение сравнения $f(t) \equiv 0 \pmod{p^{i+1}}$ такое, что $t \equiv r \pmod{p^i}$. Запишем $t$ в виде $r + p^i y$, чтобы решать сравнение теперь относительно $y$. В равенстве

(2.13) положим $x = p^i y$. Таким образом,

$$f(t) = f(r + p^i y) \equiv f(r) + p^i y f'(r) \pmod{p^{2i}}.$$

Если целое число $f'(r)$ не делится на $p$, то для решения уравнения

$$f(r) + p^i y f'(r) \equiv 0 \pmod{p^{2i}}$$

можно использовать методы разд. 2.1.1, а именно, поделить обе части равенства на $p^i$ (напомним, что $f(r)$ делится на $p^i$), найти величину, обратную к $f'(r) \pmod{p^i}$, и положить $y = -zf(r)p^{-i} \bmod p^i$. Таким образом, мы сделали больше, чем требовалось, мгновенно перейдя от модуля $p^i$ к модулю $p^{2i}$. Однако было добавлено требование, чтобы величина $r$ удовлетворяла требованию $f'(r) \not\equiv 0 \pmod{p}$. В общем случае, если $f(r) \equiv f'(r) \equiv 0 \pmod{p}$, то не может существовать целых чисел $t \equiv r \pmod{p}$ таких, что $f(t) \equiv 0 \pmod{p^2}$. Например, пусть $f(x) = x^2 + 3$ и простое число $p = 3$. Имеется корень $x = 0$, так как $f(0) \equiv 0 \pmod{3}$. Однако сравнение $f(x) \equiv 0 \pmod{9}$ не имеет решений. Дополнительные критерии того, когда корень многочлена может быть поднят до больших степеней модуля, приведены в разд. 3.5.3 книги Коэна [Cohen 2000].

Описанный метод называется поднятием Гензеля в честь немецкого математика К. Гензеля. Доказательство, по сути, дает критерий существования решения уравнения $f(x) = 0$ в «$p$-адических» числах: решение существует, если имеется целое число $r$, удовлетворяющее условиям $f(r) \equiv 0 \pmod{p}$ и $f'(r) \not\equiv 0 \pmod{p}$. Для нас важнее, однако, использвание этой идеи в качестве алгоритма решения полиномиальных сравнений по модулю степеней простых чисел. Подведем итог обсуждения следующим алгоритмом.

**Алгоритм 2.3.11** (поднятие Гензеля). Пусть заданы многочлен $f(x) \in \mathbf{Z}[x]$, простое число $p$ и целое число $r$, удовлетворяющее $f(r) \equiv 0 \pmod{p}$ (возможно, полученное с помощью алгоритма 2.3.10), и пусть $f'(r) \not\equiv 0 \pmod{p}$. Этот алгоритм описывает, как построить последовательность целых чисел $r_0, r_1, \ldots$, таких, что для всех $i < j$ справедливы сравнения $r_i \equiv r_j \pmod{p^{2^i}}$ и $f(r_i) \equiv 0 \pmod{p^{2^i}}$. Ответ строится последовательно, т. е. мы устанавливаем $r_0$ и показываем, как получить $r_{i+1}$ как функцию уже известной величины $r_i$.

1. [Начальная установка]
   $r_0 = r$;
2. [Функция $newr()$, вычисляющая $r_{i+1}$ по $r_i$]
   $newr(r_i)$ {
   $x = f(r_i)p^{-2^i}$;
   $z = (f'(r))^{-1} \bmod p^{2^i}$; // С помощью алгоритма 2.1.4.
   $y = -xz \bmod p^{2^i}$;
   $r_{i+1} = r_i + yp^{2^i}$;
   return $r_{i+1}$;
   }

Отметим, что для $j \geqslant i$ имеем $r_j \equiv r_i \pmod{p^{2^i}}$, так что последовательность $(r_i)$ сходится в $p$-адических числах к корню $f(x)$. На самом деле подня-

тие Гензеля может рассматриваться как $p$-адическая версия методов Ньютона, рассматриваемых в разд. 9.2.2.

### 2.3.4. Представление числа квадратичной формой

Обратимся теперь к задаче, важной для приложений, таких как эллиптические кривые и проверка простоты. Это задача поиска квадратичных диофантовых представлений для положительного целого $d$ и нечетного простого $p$ в виде

$$x^2 + dy^2 = p,$$

или при исследовании комплексных квадратичных порядков отрицательного дискриминанта $D \equiv 0, 1 \pmod 4$ (равенство из книги [Cohen 2000]),

$$x^2 + |D|y^2 = 4p.$$

Имеется красивый подход к этим целочисленным задачам. Следующие два алгоритма не только элегантны, но и весьма эффективны. Между прочим, следующий алгоритм обычно приписывался Корнаккье до тех пор, пока недавно не стало известно, что Смит нашел его раньше, а именно, в 1885 г.

**Алгоритм 2.3.12** (представить $p$ в виде $x^2 + dy^2$: метод Корнаккьи—Смита). Пусть задано нечетное простое число $p$ и целое $d$, не кратное $p$. Этот алгоритм либо сообщает, что уравнение $p = x^2 + dy^2$ целочисленного решения не имеет, либо возвращает решение $(x, y)$.

1. [Проверка разрешимости]
   if$(\left(\frac{-d}{p}\right) == -1)$ return { }; // Пустой возврат: нет решений.
2. [Начальный квадратный корень]
   $x_0 = \sqrt{-d} \bmod p$; // С помощью алгоритма 2.3.8 или 2.3.9.
   if$(2x_0 < p)$ $x_0 = p - x_0$; // Нормировать корень.
3. [Начало алгоритма Евклида]
   $(a, b) = (p, x_0)$;
   $c = \lfloor \sqrt{p} \rfloor$; // С помощью алгоритма 9.2.11.
4. [Алгоритм Евклида]
   while$(b > c)$ $(a, b) = (b, a \bmod b)$;
5. [Окончательный ответ]
   $t = p - b^2$;
   if$(t \not\equiv 0 \pmod d)$ return { }; // Пустой возврат.
   if$(t/d$ not a square$)$ return { }; // Пустой возврат.
   return $(\pm b, \pm\sqrt{t/d})$; // Решение найдено.

Этот алгоритм полностью решает рассматриваемую вычислительную диофантову задачу. Отметим, что процедура поиска целочисленных квадратных корней (алгоритм 9.2.11) вызывается дважды. Второй вызов — определение того, является ли $t/d$ полным квадратом — может быть проведен в общем так же, как в тексте обсуждения после алгоритма 9.2.11. Кстати, доказательство того, что алгоритм 2.3.12 работает корректно, по словам Коэна (см. [Cohen

2000]) «доставляет легкую головную боль». Элегантное доказательство, принадлежащее Ленстре, приведено в статье [Schoof 1995], а четкое с точки зрения алгоритмиста объяснение (для $d = 1$) — в работе [Bressoud and Wagon 2000, p. 283].

Второй случай, а именно случай диофантова уравнения $x^2 + |D|y^2 = 4p$ для $D < 0$, может быть решен следующим способом (см. [Cohen 2000]). Во-первых, заметим, что если $D \equiv 0 \pmod 4$, то $x$ четно, и тогда проблема сводится к решению уравнения $(x/2)^2 + (|D|/4)y^2 = p$, что мы только что сделали. Если $D \equiv 1 \pmod 8$, имеем $x^2 - y^2 \equiv 4 \pmod 8$, и потому $x, y$ оба являются четными, и мы снова свели задачу к предыдущей. С учетом этого можно было бы использовать следующий алгоритм только для $D \equiv 5 \pmod 8$, однако на самом деле он будет работать для случаев $D \equiv 0, 1 \pmod 4$, которые как окажется, будут удобными.

**Алгоритм 2.3.13** (представить $4p$ в виде $x^2 + |D|y^2$ (измененный метод Корнаккьи—Смита)). Пусть задано простое число $p$ и отрицательное $-4p < D < 0$, где $D \equiv 0, 1 \pmod 4$. Этот алгоритм либо сообщает, что решения не существует, либо возвращает решение $(x, y)$.

1. [Случай $p = 2$]
   if($p == 2$) {
   if($D + 8$ является квадратом) return $(\sqrt{D+8}, 1)$;
   return { }; // Пустой возврат: решений нет.
   }
2. [Проверка разрешимости]
   if($(\frac{D}{p}) < 1$) return { }; // Пустой возврат.
3. [Начальный квадратный корень]
   $x_0 = \sqrt{D} \bmod p$; // С помощью алгоритма 2.3.8 или 2.3.9.
   if($x_0 \not\equiv D \pmod 2$) $x_0 = p - x_0$; // Удостовериться, что $x_0^2 \equiv D \pmod{4p}$.
4. [Начало алгоритма Евклида]
   $(a, b) = (2p, x_0)$;
   $c = \lfloor 2\sqrt{p} \rfloor$; // С помощью алгоритма 9.2.11.
5. [Алгоритм Евклида]
   while($b > c$) $(a, b) = (b, a \bmod b)$;
6. [Окончательный ответ]
   $t = 4p - b^2$;
   if($t \not\equiv 0 \pmod{|D|}$) return { }; // Пустой возврат.
   if($t/|D|$ not a square) return { }; // Пустой возврат.
   return $(\pm b, \pm\sqrt{t/|D|})$; // Решение найдено.

Опять же, алгоритм либо сообщает о том, что решения не существует, или находит по существу единственное решение уравнения $x^2 + |D|y^2 = 4p$.

## 2.4. Упражнения

**2.1.** Докажите, что число 16 является восьмой степенью некоторого числа по любому нечетному модулю.

**2.2.** Покажите, что наименьшее общее кратное НОК $(a, b)$ равно

$$\text{НОК}\,(a,b) = \frac{ab}{\text{НОД}\,(a,b)},$$

и обобщите эту формулу для случая более чем двух аргументов. Затем, используя теорему о количестве простых чисел, найдите разумную оценку для НОК всех простых чисел от 1 до $n$.

**2.3.** Напомним, что $\omega(n)$ обозначает количество различных простых делителей числа $n$. Докажите, что для любого бесквадратного натурального $n$ справедлива формула

$$\#\{(x,y)\ :\ x,y \text{ — натуральные числа, НОК}\,(x,y) = n\} = 3^{\omega(n)}.$$

**2.4.** Изучите связь между алгоритмом Евклида и простыми цепными дробями с тем, чтобы доказать теорему Ламе (первую часть теоремы 2.1.3).

**2.5.** Числа Фибоначчи определяются равенствами $u_0 = 0$, $u_1 = 1$ и $u_{n+1} = u_n + u_{n-1}$ для $n \geqslant 1$. Докажите знаменитое соотношение

$$\text{НОД}\,(u_a, u_b) = u_{\text{НОД}(a,b)},$$

показывающее, помимо прочего, что $u_n, u_{n+1}$ взаимно просты для $n > 1$, и что если $u_n$ простое, то $n$ также простое. Найдите контрпример для обратного утверждения (найдите простое $p$, такое что $u_p$ составное). Рассматривая несколько чисел Фибоначчи, угадайте — и затем докажите — простую общую формулу для обратного элемента для $u_n \pmod{u_{n+1}}$.

Числа Фибоначчи многократно появляются повсюду в этой книге, например, в разд. 1.3.3, 3.6.1 и упр. 3.25, 3.41, 9.50.

**2.6.** Покажите, что для чисел $x \approx y \approx N$, в предположении классического деления с остатком, битовая сложность классического алгоритма Евклида равна $O\left(\ln^2 N\right)$. Полезно заметить, что поиск пары частное—остаток $q$ и $r$, таких что $x = qy + r$, требует $O((1 + \ln q)\ln x)$ битовых операций, и что на частное налагается определенное ограничение в ходе цикла алгоритма Евклида.

**2.7.** Докажите, что алгоритм 2.1.4 работает, т. е. что возвращается правильная пара из НОД и обратной величины. Ответьте на следующий вопрос: когда нужно (и вообще, нужно ли) пару возврата $a, b$ понижать далее, до $a \bmod y$ и $b \bmod x$, чтобы она задавала допустимые единственные обратные величины?

**2.8.** Докажите, что при наивном применении теоремы 2.1.6 операции mod поглощают по меньшей мере $O\left(\ln^2 M\right)$ битовых операций, если операции выполняются школьным методом, но только $O\left(r \ln^2 m\right)$, если использовать алгоритм 2.1.7, где $m$ обозначает наибольшее из $m_i$.

**2.9.** Напишите программу, реализующую асимптотически быстрый алгоритм 9.5.26 для китайской теоремы об остатках с предварительными вычислениями, и используйте его, чтобы умножить два числа длины, скажем, 100

десятичных знаков, используя достаточно большое число маленьких простых модулей.

**2.10.** В соответствии с упр. 1.48, в качестве модулей китайской теоремы об остатках можно использовать числа Мерсенна, имеющие попарно взаимно простые показатели (сами числа Мерсенна не обязаны быть простыми). Какие вычислительные преимущества можно получить, выбрав такой набор модулей (см. разд. 9.2.3)? Есть ли простой способ найти обратные величины $(2^a-1)^{-1} \pmod{2^b-1}$?

**2.11.** Определите вычислительную сложность «прямолинейного» алгоритма для обратной величины, определяемого соотношением (2.3). Имеется ли хотя бы когда-нибудь ситуация, когда следует его использовать, или лучше использовать алгоритм 2.1.4 для получения $a^{-1} \bmod m$?

**2.12.** Обозначим количество неприводимых многочленов из $\mathbf{F}_p[x]$ степени $k$ со старшим коэффициентом 1 через $N_k(p)$. При помощи формулы (2.5) докажите для каждого простого числа $p$ и натурального $k$ неравенство $p^k/k \geqslant N_k(p) > p^k/k - 2p^{k/2}/k$. Докажите также, что $N_k(p)$ всегда положительно.

**2.13.** Можно ли обобщить формулу (2.5), чтобы получить число неприводимых многочленов степени $k$ в кольце $\mathbf{F}_{p^n}[x]$?

**2.14.** Покажите, как алгоритм 2.2.2 исполняет свою роль в вычислениях в конечных полях, а именно в процессе поиска мультипликативного обратного элемента в поле $\mathbf{F}_{p^n}$.

**2.15.** Докажите теорему 2.2.8.

**2.16.** Покажите, как можно правильно обобщить алгоритмы 2.3.8 и 2.3.9 для поиска квадратных корней из квадратов в конечном поле $\mathbf{F}_{p^n}$.

**2.17.** Рассматривая двоичное представление показателя $n$, покажите, что вычислительная сложность алгоритма 2.1.5 составляет $O(\ln n)$ операций. Докажите, что если $x, n$ не превосходят $m$, и нам необходимо найти $x^n \bmod m$, и используются классические операции умножения и приведения по модулю, то общая битовая сложность такого возведения в степень растет как куб числа битов в $m$.

**2.18.** Пусть мы хотим найти степень $x^y \bmod N$, где $N = pq$ — произведение двух разных простых чисел. Опишите алгоритм, сочетающий идеи двоичной схемы возведения в степень и китайской теоремы об остатках, приводящий к желаемым степеням быстрее, чем позволяет стандартная схема возведения в степень, работающая по модулю $N$.

**2.19.** Известно, что число «периодическая единица» $r_{1031} = (10^{1031}-1)/9$, составленное из 1031 десятичных единиц, простое. Определите с использованием обратной величины, какое из чисел 7, $-7$ является квадратичным вычетом по модулю периодической единицы. Затем найдите явный квадратный корень по модулю $r_{1031}$ из квадратичного вычета.

**2.20.** Используя соответствующие результаты теоремы 2.3.4, докажите, что

для простого $p > 3$

$$\left(\frac{-3}{p}\right) = (-1)^{\frac{(p-1) \bmod 6}{4}}.$$

Найдите похожее выражение для $\left(\frac{5}{p}\right)$, если $p \neq 2, 5$.

**2.21.** Покажите, что для простого числа $p \equiv 1 \pmod 4$ сумма квадратичных вычетов из отрезка $[1, p-1]$ составляет $p(p-1)/4$.

**2.22.** Докажите, что если натуральное число $a$ не является точным квадратом, то $\left(\frac{a}{p}\right) = -1$ для бесконечного множества простых $p$. (Подсказка: пусть сначала число $a$ лишь нечетно. Покажите, что найдется такое целое число $b$, что $\left(\frac{b}{a}\right) = -1$ и $b \equiv 1 \pmod 4$. Тогда любое натуральное число $n \equiv b \pmod{4a}$ удовлетворяет условию $\left(\frac{a}{n}\right) = -1$ и поэтому кратно некоторому простому числу $p$, для которого $\left(\frac{a}{p}\right) = -1$. Покажите, как таким образом возникает бесконечно много простых чисел $p$. Затем рассмотрите случай четного или отрицательного нечетного числа $a$.)

**2.23.** При помощи упр. 2.22 докажите, что если квадратный трехчлен $f(x)$ из $\mathbf{Z}[x]$ неприводим, то $f(x) \bmod p$ неприводим над $\mathbf{Z}_p[x]$ для бесконечного множества простых чисел $p$. Покажите, что многочлен $x^4 + 1$ неприводим над $\mathbf{Z}[x]$, однако он приводим над любым $\mathbf{Z}_p[x]$. А что будет для многочлена третьей степени?

**2.24.** Постройте алгоритм вычисления символа Якоби $\left(\frac{a}{m}\right)$ по аналогии с алгоритмом 9.4.2 — двоичным алгоритмом вычисления наибольшего общего делителя.

**2.25.** Пусть $p$ — простое, такое что $p \equiv 3 \pmod 4$, и пусть задана пара квадратных корней из любого числа $x \not\equiv 0 \pmod p$. Докажите, что один из корней является квадратичным вычетом, а другой является невычетом. (Корень, являющийся квадратичным вычетом, называется главным значением корня.) См. упр. 2.26 и 2.42 по поводу применений главного значения корня.

**2.26.** Обозначим через $\mathbf{Z}_n^*$ мультипликативную группу из тех элементов $\mathbf{Z}_n$, которые взаимно просты с $n$.

(1) Пусть $n$ — нечетное и имеет $k$ различных простых сомножителей. Пусть $J$ означает множество элементов $x \in \mathbf{Z}_n^*$, символ Якоби которых $\left(\frac{x}{n}\right) = 1$, и пусть $S$ означает множество полных квадратов в $\mathbf{Z}_n^*$. Покажите, что $J$ является подгруппой в $\mathbf{Z}_n^*$, содержащей $\varphi(n)/2$ элементов, и что $S$ является подгруппой $J$.

(2) Покажите, что квадраты в $\mathbf{Z}_n^*$ имеют ровно $2^k$ квадратных корней в $\mathbf{Z}_n^*$, и тем самым докажите, что $\#S = \varphi(n)/2^k$.

(3) Пусть теперь $n$ является целым числом Блюма, т. е. $n = pq$, где $p, q$ — различные простые числа, $p, q \equiv 3 \pmod 4$. (Числа Блюма важны в криптографии (см. [Menezes et al. 1997], а также разд. 8.2 этой книги).) Из пунктов (1) и (2) вытекает, что $\#S = \frac{1}{2}\#J$, так что половина элементов $J$ являются квадратами, а половина — нет. Согласно пункту (2), элемент $S$ имеет ровно 4 квадратных корня. Покажите, что ровно один из этих корней сам принадлежит $S$.

(4) Для числа Блюма $n = pq$ покажите, что функция возведения в квадрат $s(x) = x^2 \bmod n$ является перестановкой подгруппы $S$, и что обратная к ней функция имеет вид

$$s^{-1}(y) = y^{((p-1)(q-1)+4)/8} \bmod n.$$

**2.27.** Используя теорему 2.3.7 докажите два равенства в соотношениях (2.12).

**2.28.** Здесь мы докажем известный квадратичный закон взаимности (2.11) для двух различных простых чисел $p, q$. Используя определение 2.3.6, покажите, что функция $G$ мультипликативна, т. е., если НОД$(m, n) = 1$, то

$$G(m; n)G(n; m) = G(1; mn).$$

(Указание: сумма $mj^2/n + nk^2/m$ похожа — в определенном смысле — на $(mj + nk)^2/(mn)$.) Выведите из этого равенства и теоремы 2.3.7 соотношение для простых чисел $p, q$

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}.$$

Это наилучшие примеры потенциальной мощи тригонометрических сумм; фактически, этот подход является одним из наиболее эффективных путей получения закона взаимности. Обобщите этот результат, чтобы получить формулу из теоремы 2.3.4 для $\left(\frac{2}{p}\right)$. Можно ли обобщить этот результат до более общего утверждения о взаимности (например, для взаимно простых $m, n$) из теоремы 2.3.4? Кстати, гауссовы суммы для не простых параметров $m, n$ можно вычислить с помощью явной формулы, используя методику из упр. 1.66 или методы, описанные в таких источниках, как [Graham and Kolesnik 1991].

**2.29.** Это упражнение предназначено для приобретения навыков манипуляции гауссовыми суммами. Задание состоит в том, чтобы посчитать точное число арифметических прогрессий заданной длины, членами которых являются только квадратичные вычеты по модулю простого $p$. Формальное число прогрессий длины 3 полагается равным

$$A(p) = \#\left\{(r, s, t) \ : \ \left(\tfrac{r}{p}\right) = \left(\tfrac{s}{p}\right) = \left(\tfrac{t}{p}\right) = 1;\ r \neq s;\ s - r \equiv t - s \pmod p\right\}.$$

Отметим, что полагая $0 \leqslant r, s, t \leqslant p - 1$, мы игнорируем тривиальные прогрессии $(r, r, r)$, и что 0 не является квадратичным вычетом. Таким образом, для простого числа $p = 11$, квадратичными вычетами которого являются числа $\{1, 3, 4, 5, 9\}$, имеется всего $A(11) = 10$ арифметических прогрессий длины 3. (Одна из них — это $4, 9, 3$; таким образом, мы допускаем зацикливание по модулю 11; также разрешаются нисходящие последовательности, такие как $5, 4, 3$.)

Во-первых, докажите, что

$$A(p) = -\frac{p-1}{2} + \frac{1}{p}\sum_{k=0}^{p-1}\sum_{r,s,t} e^{2\pi i k(r-2s+t)/p},$$

где каждое из $r, s, t$ пробегает все квадратичные вычеты. Затем используйте

соотношения (2.12), чтобы доказать, что

$$A(p) = \frac{p-1}{8}\left(p - 6 - 2\left(\frac{2}{p}\right) - \left(\frac{-1}{p}\right)\right).$$

Наконец, выведите для точного числа прогрессий эффективную формулу

$$A(p) = (p-1)\left\lfloor \frac{p-2}{8} \right\rfloor.$$

Интересное обобщение этого упражнения — проанализировать прогрессии большей длины. Другой вариант: сколько прогрессий данной длины можно было бы ожидать среди *случайной* половины вычетов $\{1,\ 2,\ 3,\ \ldots,\ p-1\}$ (см. упр. 2.41)?

**2.30.** Докажите, что алгоритмы извлечения квадратных корней 2.3.8 и 2.3.9 работают корректно.

**2.31.** Докажите, что следующий алгоритм (несомненно, напоминающий приведенный в тексте алгоритм 2.3.9) находит квадратные корни $\pmod p$ для простого нечетного $p$. Пусть $x$ — квадратичный вычет, для которого мы хотим найти квадратный корень. Определим частную последовательность Люка $(V_k)$ равенствами $V_0 = 2, V_1 = h$, и для $k > 1$

$$V_k = hV_{k-1} - xV_{k-2},$$

где $h$ таково, что $\left(\frac{h^2-4x}{p}\right) = -1$. Затем вычислим квадратный корень из $x$ как

$$y = \frac{1}{2}V_{(p+1)/2} \pmod p.$$

Отметим, что числа Люка можно вычислить с помощью двоичной цепи Люка; см. алгоритм 3.6.7.

**2.32.** Реализуйте алгоритм 2.3.8 или 2.3.9 (или некоторый другой), чтобы решить сравнения

$$x^2 \equiv 3615 \pmod{2^{16}+1},$$

$$x^2 \equiv 55251255643048601698408223 7 \pmod{2^{89}-1}.$$

**2.33.** Покажите, как можно улучшить алгоритм 2.3.8, избегая части возведений в степень, вероятно, используя для этого предварительные вычисления.

**2.34.** Докажите, что первообразный корень для нечетного простого $p$ является квадратичным невычетом.

**2.35.** Докажите, что алгоритм 2.3.12 (или 2.3.13) работает корректно. Как упоминается в тексте, доказательство не исключительно простое. Возможно, проще сначала доказать алгоритм для специального случая, а именно, найти представление $p = a^2 + b^2$, если $p \equiv 1 \pmod 4$. Такое представление всегда существует и единственно.

**2.36.** Поскольку у нас есть алгоритмы, извлекающие квадратные корни по модулю простых чисел, постройте алгоритм для извлечения квадратного корня $\pmod n$, где $n = pq$ — произведение двух явно заданных простых чисел.

(Здесь будет полезна китайская теорема об остатках.) Как можно извлекать квадратные корни по модулю степени простого числа $n = p^k$? Как можно извлекать квадратные корни по модулю $n$, если известно полное разложение $n$ на простые множители?

Отметим, что если разложение числа $n$ не известно, извлечение квадратного корня чрезвычайно сложно — по сути, эквивалентно самому разложению (см. упр. 6.5).

**2.37.** Докажите, что для нечетного простого числа $p$ число корней уравнения $ax^2 + bx + c \equiv 0 \pmod{p}$, где $a \not\equiv 0 \pmod{p}$, равно $1 + \left(\frac{D}{p}\right)$, где $D = b^2 - 4ac$ — дискриминант. Для случая $1 + \left(\frac{D}{p}\right) > 0$ опишите алгоритм поиска всех корней.

**2.38.** Найдите простое число $p$, такое что наименьший примитивный корень по модулю $p$ превосходит число двоичных битов $p$. Найдите пример такого простого числа $p$, что оно также является простым числом Мерсенна (т. е. такое простое $p = M_q = 2^q - 1$, чей наименьший примитивный корень превосходит $q$). Такие находки показывают, что наименьший примитивный корень может превосходить $\lg p$. По поводу большего исследования в этом направлении см. упр. 2.39.

## 2.5. Проблемы для исследования

**2.39.** Реализуйте алгоритм поиска первообразных корней и исследуйте статистическое распределение наименьших первообразных корней.

Исследование наименьших первообразных корней в высшей степени интересно. Известно, что согласно обобщенной гипотезе Римана, 2 является первообразных корнем для бесконечного числа простых чисел, на самом деле составляющих положительную долю $\alpha = \prod(1 - 1/p(p-1)) \approx 0.3739558$, где произведение берется по всем простым числам (см. упр. 1.90). Опять же, согласно обобщенной гипотезе Римана, у положительной доли тех чисел, чей наименьший первообразный корень не равен 2, первообразным корнем является 3, и так далее (см. книгу [Hooley 1976]). Было высказано предположение, что наименьший первообразный корень простого числа $p$ составляет $O((\ln p)(\ln \ln p))$; см. [Bach 1997a]. Известно, что в предположении обобщенной гипотезы Римана наименьший первообразный корень $p$ равен $O\left(\ln^6 p\right)$ (см. [Shoup 1992]). Известно (без дополнительных предположений), что первообразный корень для простого числа $p$ равен $O(p^{1/4+\varepsilon})$ для любого $\varepsilon > 0$, и для бесконечного числа простых $p$ он превосходит $c \ln p \ln \ln \ln p$ с некоторой постоянной $c$; последнее является результатом Грэхема и Рингроуза. Исследование наименьшего первообразного корня весьма похоже на исследования наименьшего квадратичного невычета — по этому поводу см. упр. 2.41.

**2.40.** Исследуйте использование китайской теоремы об остатках в далеких на первый взгляд областях вычисления целочисленной свертки, или быстрого преобразования Фурье, или криптографии с открытым ключом. Хорошим источником является [Ding et al. 1996].

**2.41.** Исследуем то, что можно было бы назвать «статистическими» свойствами символа Лежандра. Для нечетного простого $p$ обозначим через $N(a,b)$ число остатков, для которых пара последовательных квадратичных символов совпадает с парой $(a,b)$; т. е. мы хотим посчитать те целые числа $x \in [1, p-2]$, для которых

$$\left(\left(\frac{x}{p}\right), \left(\frac{x+1}{p}\right)\right) = (a,b)$$

для всех $a, b$, принимающих значения $\pm 1$. Докажите, что

$$4N(a,b) = \sum_{x=1}^{p-2} \left(1 + a\left(\frac{x}{p}\right)\right)\left(1 + b\left(\frac{x+1}{p}\right)\right)$$

и поэтому

$$N(a,b) = \frac{1}{4}\left(p - 2 - b - ab - a\left(\frac{-1}{p}\right)\right).$$

Установите следствие, что число пар из последовательных квадратичных вычетов равно $(p-5)/4$ и $(p-3)/4$ соответственно для случаев $p \equiv 1, 3 \pmod 4$. Используя формулу для $N(a,b)$, докажите, что для каждого простого $p$ сравнение

$$x^2 + y^2 \equiv -1 \pmod p$$

разрешимо.

Одним из хороших свойств формулы для $N(a,b)$ является то, что если считать, что символ Лежандра порождается «случайным подбрасыванием монеты», то любая заданная пара $(\pm 1, \pm 1)$ обязательно встретится примерно $p/4$ раз.

Это имеет смысл: символ Лежандра является в некотором смысле случайным. Но с другой точки зрения, он не вполне случаен. Давайте оценим сумму:

$$s_{A,B} = \sum_{A \leqslant x < B} \left(\frac{x}{p}\right),$$

которую можно в некотором эвристическом смысле понимать как случайное блуждание с числом шагов $N = B - A$. Исходя из пояснений, следующих за теоремой 2.3.7, покажите, что

$$|s_{A,B}| \leqslant \frac{1}{\sqrt{p}} \sum_{b=0}^{p-1} \left|\frac{\sin(\pi N b/p)}{\sin(\pi b/p)}\right| \leqslant \frac{1}{\sqrt{p}} \sum_{b=0}^{p-1} \frac{1}{|\sin(\pi b/p)|}.$$

Наконец, получите неравенство Пойя—Виноградова:

$$|s_{A,B}| < \sqrt{p} \ln p.$$

На самом деле обычно это неравенство формулируется более широко, где вместо символа Лежандра как характера может быть подставлен любой неглавный характер. Это интересное неравенство говорит, что на самом деле «статистическая флуктуация» числа квадратичных вычетов/невычетов, начиная с любого заданного $x = A$, всегда ограничена «множителем вариации» $\sqrt{p}$ (помноженным на логарифм). Можно доказать более сильное условие. Например,

используя неравенство (см. [Cochrane 1987]), можно получить выражение

$$|s_{A,B}| < \frac{4}{\pi^2}\sqrt{p}\ln p + 0.41\sqrt{p} + 0.61,$$

а также известно, что в предположении обобщенной гипотезы Римана, $s_{A,B} = O\left(\sqrt{p}\ln\ln p\right)$ (см. [Davenport 1980]). В любом случае мы получаем, что из любых $N$ последовательных целых чисел $N/2 + O(p^{1/2}\ln p)$ являются квадратичными вычетами по модулю $p$. Мы также заключаем, что наименьший квадратичный невычет по модулю $p$ ограничен сверху в худшем случае $\sqrt{p}\ln p$. Дальнейшие результаты по поводу этого интересного неравенства обсуждаются в работах [Hildebrand 1988a, 1988b].

Неравенство Пойя—Виноградова, таким образом суженное для случая квадратичных характеров, говорит нам, что *не всякая* последовательность подбрасываний монеты может быть последовательностью символов Лежандра. Неравенство говорит, что мы не можем для больших $p$ иметь такую последовательность символов Лежандра, как $(1, 1, 1, \ldots, -1-1-1)$ (т. е. первая половина заполнена 1, а вторая — $-1$). Мы не можем сделать избыток одного значения над другим более чем $O\left(\sqrt{p}\ln p\right)$. Но в по-настоящему случайной игре с подбрасыванием монет любая последовательность из 1 и $-1$ допустима; даже если ограничить такую игру тем, чтобы число значений 1 и $-1$ совпадало, как это делает символ Лежандра, все равно остается множество последовательностей подбрасывания монеты, которые не могут быть последовательностями символа. В некотором смысле, однако, неравенство Пойя—Виноградова ставит символ Лежандра прямиком в середину распределения возможных последовательностей: это именно то, что мы могли бы ожидать от *случайной* последовательности подбрасываний монеты. Кстати, с точки зрения аналогии с подбрасыванием монеты, чему должно быть равно ожидаемое значение наименьшего квадратичного невычета по модулю $p$? По этому поводу см. упр. 2.39. По поводу другого условия на предположительно случайные квадратичные вычеты см. замечания в конце упр. 2.29.

**2.42.** Вот еще одно захватывающее направление исследований: используя старую восхитительную теорию арифметико-геометрического среднего (arithmetic-geometric mean — AGM), исследуйте удивительную идею, которую можно было бы назвать «дискретным арифметико-геометрическим средним» (discrete arithmetic-geometric mean — DAGM). Гениальным изобретением Гаусса, Лежандра и Якоби в анализе было предложенное ими постижение аналитического AGM, которое является пределом красивого итерационного процесса

$$(a, b) \mapsto \left(\frac{a+b}{2}, \sqrt{ab}\right),$$

т. е. каждый раз пара $(a, b)$ действительных чисел заменяется новой парой из арифметического и геометрического среднего, соответственно. Классическое арифметико-геометрическое среднее является действительным числом $c$, к которому сходятся эти два числа; конечно же, $(c, c) \mapsto (c, c)$, так что процесс стабилизируется для правильно выбранных начальных $a$ и $b$. Этот метод связан

с теорией эллиптических интегралов, вычислением $\pi$ до (буквально) миллиардов десятичных знаков и т. п. (см. [Borwein and Borwein 1987]).

Однако рассмотрим выполнение этой процедуры не для вещественных чисел, но для остатков по модулю простого $p \equiv 3 \pmod 4$; в таком случае число $x \pmod p$, имеющее квадратный корень, всегда имеет так называемое главное значение корня (и таким образом, всегда может быть выбрано единственное значение корня; см. упр. 2.25). Создайте теорию для дискретного арифметико-геометрического среднего по модулю $p$. По-видимому, следует понимать $\sqrt{ab}$ как главное значение корня, если этот корень существует, и что-либо похожее на другое главное значение корня, например, $\sqrt{gab}$ для фиксированного невычета $g$, в случае если $ab$ является невычетом. Интересными теоретическими вопросами являются следующие: имеет ли дискретное арифметико-геометрическое среднее интересную структуру циклов? Имеется ли связь между DAGM и классическим AGM? Если бы *существовала* какая-либо чудесная связь между дискретным и аналитическим средним, хотелось бы получить новый способ с чрезвычайной эффективностью вычислять конечные гипергеометрические последовательности вроде тех, что появляются в упр. 7.26.

Глава 3

# РАСПОЗНАВАНИЕ ПРОСТЫХ И СОСТАВНЫХ ЧИСЕЛ

Пусть задано большое целое число. Каким образом можно быстро определить, является оно простым или нет? На этот фундаментальный вопрос мы начинаем отвечать в настоящей главе.

## 3.1. Метод пробных делений

### 3.1.1. Признаки делимости

Для проверки делимости на некоторые небольшие числа существуют простые признаки, основанные на виде десятичной записи исследуемого числа. Например, число является четным, если такова его последняя цифра. (Забавно, что нематематики иногда принимают за *определение* четности этот признак, а не условие делимости на два.) Аналогичным образом, если число оканчивается на 0 или 5, то оно кратно 5.

Разумеется, признаки делимости на 2 и 5 так просты потому, что эти числа делят основание 10 нашей системы счисления. Делимость на другие числа проверить не так легко. Наверное, следующими по популярности признаками делимости являются признаки для 3 и 9, основанные на том факте, что сумма цифр числа $n$ сравнима с $n$ по модулю 9. Поэтому для проверки делимости числа $n$ на 3 или 9 достаточно адресовать тот же вопрос сумме цифр числа $n$. Отметим, что переход к сумме цифр связан с тем, что 10 на единицу превосходит 9. Если бы мы записывали числа, например, по основанию 12, то число было бы сравнимо по модулю 11 с суммой своих цифр (по основанию 12).

Вообще говоря, сложность проверки делимости, основанной на цифрах, растет по мере увеличения мультипликативного порядка основания системы счисления по модулю рассматриваемого делителя. Например, порядок $10 \pmod{11}$ равен 2, так что признак делимости на 11 достаточно прост: знакочередующая сумма цифр числа $n$ сравнима с $n \pmod{11}$. А для 7 порядок 10 равен 6, и, как известно, признаки делимости на 7 куда более громоздки.

С вычислительной точки зрения между применением признака делимости на простое число $p$ и непосредственным делением на $p$ с остатком разница не велика. Кроме того, при обыкновенном делении не требуется знать специальные формулы или приемы, зависящие от выбора числа $p$. Поэтому при работе на ЭВМ или даже во время больших вычислений на бумаге применение метода пробных делений на различные простые числа $p$ не менее эффективно, чем использование разнообразных признаков делимости.

### 3.1.2. Метод пробных делений

Метод пробных делений заключается в последовательном делении числа $n$ на пробные числа для получения частичного или полного разложения числа $n$ на множители. Сначала берется первое простое число, 2, и мы начинаем делить данное число $n$ на 2 до тех пор, пока это возможно. Затем берется следующее простое число, 3, и мы начинаем процесс деления на 3 оставшейся части числа $n$ и т. д. Если мы достигнем пробного делителя, превосходящего квадратный корень из неразложенной части, то на этом можно остановиться, поскольку сама неразложенная часть есть простое число.

Рассмотрим пример. Пусть дано число $n = 7399$. Мы пробуем разделить его на 2, 3 и 5 и находим, что эти числа делителями не являются. Далее пробуем число 7. Это делитель, причем частное равно 1057. Снова пробуем число 7 и видим, что деление опять возможно, причем частное равно 151. Еще раз пробуем число 7, но оно не делит 151. Следующий пробный делитель равен 11, но он не подходит. Затем мы должны взять число 13, но оно превосходит квадратный корень из 151, а следовательно, 151 — простое число. Таким образом, мы получили разложение на множители вида $7399 = 7^2 \cdot 151$.

Пробные делители не обязаны непременно быть простыми числами. Если мы предпримем попытку деления на составное число $d$, предварительно исключив из числа $n$ простые делители числа $d$, то просто окажется, что это число $d$ не делит оставшуюся часть числа $n$. Таким образом, мы лишь потеряем некоторое время, но это не помешает найти разложение на простые множители.

Рассмотрим пример. Пусть $n = 492$. Начинаем делить на 2 и видим, что эта попытка удалась и частное равно 246. Снова делим на 2 и получаем частное 123. Дальнейшее деление на 2 невозможно. Переходим к следующему пробному делителю, 3, и получаем частное 41. Затем последовательно делим на 3, 4, 5 и 6 и убеждаемся, что эти числа не делят оставшуюся часть 41. Поскольку следующий пробный делитель, 7, превосходит $\sqrt{41}$, мы получили разложение на простые множители $492 = 2^2 \cdot 3 \cdot 41$.

Теперь давайте рассмотрим соседнее число $n = 491$. Попытки деления на последовательные числа $2, 3, \ldots, 22$ не удаются. Следующее пробное число равно 23, но $23^2 > 491$, так что мы установили простоту числа 491.

Рассмотренный процесс можно ускорить, заметив, что после 2 все простые числа нечетны. Таким образом, в качестве пробных делителей достаточно использовать 2 и все нечетные числа. В случае $n = 491$ это позволило бы нам избежать деления на все четные числа от 4 до 22. Приведем краткое описание алгоритма метода пробных делений на 2 и все нечетные числа, большие двух.

**Алгоритм 3.1.1** (метод пробных делений). Дано целое число $n > 1$. Алгоритм создает мультимножество $\mathcal{F}$ простых делителей числа $n$. (Термин «мультимножество» означает, что элементы этого множества могут повторятся, т. е. они входят в это множество с некоторой кратностью.)

1. [Деление на 2]
   $\mathcal{F} = \{\ \}$; // Пустое мультимножество.
   $N = n$;

```
    while(2 | N) {
        N = N/2;
        F = F ∪ {2};
    }
2. [Основной цикл деления]
    d = 3;
    while(d^2 ⩽ N) {
        while(d | N) {
            N = N/d;
            F = F ∪ {d};
        }
        d = d + 2;
    }
    if(N == 1) return F;
    return F ∪ {N};
```

Поскольку все простые числа, большие 3, сравнимы или с 1, или с 5 (mod 6), можно выбирать пробные делители только этого вида, поочередно прибавляя к предыдущему пробному делителю 2 или 4. Это частный случай «колеса», т. е. конечного набора прибавляемых чисел, применяемого бесконечное множество раз. Например, все простые числа, большие 5, лежат в восьми классах вычетов (mod 30), причем колесо, пробегающее все эти классы (начиная с 7), имеет вид

$$4, 2, 4, 2, 4, 6, 2, 6.$$

Сложность колеса растет очень быстро. Например, колесо, пробегающее числа, не кратные ни одному из простых чисел, меньших 30, состоит из 1021870080 чисел. Кроме того, по сравнению с простым колесом 2, 4, построенным для простых чисел 2 и 3, мы экономим не многим более 50% попыток деления. (Именно, около 52.6% всех чисел, взаимно простых с 2 и 3, имеют простой делитель, меньший 30.) Поэтому использование такого неуклюжего колеса представляется несколько нелепым. Обеспокоенность потерей времени из-за деления на составные числа можно преодолеть легче и эффективнее: достаточно просто подготовить список простых чисел, которые будут использоваться в качестве пробных делителей. Действенные способы создания такого списка мы рассмотрим в разд. 3.2.

### 3.1.3. Практические соображения

Метод пробных делений прекрасно подходит для проверки на простоту в том случае, когда исследуемое число не велико. Разумеется, «не велико» — понятие субъективное, зависящее от быстроты вычислительной техники и того времени, которое Вы можете позволить компьютеру работать. Кроме того, имеет значение, желаете ли Вы проверить какое-то одно заинтересовавшее Вас число или собираетесь встроить метод пробных делений в виде процедуры, многократно вызываемой другим алгоритмом. Говоря очень приблизительно, современная рабочая станция позволяет за одну минуту распознать методом

пробных делений простые числа, состоящие не более чем из 13 десятичных разрядов, а за сутки, быть может, удастся исследовать и 19-значное число. (Впрочем, подобные эмпирические законы, естественно, зависят от производительности вычислительной техники в каждый конкретный период времени.)

Метод пробных делений также является действенным методом частичного разложения на множители $n = FR$ (см. разд. 3.1.2). Дело в том, что для каждой фиксированной границы пробных делителей $B \geqslant 2$ как минимум четверть всех чисел имеют делитель $F$, который превосходит $B$ и состоит исключительно из простых чисел, не превосходящих $B$ (см. упр. 3.4).

Кроме того, метод пробных делений позволяет легко и эффективно распознавать гладкие числа, или числа, не имеющие больших простых делителей (см. определение 1.4.8). Иногда бывает полезным иметь в наличии «тест на гладкость», который позволяет для заданного параметра $B$ узнать, является ли $B$-гладким данное число $n$, т. е. все ли делители числа $n$ не превосходят $B$. Метод пробных делений до границы $B$ не только дает ответ на этот вопрос, но и предоставляет разложение на простые множители наибольшего $B$-гладкого делителя числа $n$. Поскольку эта глава посвящена не разложению на простые множители, а распознаванию простых и составных чисел, мы на некоторое время отложим дальнейшее обсуждение тестов на гладкость.

### 3.1.4. Теоретические соображения

Предположим, что мы желаем воспользоваться методом пробных делений для полного разложения числа $n$ на простые множители. Какое время нам потребуется в самом худшем случае? Очевидно, что самый плохой случай будет тогда, когда само число $n$ простое и нам придется использовать все возможные делители вплоть до $\sqrt{n}$. Если в качестве пробных делителей мы используем лишь простые числа, то количество попыток будет примерно равно $2\sqrt{n}/\ln n$. Если в качестве пробных делителей взять 2 и нечетные числа, то количество проб будет примерно равно $\frac{1}{2}\sqrt{n}$. Применение колеса, рассмотренное выше, позволяет несколько уменьшить постоянную $\frac{1}{2}$.

Итак, мы оценили время работы метода пробных делений при распознавании простоты. А какова сложность соответствующего алгоритма полного разложения на множители составного числа? Здесь снова в худшем случае получается $\sqrt{n}$ (достаточно рассмотреть случай $n = 2p$, где $p$ — простое число). Также можно поставить вопрос о средней сложности разложения составных чисел. Здесь также получается почти $\sqrt{n}$ из-за величины вклада составных чисел, имеющих очень большой простой делитель. Но поскольку такие числа редки, может быть интересным отбросить 50% худших чисел и рассчитать среднее время работы метода пробных делений для полного разложения на множители оставшихся чисел. Оказывается, что получится $n^c$, где $c = 1/(2\sqrt{e}) \approx 0.30327$ (см. упр. 3.5).

Как мы увидим далее в этой и следующей главах, проблема распознавания простых чисел является гораздо менее сложной, чем задача полного разложения на множители. Дело в том, что для распознавания простых чисел разрабо-

таны гораздо лучшие алгоритмы, чем метод пробных делений. Поэтому, если при разложении на множители используется метод пробных делений, то его можно ускорить путем сочетания с быстрым тестом на простоту, применяемым каждый раз, когда получается новое значение неразложенной части числа $n$: если это число простое, то процесс разложения можно остановить. При таком ускорении метода пробных делений время полного разложения составного числа $n$ по существу равно квадратному корню из второго по величине простого делителя числа $n$ (считая с наибольшего).

В этом случае также велик вклад редкого множества чисел, а именно, чисел, равных произведению двух примерно равных простых чисел. При этом соответствующее среднее время равно $\sqrt{n}$. Но если отбросить 50% худших чисел, то можно получить более хорошую оценку для оставшихся: $n^c$, где $c \approx 0.23044$.

## 3.2. Просеивание

Просеивание может быть высоко эффективным средством определения простоты и разложения на множители в тех случаях, когда требуется получить результат для каждого числа из большого множества натуральных чисел, расположенных с равными интервалами. Среднее число арифметических операций на долю каждого числа этого множества может быть очень небольшим и, по существу, ограниченным.

### 3.2.1. Просеивание для распознавания простых чисел

Скорее всего, большинству читателей знакомо решето Эратосфена. Его наиболее известное применение состоит в нахождении всех простых чисел, не превосходящих заданной границы $N$. Начнем с массива из $N-1$ «единиц», соответствующих числам от 2 до $N$. Первая из них отвечает числу 2, так что все единицы, соответствующие числам 4, 6, 8 и т. д. заменяются на нули. Затем рассмотрим число 3 и заменим все единицы, отвечающие числам 6, 9, 12 и т. д., на нули. (Если на каком-то месте уже стоит нуль, то мы его не трогаем.) Продолжаем этот процесс далее: если следующий элемент массива соответствует простому числу $p$, мы обращаем в нуль все элементы, отвечающие числам $2p$, $3p$, $4p$ и т. д. Однако, если новое число $p$ таково, что $p^2 > N$, то наш процесс можно остановить. В самом деле, при просеивании через это число $p$ ничего не изменится, поскольку на всех соответствующих местах уже стоят нули. Итак, мы получили список, в котором единицы соответствуют простым, а нули — составным числам, не превосходящим $N$.

При движении по элементам массива, соответствующим числам $2p$, $3p$, $4p$ и т. д., мы начинаем со стартового простого числа $p$ и последовательно прибавляем $p$ до тех пор, пока полученное число не станет больше $N$. Следовательно, при просеивании выполняется только одна арифметической операция — сложение. Число шагов в решете Эратосфена пропорционально $\sum_{p \leqslant N} N/p$, где $p$

пробегает простые числа. Но

$$\sum_{p \leqslant N} \frac{N}{p} = N \ln \ln N + O(N) \tag{3.1}$$

(см. теорему 427 в книге [Hardy and Wright 1979]). Таким образом, число шагов, которые требуются для нахождения всех простых чисел, не превосходящих $N$, пропорционально $\ln \ln N$. Необходимо отметить, что функция $\ln \ln N$ хотя и стремится к бесконечности, но делает это *очень* медленно. В частности, $\ln \ln N < 10$ для всех $N \leqslant 10^{9565}$ (естественно, здесь $N \geqslant 3$).

Наибольшее ограничение, налагаемое компьютером на решето, связано с огромным объемом необходимой памяти. Иногда требуется хранить массив почти из $N$ элементов. Однако если длина массива падает ниже $\sqrt{N}$, то эффективность решета Эратосфена начинает снижаться. Время, необходимое для просеивания промежутка длины $M$ через простые числа $\leqslant \sqrt{N}$, пропорционально

$$M \ln \ln N + \pi\left(\sqrt{N}\right) + O(M),$$

где через $\pi(x)$ обозначено количество простых чисел, не превосходящих $x$. Поскольку согласно асимптотическому закону распределения простых чисел (теорема 1.1.4) справедливо соотношение $\pi\left(\sqrt{N}\right) \sim 2\sqrt{N}/\ln N$, при малых значениях $M$ это слагаемое может быть гораздо больше, чем «главный член» $M \ln \ln N$. На самом деле, до сих пор не найден метод выявления всех простых чисел промежутка $\left[N, N + N^{1/4}\right]$, который бы работал существенно быстрее индивидуального исследования каждого числа. Четкое описание этого вопроса см. в упр. 3.46.

## 3.2.2. Псевдокод для решета Эратосфена

Приведем практический псевдокод реализации обычного решета Эратосфена для отыскания всех простых чисел на промежутке.

**Алгоритм 3.2.1** (решето Эратосфена для практических целей). Этот алгоритм выявляет все простые числа интервала $(L, R)$ путем создания последовательности из нулей и единиц, соответствующих простоте, по результатам последовательного прогона для $B$ нечетных чисел. Будем считать, что числа $L, R$ четны, причем $R > L$, $B \mid R - L$ и $L > P = \lceil \sqrt{R} \rceil$. Кроме того, мы будем считать доступной таблицу из $\pi(P)$ простых чисел $p_k \leqslant P$.

1. [Инициализация отступов]
   for($k \in [2, \pi(P)]$) $q_k = \left(-\frac{1}{2}(L + 1 + p_k)\right) \bmod p_k$;
2. [Обработка блоков]
   $T = L$;
   while($T < R$) {
   for($j \in [0, B-1]$) $b_j = 1$;
   for($k \in [2, \pi(P)]$) {
   for($j = q_k$; $j < B$; $j = j + p_k$) $b_j = 0$;

$q_k = (q_k - B) \bmod p_k,$
}
for($j \in [0, B-1]$) {
if($b_j == 1$) report $T + 2j + 1$, // Число $p = T + 2j + 1$ — простое
}
$T = T + 2B,$
}

Отметим, что этот алгоритм можно применять как для отыскания всех простых чисел промежутка $(L, R)$, так и для точного подсчета количества этих простых чисел. Усовершенствованные методы подсчета количества простых чисел описаны в разд. 3.7. Использование колеса (см. разд. 3.1) позволяет несколько улучшить базовый алгоритм решета 3.2.1 (см. упр. 3.6).

### 3.2.3. Просеивание для создания таблицы делителей

Внесение небольших изменений в алгоритм решета Эратосфена позволяет усовершенствовать его так, чтобы помимо всех простых чисел $\leqslant N$ этот алгоритм предъявлял также и наименьший простой делитель каждого составного числа $\leqslant N$. Для этого вместо изменения «единицы» на «нуль» при прохождении с шагом $p$ мы будем заменять ее на $p$ (если только на этом месте не стоит ранее записанное меньшее простое число).

Время работы такого алгоритма по сравнению с базовым методом решета Эратосфена не изменяется, однако требуется больший объем памяти.

Таблица делителей может быть полезна для получения полного разложения на множители. Например, на месте числа 12033 будет стоять 3, поскольку наименьший простой делитель числа 12033 равен 3. Разделив 12033 на 3, получим 4011, на месте которого также стоит 3. Снова делим на 3, получая число 1337, которому соответствует 7. Разделив на 7, получим 191, на месте которого стоит 1. Значит, 191 — простое число и мы получили разложение на простые множители

$$12033 = 3^2 \cdot 7 \cdot 191.$$

Таблицы делителей появились гораздо раньше, чем компьютеры. Обширные таблицы делителей, составленные вручную, были крайне необходимы исследователям, производившим вычислительную работу в теории чисел за много лет до появления ЭВМ

### 3.2.4. Просеивание для полного разложения на множители

Как и выше, за счет некоторого увеличения занимаемого места, но очень малого дополнительного времени, можно приспособить решето Эратосфена для получения полного разложения на простые множители каждого числа из заданного промежутка. Это делается путем добавления простого числа $p$ в список, соответствующим позициям $p,\ 2p,\ 3p, \ldots,\ p\lfloor N/p \rfloor$. Здесь также требует-

ся просеивание через степени $p^a$ простых чисел $p \leqslant \sqrt{N}$, для которых $p^a$ не превосходит $N$. К каждому числу, кратному $p^a$, прикрепляется еще один экземпляр простого числа $p$. Во избежание просеивания через простые числа из промежутка $\left(\sqrt{N}, N\right]$, для завершения разложения на множители можно применить деление.

Пусть, к примеру, $N = 20000$. Проследим, что произойдет с элементом $m = 12033$. Просеивая через 3, мы изменяем 1 на 12033-м месте на 3. Просеивая через 9, мы изменяем 3 на 12033-м месте на 3, 3. Просеивая через 7, получаем на этом месте 3, 3, 7. В конце просеивания (через все простые числа вплоть до 139 и их степени вплоть до 20000), мы возвращаемся к каждой позиции и перемножаем компоненты соответствующего списка. На 12033-м месте мы умножаем $3 \cdot 3 \cdot 7$, получив 63. Далее делим 12033 на 63 и заносим частное 191 в список. Итак, итоговый список, соответствующий числу 12033, таков: 3, 3, 7, 191. Мы получили полное разложение на множители числа 12033.

### 3.2.5. Просеивание для распознавания гладких чисел

Алгоритм решета Эратосфена для полного разложения на множители может быть упрощен и превращен в средство распознавания всех $B$-гладких чисел (см. определение 1.4.8) отрезка $[2, N]$. Будем считать, что $2 \leqslant B \leqslant \sqrt{N}$. Для этого достаточно реализовать описанный выше метод с двумя упрощениями:

(1) не просеивать через $p^a$, если $p$ превосходит $B$,

(2) если произведение чисел списка не совпадает с соответствующим этому месту числом, не стоит затруднять себя делением этого числа на полученное произведение.

Дело в том, что $B$-гладкими числами будут именно те, которые совпадают с произведением чисел списка на соответствующих им местах.

Описанный процесс можно несколько упростить. Поскольку нас интересует не разложение на множители, а лишь выявление $B$-гладких чисел, можно хранить не весь список простых делителей, а одно число, которое умножается на $p$ при каждом попадании $p^a$ в соответствующее место. Тогда в конце просеивания $B$-гладкими числами будут именно те, которые совпадают с полученным произведением.

Например, пусть $B = 10$ и $N = 20000$. Произведение, соответствующее числу 12033, начинается с 1 и затем становится равным 3, 9 и, наконец, 63. Значит, число 12033 не является 10-гладким. А, например, на месте, соответствующем числу 12000, мы получим последовательно 2, 4, 8, 16, 32, 96, 480, 2400 и, наконец, 12000. Значит, число 12000 является 10-гладким.

Важный прием ускорения этого метода заключается в работе с логарифмами получающихся чисел. Строгое выполнение арифметических операций с логарифмами требует бесконечной точности, но нам такая строгость и не требуется. Пусть, например, мы используем ближайшее целое число к логарифму по основанию 2. Для 12000 это 14. Также можно воспользоваться приближениями $\lg 2 \approx 1$ (здесь, конечно, точное равенство), $\lg 3 \approx 2$, $\lg 5 \approx 2$, $\lg 7 \approx 3$. Тогда

число, соответствующее позиции 12000, принимает последовательно значения 1, 2, 3, 4, 5, 7, 9, 11, 13. Последнее число достаточно близко к 14, чтобы мы могли признать гладкость числа 12000. Вообще говоря, при поиске $B$-гладких чисел можно пренебрегать остатком, меньшим $\lg B$.

Нельзя не отметить огромное преимущество такой работы с приближенными логарифмами. Во-первых, числа, с которыми проводятся действия, очень малы. Во-вторых, применяемой арифметической операцией является сложение, реализация которого на большинстве ЭВМ гораздо быстрее, чем для умножения и деления. Кроме того, при больших значениях аргумента логарифмическая функция изменяется очень медленно, так что всем близким позициям решета соответствует, по существу, один и то же логарифм. Например, для 12000 мы получили число 14. Это же значение будет и для всех чисел от $2^{13\,5}$ до $2^{14\,5}$, т. е. от 11586 до 23170.

Ниже мы рассмотрим важное применение такого решета в алгоритмах разложения на множители. Кроме того, как указано в разд. 6.4, просеивание для распознавания гладких чисел является существенной компонентой некоторых методов дискретного логарифмирования. В таких задачах от нас не требуется *безупречный* результат просеивания: достаточно, чтобы было найдено большинство $B$-гладких чисел, а количество чисел, ошибочно признанных $B$-гладкими, было не велико. Это освобождающее соображение позволяет еще больше ускорить просеивание. Время просеивания через простое число $p$ пропорционально произведению длины просеиваемого множества на $1/p$. В частности, небольшие простые числа требуют наибольшего времени. Но вклад их логарифмов в сумму очень мал, так что можно согласиться пропустить при просеивании эти простые числа, тем самым несколько увеличив остаточный член решета. Возвращаясь к описанному примеру, откажемся, скажем, от просеивания по модулям 2, 3, 4 и 5. Будем просеивать лишь через более высокие степени 2, 3 и 5, а также через все степени 7 для распознавания 10-гладких чисел. Тогда на месте, соответствующем числу 12000, получим 3, 4, 5, 9 и 11. Полученное значение достаточно близко к 14, так что число 12000 признается гладким. При этом мы не только не пропустили гладкое число, но и избежали самой трудоемкой части метода решета.

### 3.2.6. Просеивание многочлена

Пусть $f(x)$ — многочлен с целыми коэффициентами. Рассмотрим числа $f(1), f(2), \ldots, f(N)$. Допустим мы желаем найти среди этих чисел простые, или создать для них таблицу делителей, или выявить $B$-гладкие числа. Все эти задачи можно легко решить при помощи решета. На самом деле, мы уже сталкивались с частным случаем одной из этих проблем, когда заметили, что при отыскании простых чисел методом решета достаточно использовать, по существу, только нечетные числа $\leqslant N$, т. е. значения многочлена $f(x) = 2x+1$.

Для просеивания последовательности $f(1), f(2), \ldots, f(N)$ присвоим всем элементам массива, занумерованным числами $1, 2, \ldots, N$, значение 1. Важно отметить, что если число $p$ простое и число $a$ таково, что $f(a) \equiv 0 \pmod p$,

то $f(a+kp) \equiv 0 \pmod p$ при всех целых значениях $k$. Как известно, число решений первого из этих сравнений не превосходит $\deg f$ и совпадает с числом различных арифметических прогрессий $\{a+kp\}$ для каждого просеивающего простого числа $p$.

Проиллюстрируем сказанное на примере многочлена $f(x) = x^2+1$. Наша цель — найти простые числа вида $x^2+1$, где $x$ целое число, $1 \leqslant x \leqslant N$. Для каждого простого числа $p \leqslant N$ решаем сравнение $x^2+1 \equiv 0 \pmod p$ (см. разд. 2.3.2). При $p \equiv 1 \pmod 4$ существуют ровно два решения, при $p \equiv 3 \pmod 4$ корней нет, а при $p = 2$ есть только один корень. Для каждого простого числа $p$ и корня сравнения $a$ (т. е. $a^2+1 \equiv 0 \pmod p$ и $1 \leqslant a < p$) пробегаем класс вычетов $a \pmod p$ вплоть до $N$, заменяя все единицы на нули. Однако самое маленькое значение $a$ может соответствовать текущему простому числу $p$, но этот случай легко исправить, проверив условие $a < \sqrt{p}$ или численно установив равенство $a^2+1 = p$. Разумеется, если $p = a^2+1$, мы должны на соответствующем месте оставить единицу.

Напомним, что этот метод работает потому, что сравнение $a^2+1 \equiv 0 \pmod p$ справедливо тогда и только тогда, когда $(a+kp)^2+1 \equiv 0 \pmod p$ для всех целых значений $k$ (нам нужны лишь те значения $k$, при которых $1 \leqslant a+kp \leqslant N$).

Важное отличие от обычного метода решета Эратосфена заключается в глубине поиска простых чисел. Согласно основному принципу, просеивающие простые числа должны достигать квадратного корня из наибольшего члена последовательности $f(1), f(2), \ldots, f(N)$ (мы считаем все эти значения положительными). В случае многочлена $x^2+1$ это означает, что мы должны использовать все простые числа $\leqslant N$, не останавливаясь на значении $\sqrt{N}$, как положено в обычном решете Эратосфена.

После того как найдены все корни сравнений $x^2+1 \equiv 0 \pmod p$, просеивание многочлена $x^2+1$ для поиска простых чисел при $x \leqslant N$ занимает примерно то же время, что и обычное решето Эратосфена. Это кажется невероятным, поскольку теперь для многих простых чисел мы должны просеивать два класса вычетов, рассматривая все простые числа вплоть до $N$, а не $\sqrt{N}$. В ответ на первое высказывание заметим, что оно верно, но количество простых чисел, для которых не надо просеивать ни один класс вычетов, также велико. Что касается второго возражения, то дело здесь в том, что сумма величин, обратных ко всем простым числам между $\sqrt{N}$ и $N$, ограничена при растущих значениях $N$ (и стремится к $\ln 2$), так что дополнительное время просеивания равно всего лишь $O(N)$. Кроме того, справедливо не только соотношение

$$\sum_{p \leqslant \sqrt{N}} \frac{1}{p} = \ln\ln N + O(1),$$

но и равенство

$$\frac{1}{2} + 2 \sum_{p \leqslant N,\ p \equiv 1 \pmod 4} \frac{1}{p} = \ln\ln N + O(1)$$

(см. гл. 7 книги [Davenport 1980]).

При поиске $B$-гладких чисел важно уметь просеивать последовательные значения многочлена (см. разд. 3.2.5). Можно сказать, что все идеи того раздела развивают изложенные здесь методы.

### 3.2.7. Теоретические соображения

Сложность $N \ln\ln N$ решета Эратосфена может быть слегка уменьшена при помощи нескольких ценных соображений. Следующий алгоритм основан на идеях Мэрсона и Притчарда (см. [Pritchard 1981]). Он требует всего лишь $O(N/\ln\ln N)$ шагов, каждый из которых есть или учет, или сложение чисел $\leqslant N$. (Напомним, что в явном виде псевдокод традиционного решета Эратосфена приведен в разд. 3.2.2.)

**Алгоритм 3.2.2** (улучшенное решето Эратосфена). Для заданного числа $N \geqslant 4$ алгоритм находит все простые числа на отрезке $[1, N]$. Пусть $p_l$ обозначает $l$-е простое число. Положим $M_l = p_1 p_2 \ldots p_l$ и обозначим через $S_l$ множество всех чисел промежутка $[1, N]$, взаимно простых с $M_l$. Заметим, что если $p_{m+1} > \sqrt{N}$, то множество простых чисел отрезка $[1, N]$ есть $(S_m \setminus \{1\}) \cup \{p_1, p_2, \ldots, p_m\}$. Алгоритм рекурсивно находит множества $S_k, S_{k+1}, \ldots, S_m$, начиная с умеренного значения $k$, и заканчивает работу при $m = \pi(\sqrt{N})$.

1. [Настройка]
   Присвоим целому числу $k$ такое значение, что $M_k \leqslant N/\ln N < M_{k+1}$;
   Положим $m = \pi(\sqrt{N})$;
   При помощи обычного решета Эратосфена (алгоритм 3.2.1) находим простые числа $p_1, p_2, \ldots, p_k$, а также множество целых чисел отрезка $[1, M_k]$, взаимно простых с $M_k$;
2. [Вращать колесо]
   Вращать «колесо» для $M_k$ (см. разд. 3.1), чтобы составить множество $S_k$;
   $S = S_k$;
3. [Найти промежутки]
   for($l \in [k+1, m]$) {
   $p = p_l =$ наименьший элемент множества $S$, превосходящий 1;
   // В этот момент $S = S_{l-1}$.
   Нахождение *множества* $G$ длин промежутков между последовательными элементами множества $S \cap [1, N/p]$;
   // Каждое число входит во множество $G$ только раз.
   Нахождение множества $pG = \{pg : g \in G\}$;
   // Используем метод последовательных удвоений (см. алгоритм 2.1.5).
4. [Найти специальное множество]
   Находим множество $pS \cap [1, N] = \{ps : ps \leqslant N, s \in S\}$ так: если $s$ и $s'$ — последовательные элементы множества $S$, причем $s'p \leqslant N$ и значение $sp$ уже вычислено, то $ps' = ps + p(s' - s)$;
   // Отметим, что разность $s' - s$ принадлежит множеству $G$, а число $p(s' - s)$ уже было вычислено на шаге [Найти промежутки].

Так что значение $ps'$ можно найти посредством вычитания (для получения $s'-s$), поиска (элемента $p(s'-s)$) и сложения. (Поскольку наименьший элемент множества $S$ равен 1, начальное значение $ps$ равно самому числу $p$ и вычисления не требуются.)

5. [Найти следующее множество $S$]
$S = S \setminus (pS \cap [1, N])$; // Теперь $S = S_l$.
$l = l + 1$;
}

6. [Возвратить множество простых чисел на отрезке $[1, N]$]
return $(S \setminus \{1\}) \cup \{p_1, p_2, \ldots, p_m\}$;

Каждое множество $S_l$ состоит из чисел отрезка $[1, N]$, которые взаимно просты с каждым из чисел $p_1, p_2, \ldots, p_l$. Следовательно, следующий за единицей элемент является $(l+1)$-м простым числом, $p_{l+1}$. Оценим количество операций алгоритма 3.2.2. Число операций на шаге [Настройка] равно $O((N/\ln N)\sum_{i\leqslant k} 1/p_i) = O(N \ln\ln N/\ln N)$. (На самом деле, выражение $\ln\ln N$ можно заменить на $\ln\ln\ln N$, но для наших целей это не требуется.) Шаг [Вращать колесо] требует $\#S_k \leqslant \lceil N/M_k \rceil \varphi(M_k) = O(N\varphi(M_k)/M_k)$ операций, где $\varphi$ — функция Эйлера. Отношение $\varphi(M_k)/M_k$ в точности равно произведению чисел $1 - 1/p_i$, где $i = 1, \ldots, k$. По теореме Мертенса 1.4.2, это произведение асимптотически эквивалентно $e^{-\gamma}/\ln p_k$. Далее, по теореме 1.1.4 о простых числах $p_k$ асимптотически эквивалентно $\ln N$. Следовательно, число операций на шаге [Вращать колесо] есть $O(N/\ln\ln N)$.

Остается подсчитать количество операций на шаге [Найти промежутки] и на шаге [Найти специальное множество]. Пусть $S = S_{l-1}$ и $G_l = G$. В силу теоремы Мертенса, множество $S \cap [1, N/p_l]$ содержит $O(N/(p_l \ln p_{l-1}))$ элементов. Поэтому общее число шагов, требуемых для нахождения всех множеств $G_l$, не превосходит некоторой постоянной, умноженной на

$$\sum_{l=k+1}^{m} N/(p_l \ln p_{l-1}) = O(N/\ln p_k) = O(N/\ln\ln N).$$

Число сложений, необходимых для вычисления значений $gp_l$, соответствующих элементу $g$ множества $G_l$, методом последовательного удвоения есть $O(\ln g) = O(\ln N)$. Сумма всех значений $g$ множества $G_l$ не превосходит $N/p_l$, так что число элементов множества $G_l$ есть $O\left(\sqrt{N/p_l}\right)$. Следовательно, общее число сложений на шаге [Найти промежутки] не превосходит константы, умноженной на

$$\sum_{l=k+1}^{m} \sqrt{\frac{N}{p_l}} \ln N \leqslant \sum_{i=2}^{\lfloor\sqrt{N}\rfloor} \sqrt{\frac{N}{i}} \ln N \leqslant \int_1^{\sqrt{N}} \sqrt{\frac{N}{t}} \ln N \, dt \leqslant 2N^{3/4} \ln N.$$

Оценку числа операций на шаге [Найти специальное множество] проведем более аккуратно. Каждая из этих операций есть просто учетное действие удаления элемента множества. Поскольку каждый элемент может быть удален лишь один раз, достаточно найти общее число удалений за все итерации шага

[Найти специальное множество]. Но это число равно мощности множества $S_k$ минус количество простых чисел промежутка $\left[\sqrt{N}, N\right]$. Значит, рассматриваемое количество операций не превосходит значения $\#S_k$, для которого мы уже получили оценку $O(N/\ln\ln N)$.

## 3.3. Распознавание гладких чисел

Многие теоретико-числовые алгоритмы включают в себя распознавание гладких чисел из заданного множества как существенную подпрограмму. Методов распознавания гладких чисел довольно много, в частности, для этого подойдет любой алгоритм разложения на множители. Однако некоторые алгоритмы разложения на множители (например, метод пробных делений) сначала находят простые делители в порядке их возрастания, начиная с наименьшего. Такой подход, очевидно, позволяет определить, что число не является $y$-гладким, не обязательно доводя процесс разложения на множители до конца. Помимо метода пробных делений этим свойством обладают ро-метод Полларда и метод эллиптических кривых Ленстры, которые мы рассмотрим ниже. Грубые оценки сложности (числа операций) каждого из этих трех алгоритмов при использовании в качестве теста на $y$-гладкость числа таковы: метод пробных делений — $y^{1+o(1)}$, ро-метод Полларда — $y^{1/2+o(1)}$, метод эллиптических кривых — $\exp((2\ln y \ln\ln y)^{1/2}) = y^{o(1)}$. Здесь под словом «операция» понимается арифметическое действие с числами порядка исследуемого числа. (Следует отметить, что приведенные оценки сложности как ро-метода, так и метода эллиптических кривых носят эвристический характер.)

Иногда гладкие числа можно распознать посредством просеивания, причем сделать это очень быстро. Например, если задано множество последовательных целых чисел (или даже множество значений многочлена с целыми коэффициентами низкой степени на последовательных целых числах) длины $L \geqslant y$ с максимальным элементом $M$, то исследование всех $L$ чисел на $y$-гладкость можно осуществить за время порядка $L\ln\ln M \ln\ln y$ или около $\ln\ln M \ln\ln y$ битовых операций на одно число. (Множитель $\ln\ln M$ возникает от использования приближенных логарифмов, рассмотренного в разд. 3.2.5.) Более того, просеивание происходит настолько быстро, что на доступ к числам в памяти уходит больше времени, чем на реальные вычисления.

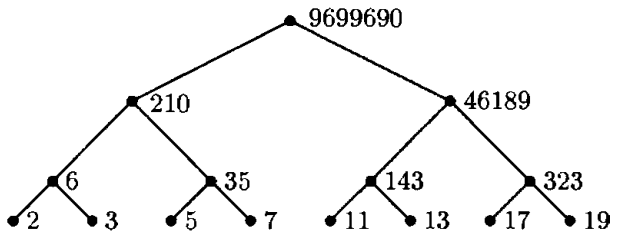
В этом разделе мы обсудим новый эффективный метод Д. Бернштейна (см. [Bernstein 2004d]), позволяющий распознавать $y$-гладкие числа в любом множестве, содержащем не менее $y$ чисел, и работающий почти так же быстро, как просеивание: он требует $(\ln^2 y \ln M)^{1+o(1)}$ битовых операций на одно число для чисел, не превосходящих $M$. Однако для реализации метода, обладающего настолько малой сложностью, приходится привлекать такие нетривиальные алгоритмы работы с большими числами, как быстрое преобразование Фурье или другие методы свертки (см. гл. 9 и работу [Bernstein 2004e]).

Рассмотрим метод Бернштейна на примере. Положим границу гладкости $y$ равной 20 и исследуем числа $1001, 1002, \ldots, 1008$. (Вообще говоря, числа не обя-

заны быть последовательными, мы взяли их такими лишь для упрощения дальнейшего изложения.) Непосредственная проверка показывает, что 20-гладкими числами из нашего списка являются только первое и последнее, а именно, 1001 и 1008. Предлагаемый ниже алгоритм не только обнаруживает эти числа, но и указывает наибольший 20-гладкий делитель каждого из оставшихся.

Согласно методу Бернштейна, сначала необходимо вычислить произведение всех простых чисел, не превосходящих 20 (естественно, в нашем случае), которое равно 9699690, а затем найти остатки от деления этого произведения на каждое из исследуемых восьми чисел. Пусть $x$ — одно из этих чисел и $9699690 \bmod x = r$. Тогда $r = ab$, где $a = \text{НОД}(9699690, x)$ и $\text{НОД}(b, x) = 1$. Если наибольший показатель в каноническом разложении числа $x$ на простые множители не превосходит $2^k$, то $\text{НОД}(r^{2^k} \bmod x, x)$ и есть 20-гладкая часть числа $x$. Поскольку $2^{2^4} > 1008$, в нашем случае можно взять $k = 4$. Посмотрим, что будет происходить с числом $x = 1008$. Сначала мы получим $r = 714$. Затем вычисления $714^{2^i} \bmod 1008$ для $i = 1, 2, 3, 4$ дадут, соответственно, 756, 0, 0, 0. Разумеется, нам следовало бы остановиться при получении первого 0, поскольку это сразу означало, что число 1008 — 20-гладкое. Для $x = 1004$ при этом подходе мы получим $r = 46$, а остатки степеней будут равны 108, 620, 872, 356. Затем вычисление $\text{НОД}(356, 1004)$ дает число 4. Конечно, наши действия можно назвать «длинным окружным путем»! Но как будет показано далее, этот подход допускает красивое ускорение: просто надо работать не с каждым числом по отдельности, а со всеми вместе.

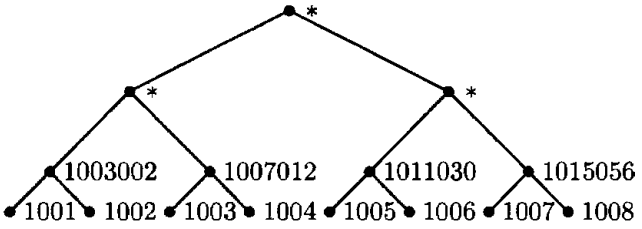
Следуя методу книги [Bernstein 2004e], построим двоичное дерево, соответствующее произведению 9699690 простых чисел, меньших 20, которое назовем «деревом произведений»: простые числа последовательно приписываются листьям дерева и парами перемножаются, а корню дерева соответствует полное произведение $P = 9699690$.




Дерево произведений для множества $\mathcal{P} = \{2, 3, 5, 7, 11, 13, 17, 19\}$

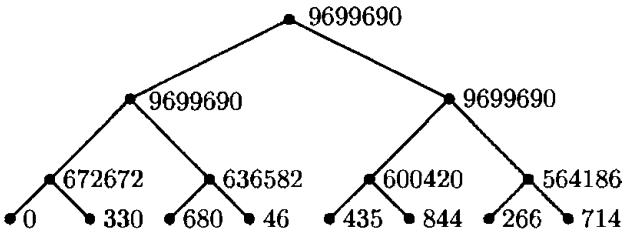
Нам требуется найти все вычеты $P \bmod x$, где $x$ пробегает множество чисел, исследуемых на гладкость. Если мы станем делать это для всех $x$ по очереди, то при больших значениях $P$ процесс затянется слишком надолго. Вместо этого мы начнем с перемножения чисел $x$ по аналогии с только что рассмотренными простыми числами. Будем строить дерево произведений, при этом поскольку нам не потребуются произведения, превосходящие $P$, будем обозначать такие

большие значения звездочкой. Рассмотрим дерево произведений $T$, соответствующее числам $1001, 1002, \ldots, 1008$.




Дерево произведений $T$ для множества $\mathcal{X} = \{1001, 1002, \ldots, 1008\}$

Затем, продолжая следовать методу книги [Bernstein 2004e], построим «дерево остатков» числа $P$ от деления на числа, приписанные соответствующим вершинам дерева $T$. Вообще говоря, если заданы натуральное число $P$ и дерево произведений $T$, то дерево остатков $P \bmod T$ получается путем замены чисел, приписанных вершинам дерева $T$, на их остатки от деления на $P$. При этом сначала вместо числа $R$, приписанного корню дерева $T$, ставится $P \bmod R$, а затем происходит спуск к листьям, во время которого каждое число заменяется на его остаток от деления на новое значение, приписанное его родителю (ближайшей вышестоящей вершине). В рассматриваемом нами случае, когда дерево произведений $T$ построено для чисел $1001, \ldots, 1008$, а $P = 9699690$, дерево остатков будет выглядеть следующим образом (очевидно, при этом все звездочки дерева $T$ заменяются на $P$).




Дерево остатков $P \bmod T$

На месте каждого из исследуемых на гладкость чисел $x$ в полученном дереве остатков стоит значение $P \bmod x$. Остается лишь возвести этот остаток в квадрат по модулю $x$ достаточное количество раз и, наконец, найти НОД полученного числа и $x$. Если ответ совпал с $x$, то это означает, что число $x$ — гладкое по отношению к простым, составляющим $P$, а меньшее значение само по себе равно наибольшему делителю числа $x$, обладающему заданным свойством. Вот псевдокод этого удивительного алгоритма.

**Алгоритм 3.3.1** (тест Бернштейна распознавания гладкости для множества чисел). Даны конечные множества натуральных чисел $\mathcal{X}$ и простых чисел $\mathcal{P}$. Для каждого элемента $x \in \mathcal{X}$ алгоритм возвращает его наибольший делитель, составленный из простых чисел множества $\mathcal{P}$.

1. [Вычислить деревья произведений]
   Вычислить дерево произведений для множества $\mathcal{P}$;
   Вычислить произведение $P$ всех элементов множества $\mathcal{P}$;
   // Число $P$ приписано корню дерева произведений для множества $\mathcal{P}$.
   Вычислить дерево произведений $T$ для множества $\mathcal{X}$ (только значения, не большие $P$);
2. [Вычислить дерево остатков]
   Вычислить дерево остатков $P \bmod T$;
   // Последовательность действий описана в тексте.
3. [Найти гладкие делители]
   Найти наименьшее натуральное число $k$, для которого $\max \mathcal{X} \leqslant 2^{2^k}$;
   for($x \in \mathcal{X}$){
   Найти $P \bmod x$ в дереве остатков $P \bmod T$;
   // Никаких дополнительных вычислений остатков не требуется.
   $r = P \bmod x$;
   $s = r^{2^k} \bmod x$;
   // Вычислить $s$ путем последовательного возведения в квадрат и взятия остатка.
   $g = \text{НОД}(s, x)$;
   return «наибольший делитель числа $x$, составленный из простых чисел множества $\mathcal{P}$, равен $g$»;
   }

Алгоритм Бернштейна 3.3.1 стал важным вкладом в инструментарий вычислительной теории чисел. Он может с успехом применяться для ускорения многих алгоритмов, использующих свойство гладкости. Примером того служит шаг [Разложить порядки] теста Аткина—Морейна на простоту (алгоритм 7.6.3). Более того, метод Бернштейна бывает полезен даже в тех ситуациях, когда для разложения на множители применяется просеивание, например, методом квадратичного решета или методом решета числового поля (см. гл. 6). Дело в том, что при работе этих алгоритмов доля гладких чисел может быть так мала, что выгоднее провести лишь частичное просеивание (забыть о малых простых числах в фактор-базе, из-за которых происходит бóльшая часть обращений к памяти) и настроить решето так, чтобы кандидаты с большим гладким делителем записывались в (существенно меньшее, но все еще большое) множество, к элементам которого затем применить алгоритм 3.3.1. Эта идея исключения малых простых чисел из решета описана еще в статье [Pomerance 1985], но алгоритм 3.3.1 позволяет сделать ее еще более эффективной.

## 3.4. Псевдопростые числа

Предположим, у нас есть теорема: *Если $n$ — простое число, то для числа $n$ справедливо $S$*, где $S$ — некоторое легко проверяемое арифметическое утверждение. Если нам дано большое число $n$ и мы желаем исследовать его на простоту, можно для начала проверить, справедливо высказывание $S$ о числе $n$ или нет. Если нет, то мы доказали составность числа $n$. Однако, если рассматриваемое утверждение о числе $n$ справедливо, то о простоте числа $n$ ничего с уверенностью утверждать нельзя. Таким образом, возникает понятие $S$-псевдопростого числа, которое означает составное, удовлетворяющее условию $S$.

Например, рассмотрим теорему: *Если $n$ — простое число, то $n$ или равно 2, или нечетно.* Конечно, это арифметическое свойство легко проверить для любого заданного числа $n$. Однако нетрудно видеть, что этот тест, скажем так, не очень эффективен, поскольку количество соответствующих псевдопростых чисел существенно превосходит количество истинных простых чисел. Таким образом, для содержательности понятия «псевдопростоты» необходимо, чтобы псевдопростых чисел было в определенном смысле «немного».

### 3.4.1. Псевдопростые числа Ферма

В основу многих теоретико-числовых алгоритмов заложен тот факт, что значение $a^b \pmod n$ можно быстро вычислить (см. алгоритм 2.1.5). Не менее популярно и применение малой теоремы Ферма в качестве средства распознавания простых и составных чисел.

**Теорема 3.4.1** (малая теорема Ферма). *Если $n$ — простое число, то для каждого натурального числа $a$ справедливо сравнение*

$$a^n \equiv a \pmod n. \tag{3.2}$$

Доказательство малой теоремы Ферма можно найти в любом учебнике по элементарной теории чисел. Одно из наиболее простых доказательств основано на индукции по $a$ и разложении бинома $(a+1)^n$.

Если число $a$ взаимно просто с $n$, разделив обе части сравнения (3.2) на $a$, получим

$$a^{n-1} \equiv 1 \pmod n. \tag{3.3}$$

Таким образом, сравнение (3.3) имеет место каждый раз, когда $n$ — простое число, не делящее $a$.

Будем говорить, что составное число $n$ является псевдопростым (Ферма) по основанию $a$, если выполнено сравнение (3.2). Например, $n = 91$ — псевдопростое число по основанию 3, поскольку оно составное и $3^{91} \equiv 3 \pmod{91}$. Аналогичным образом, 341 — псевдопростое число по основанию 2. Основание $a = 1$ интереса не представляет, поскольку каждое составное число является псевдопростым по основанию 1. Поэтому мы будем далее считать, что $a \geqslant 2$.

**Теорема 3.4.2.** *Для каждого фиксированного основания $a \geqslant 2$ количество псевдопростых чисел Ферма по основанию $a$, которые не превосходят $x$, есть*

$o(\pi(x))$ *при* $x \to \infty$. *Иными словами, псевдопростые числа Ферма редки по сравнению с простыми числами.*

Для псевдопростых чисел, определенных сравнением (3.3), эта теорема была впервые доказана в статье [Erdős 1950]. Для возможно более широкого класса псевдопростых чисел, определенных сравнением (3.2), эта теорема была впервые доказана в статье [Li 1997].

Теорема 3.4.2 говорит о том, что использование сравнения Ферма может быть очень результативным при решении вопроса о составности или простоте. Однако это было известно задолго до появления доказательства Эрдёша благодаря практическому применению этого сравнения.

Отметим, что нечетные числа $n$ удовлетворяют сравнению (3.3) при $a = n-1$, поэтому в этом случае новой информации о числе $n$ мы не получим. Если же сравнение (3.3) выполняется для пары $n, a$, где $1 < a < n-1$, мы будем говорить, что $n$ — вероятное простое число по основанию $a$. Таким образом, если число $n$ простое, то оно является вероятным простым числом по всем основаниям $a$, где $1 < a < n-1$. Теорема 3.4.2 утверждает, что при фиксированном выборе основания $a$ наиболее вероятные простые числа по основанию $a$ скорее всего являются простыми. Таким образом, мы имеем тест, различающий элементы множества, содержащего разреженное множество составных чисел, а также все простые числа, превосходящие $a+1$, и элементы множества всех остальных составных чисел, превосходящих $a+1$.

**Алгоритм 3.4.3** (тест для вероятного простого числа). Нам даны целые числа $n > 3$ и $a$, причем $2 \leqslant a \leqslant n-2$. Алгоритм выдает или ответ «$n$ — вероятное простое число по основанию $a$», или ответ «$n$ — составное число».

1. [Вычислить степенной вычет]
   $b = a^{n-1} \bmod n$; // Применить алгоритм 2.1.5.
2. [Вынести решение]
   if($b == 1$) return «$n$ — вероятное простое число по основанию $a$»;
   return «$n$ — составное число»;

Как сказано выше, псевдопростые числа по фиксированному основанию $a$ (т. е. вероятные простые числа, являющиеся составными) достаточно редки. Однако это не мешает существованию бесконечного множества таких чисел.

**Теорема 3.4.4.** *Для каждого целого числа* $a \geqslant 2$ *существует бесконечное множество псевдопростых чисел Ферма по основанию* $a$.

Доказательство. Наша цель — показать, что если нечетное простое число $p$ не делит $a^2-1$, то $n = \left(a^{2p}-1\right)/\left(a^2-1\right)$ — псевдопростое число по основанию $a$. Например, при $a = 2$ и $p = 5$ эта формула дает $n = 341$. Сначала отметим, что

$$n = \frac{a^p-1}{a-1} \cdot \frac{a^p+1}{a+1},$$

а значит, $n$ — составное число. Применяя для простого числа $p$ сравнение (3.2) и возводя в квадрат обе части равенства, получаем $a^{2p} \equiv a^2 \pmod p$. Следовательно, $p$ делит $a^{2p}-a^2$. Но поскольку по условию $p$ не делит $a^2-1$, а

$n-1=\left(a^{2p}-a^2\right)/\left(a^2-1\right)$, мы заключаем, что $p$ делит $n-1$. Кроме того, в силу сравнения

$$n-1\equiv a^{2p-2}+a^{2p-4}+\ldots+a^2,$$

мы видим, что $n-1$, помимо прочего, является и суммой четного количества слагаемых одинаковой четности, а значит, число $n-1$ четно. Итак, мы выяснили, что числа 2 и $p$ делят $n-1$, а следовательно, этим свойством обладает и $2p$. Значит, $a^{2p}-1$ делит $a^{n-1}-1$. Но $a^{2p}-1$ кратно $n$, поэтому имеет место сравнение (3.3), а также и сравнение (3.2). □

## 3.4.2. Числа Кармайкла

При поиске простого и быстрого метода отделения простых чисел от составных можно рассмотреть сочетание тестов Ферма для различных оснований $a$. Например, хотя 341 — псевдопростое число по основанию 2, оно не является таковым по основанию 3. А число 91 — псевдопростое по основанию 3, но не по основанию 2. Вдруг не существует таких составных чисел, которые являются псевдопростыми по основаниям 2 и 3 одновременно? А если такие составные числа есть, то существует ли такое конечное множество оснований, для которого не найдется псевдопростых чисел по всем основаниям этого множества? Было бы прекрасно получить положительный ответ, поскольку тогда у нас был бы простой алгоритм вычислений для проверки на простоту.

Однако число $561=3\cdot 11\cdot 17$ является не только псевдопростым Ферма по основаниям 2 и 3, но оно псевдопростое по *всем* основаниям $a$. Кажется невероятным, что такие числа существуют, но, и в самом деле, это так. Впервые они были найдены Р. Кармайклом в 1910 г. и с тех пор такие числа носят имя этого математика.

**Определение 3.4.5.** Составное число $n$, для которого при всех натуральных числах $a$ выполнено сравнение $a^n\equiv a\pmod n$, называется числом Кармайкла.

Числа Кармайкла легко распознать по разложению на простые множители.

**Теорема 3.4.6** (критерий Корсельта). *Целое число $n$ является числом Кармайкла тогда и только тогда, когда оно положительное, составное, бесквадратное и для каждого простого числа $p$, делящего $n$, число $p-1$ делит $n-1$.*

**Замечание.** А. Корсельт опубликовал этот критерий для чисел Кармайкла в 1899 г., за 11 лет до того, как Кармайкл привел пример такого числа. Возможно, Корсельт был уверен в отсутствии этих чисел и вывел свой критерий как первый шаг доказательства такого утверждения.

Доказательство. Пусть, сначала, $n$ — число Кармайкла. Тогда оно составное. Пусть $p$ — простой делитель числа $n$. Поскольку $p^n\equiv p\pmod n$, мы видим, что $p^2$ не делит $n$. Значит, $n$ свободно от квадратов. Пусть, далее, $a$ — первообразный корень по модулю $p$. Поскольку $a^n\equiv a\pmod n$, мы получаем $a^n\equiv a\pmod p$, откуда следует сравнение $a^{n-1}\equiv 1\pmod p$. Но $a\pmod p$ имеет порядок $p-1$, а следовательно, $p-1$ делит $n-1$.

Обратно, предположим теперь, что натуральное число $n$ составное, бесквадратное и для каждого простого числа $p$, делящего $n$, число $p-1$ делит $n-1$. Мы должны доказать, что сравнение $a^n \equiv a \pmod{n}$ выполняется при всех $a$. Поскольку $n$ свободно от квадратов, достаточно показать, что $a^n \equiv a \pmod{p}$ для всех $a$ и всех простых чисел $p$, делящих $n$. Итак, пусть $p \mid n$ и $a$ целое число. Если число $a$ не кратно $p$, то $a^{p-1} \equiv 1 \pmod{p}$ (согласно (3.3)), а поскольку $p-1$ делит $n-1$, мы получаем $a^{n-1} \equiv 1 \pmod{p}$. Таким образом, $a^n \equiv a \pmod{p}$. Но это сравнение, очевидно, имеет место и тогда, когда $a$ кратно $p$, а значит, и при всех $a$. Тем самым, теорема доказана. □

Бесконечно ли множество чисел Кармайкла? К огорчению создателей тестов на простоту, ответ снова положителен. Это было доказано в статье [Alford et al. 1994a]. П. Эрдёш в 1956 г. провел эвристические рассуждения в пользу того, что чисел Кармайкла не только бесконечно много, но они и не так редки, как можно было ожидать. Точнее говоря, если обозначить количество чисел Кармайкла, не превосходящих $x$, через $C(x)$, то Эрдёш высказал предположение о том, что для каждого $\varepsilon > 0$ существует такое число $x_0(\varepsilon)$, что $C(x) > x^{1-\varepsilon}$ при всех $x \geqslant x_0(\varepsilon)$. Доказательство Альфорда, Грэнвилля и Померанса развивает идеи Эрдёша и применяет новые соображения.

**Теорема 3.4.7** (Альфорд, Грэнвилль, Померанс). *Существует бесконечное множество чисел Кармайкла. Более того, при достаточно больших значениях $x$ количество $C(x)$ чисел Кармайкла, не превосходящих $x$, удовлетворяет неравенству $C(x) > x^{2/7}$.*

Доказательство выходит за рамки настоящей книги и может быть найдено в статье [Alford et al. 1994a].

Численное значение, скрывающееся за термином «достаточно большое значение» из теоремы 3.4.7, найдено не было, но, вероятно, здесь можно указать 96-е число Кармайкла 8719309. Вычисления из работы [Pinch 1993] подсказывают, что неравенство $C(x) > x^{1/3}$ может быть выполнено для всех $x \geqslant 10^{15}$. В частности, на промежутке $[1, 10^{15}]$ существуют ровно 105212 чисел Кармайкла. Хотя Эрдёш и предполагал, что $C(x) > x^{1-\varepsilon}$ при $x \geqslant x_0(\varepsilon)$, нам не известно ни одно численное значение $x$, при котором $C(x) > x^{1/2}$.

Существует ли «теорема о числах Кармайкла», которая подобно теореме о простых числах устанавливала бы асимптотическую формулу для $C(x)$? Пока отсутствует даже предположение о виде такой формулы. Однако выдвинута несколько более слабая гипотеза.

**Гипотеза 3.4.1** (Эрдёш, Померанс). *Количество $C(x)$ чисел Кармайкла, не превосходящих $x$, удовлетворяет асимптотическому соотношению*

$$C(x) = x^{1-(1+o(1))\ln\ln\ln x/\ln\ln x} \quad \textit{при} \quad x \to \infty.$$

Предполагается, что такая же формула имеет место и для $P_2(x)$ — количества псевдопростых чисел, не превосходящих $x$. В статье [Pomerance 1981] доказано, что при всех достаточно больших значениях $x$ справедливы неравенства

$$\begin{aligned} C(x) &< x^{1-\ln\ln\ln x/\ln\ln x}, \\ P_2(x) &< x^{1-\ln\ln\ln x/(2\ln\ln x)}. \end{aligned}$$

## 3.5. Вероятно простые числа и свидетели

Рассмотренное в предыдущем разделе понятие псевдопростого числа Ферма является удобным, поскольку свойство псевдопростоты легко проверяемо и для каждого основания $a > 1$ количество псевдопростых чисел мало́ по сравнению с количеством простых чисел (теорема 3.4.2). Однако существуют числа Кармайкла, т. е. составные числа, для которых сравнение (3.2) не позволяет распознать их составность. Как было сказано, чисел Кармайкла бесконечно много. Кроме того, существует также бесконечное множество чисел Кармайкла, у которых отсутствуют малые простые делители (см. [Alford et al. 1994b]), поэтому для таких чисел даже несколько более сильное сравнение (3.3) с вычислительной точки зрения не эффективно.

Нам бы очень хотелось иметь легкий тест, для которого не нашлось бы псевдопростых чисел. Поскольку это невозможно, мы готовы довольствоваться семейством тестов, таких, что каждое составное число не является псевдопростым для фиксированной положительной доли тестов этого семейства. Семейство тестов Ферма не удовлетворяет этому условию в силу бесконечности множества чисел Кармайкла. Однако нашей цели можно достичь, применив несколько иной вариант малой теоремы Ферма 3.4.1.

**Теорема 3.5.1.** *Пусть $n$ — нечетное простое число и $n - 1 = 2^s t$, где число $t$ нечетно. Если число $a$ не делится на $n$, то*

$$\begin{cases} \text{или } a^t \equiv 1 \pmod n, \\ \text{или } a^{2^i t} \equiv -1 \pmod n \text{ для некоторого } i, \text{ где } 0 \leqslant i \leqslant s-1. \end{cases} \tag{3.4}$$

Доказательство теоремы 3.5.1 основано лишь на применении малой теоремы Ферма 3.4.1 и того факта, что если $n$ — нечетное простое число, то единственными решениями сравнения $x^2 \equiv 1 \pmod n$ в $\mathbf{Z}_n$ являются $x \equiv \pm 1 \pmod n$. Доведение рассуждения до конца предоставляем читателю.

По аналогии с вероятными простыми числами определим теперь сильно вероятное простое число по основанию $a$. Это нечетное число $n > 3$, для которого (3.4) имеет место при всех $a$ из промежутка $1 < a < n - 1$. Поскольку каждое сильно вероятное псевдопростое число по основанию $a$ автоматически является и вероятным простым числом по основанию $a$, и поскольку каждое простое число, превосходящее $a + 1$, является сильно вероятным простым числом по основанию $a$, единственное отличие этих понятий заключено в том, что тест на сильно вероятное простое число пройдет меньше составных чисел.

**Алгоритм 3.5.2** (тест на сильно вероятное простое число). Нам дано нечетное число $n > 3$ в виде $n = 1 + 2^s t$, где число $t$ нечетно. Кроме того, нам задано целое число $a$ из промежутка $1 < a < n - 1$. Алгоритм выдает или ответ «$n$ — сильно вероятное простое число по основанию $a$», или ответ «$n$ — составное число».

1. [Нечетная часть числа $n - 1$]
   $b = a^t \bmod n$; // Применить алгоритм 2.1.5.
   if($b == 1$ or $b == n - 1$) return «$n$ — сильно вероятное простое число по основанию $a$»;

2. [Степень 2 в числе $n-1$]
for($j \in [1, s-1]$) { // $j$ — просто счетчик.
$b = b^2 \bmod n$;
if($b == n-1$) return «$n$ — сильно вероятное простое число по основанию $a$»;
}
return «$n$ — составное число»;

Этот тест был впервые описан в статье [Artjuhov 1966/67], а через десять лет он был заново разработан и популяризирован Дж. Селфриджем.

Как сказано выше, если для некоторого числа $a$ нечетное число $n$ не удовлетворяет условию (3.4), то $n$ — составное число. Например, в предыдущем разделе мы видели, что 341 — псевдопростое число по основанию 2. Но условие (3.4) *не* выполняется при $n = 341$ и $a = 2$. В самом деле, $340 = 2^2 \cdot 85$, $2^{85} \equiv 32 \pmod{341}$ и $2^{170} \equiv 1 \pmod{341}$. Кроме того, мы видим, что 32 является нетривиальным корнем из 1 по модулю 341.

Рассмотрим теперь пару $n = 91$ и $a = 10$. Поскольку $90 = 2^1 \cdot 45$ и $10^{45} \equiv -1 \pmod{91}$, условие (3.4) выполняется.

**Определение 3.5.3.** Будем говорить, что $n$ — сильно псевдопростое число по основанию $a$, если $n$ — нечетное составное число, $n - 1 = 2^s t$, где число $t$ нечетно, и имеет место условие (3.4).

Таким образом, 341 не является сильно псевдопростым числом по основанию 2, а 91 — сильно псевдопростое число по основанию 10. Дж. Селфридж применял теорему 3.5.1 в качестве теста для псевдопростых чисел в начале 1970-х гг., и именно он ввел понятие «сильно псевдопростое число». Очевидно, что если $n$ — сильно псевдопростое число по основанию $a$, то оно является и псевдопростым по основанию $a$. Как показывает пример $n = 341$ и $a = 2$, обратное утверждение неверно.

Пусть $n$ -- нечетное составное число. Положим

$$\mathcal{S}(n) = \{a \pmod n : n \text{ — сильно псевдопростое число по основанию } a\} \quad (3.5)$$

и пусть $S(n) = \#\mathcal{S}(n)$. Следующий результат был получен независимо в работах [Monier 1980] и [Rabin 1980].

**Теорема 3.5.4.** *Для каждого нечетного составного числа $n > 9$ справедливо неравенство $S(n) \leqslant \frac{1}{4}\varphi(n)$.*

Напомним, что функция Эйлера $\varphi(n)$ равна количеству целых чисел отрезка $[1, n]$, взаимно простых с числом $n$, т. е. порядку группы $\mathbf{Z}_n^*$. Если известно разложение числа $n$ на простые множители, то нетрудно вычислить $\varphi(n)$: справедлива формула $\varphi(n) = n \prod_{p \mid n}(1 - 1/p)$, где $p$ пробегает все простые делители числа $n$.

Перед тем как доказать теорему 3.5.4, объясним сначала значение этого результата. Если нам дано нечетное число $n$ и требуется исследовать его на простоту, мы можем сначала проверить условие (3.4) для некоторого числа $a$ из промежутка $1 < a < n - 1$. Если условие (3.4) не выполнено, то составность

числа $n$ доказана. Можно назвать число $a$ свидетелем составности числа $n$. Введем теперь строгое определение.

**Определение 3.5.5.** Пусть для нечетного составного числа $n$ и целого числа $a$ из отрезка $[1, n-1]$ не выполнено условие (3.4). Тогда будем говорить, что $a$ является свидетелем для числа $n$. Иными словами, для нечетного составного числа $n$, свидетелем является то основание, по которому число $n$ не является сильно псевдопростым.

Таким образом, для доказательства составности числа $n$ достаточно найти хотя бы одного свидетеля.

Теорема 3.5.4 утверждает, что если $n$ — нечетное составное число, то как минимум $3/4$ всех целых чисел отрезка $[1, n-1]$ являются свидетелями для $n$. Поскольку тест на сильно псевдопростое число допускает очень быстро работающую реализацию, проверить, является ли данное число $a$ свидетелем числа $n$, достаточно легко. Поэтому может показаться, что задача предъявить свидетеля для составного числа очень проста. Можно сказать, что это и в самом деле так, если воспользоваться вероятностным методом. Следующий алгоритм обычно называется «тестом Миллера—Рабина», хотя, как легко видеть, это просто сочетание алгоритма 3.5.2 со случайным выбором основания $a$. (Исходный тест Миллера [Miller 1976] был несколько более сложным и детерминированным, основанным на ERH. Приведенный ниже вероятностный алгоритм был предложен Рабином [Rabin 1976, 1980].)

**Алгоритм 3.5.6** (случайный тест на составность)**.** Нам задано нечетное число $n > 3$. Этот вероятностный алгоритм ищет свидетеля для $n$, чтобы доказать его составность. Если найден свидетель $a$, то выдается ответ ($a$, ДА). В противном случае выдается ответ ($a$, НЕТ).

1. [Выбор возможного свидетеля]
   Выбор случайного числа $a \in [2, n-2]$;
   Посредством алгоритма 3.5.2 решается, является ли $n$ сильно вероятным простым числом по основанию $a$;
2. [Объявления результата]
   if($n$ — сильно вероятное простое число по основанию $a$) return ($a$, НЕТ);
   return ($a$, ДА);

Согласно теореме 3.5.4, если $n > 9$ — нечетное составное число, то вероятность того, что алгоритму 3.5.6 не удастся найти свидетеля для $n$ меньше $1/4$. Кроме того, мы можем применить этот алгоритм и повторно. Вероятность неудачи при поиске свидетеля посредством $k$ (независимых) итераций алгоритма 3.5.6 меньше $1/4^k$. Очевидно, выбирая число $k$ большим, мы можем сделать эту вероятность пренебрежимо малой.

Алгоритм 3.5.6 — очень эффективное средство распознавания составных чисел. Но каков будет результат его работы, если мы подадим на вход нечетное простое число? Разумеется, свидетеля найти не удастся, поскольку в силу теоремы 3.5.1 у простых чисел свидетелей нет.

Пусть нам неизвестно, является данное большое нечетное число $n$ простым или составным. Предположим, что, скажем, 20 итераций алгоритма 3.5.6 не

позволили найти для $n$ свидетеля. Какой можно сделать вывод? Строго говоря, на этом основании ничего о простоте или составности числа $n$ утверждать нельзя. Конечно, представляется логичным *предположить*, что число $n$, скорее всего, простое. Вероятность того, что 20 итераций алгоритма 3.5.6 не позволят найти свидетеля для данного составного нечетного числа, не превосходит $4^{-20}$, что меньше, чем одна триллионная. Следовательно, число $n$ и в самом деле наверняка простое. Но его простота остается недоказанной и, более того, это число может оказаться и составным.

В гл. 4 описаны методы, позволяющие установить простоту тех чисел, которых мы пока можем в этом только подозревать. Однако на практике обычно достаточно часто используют числа, которые почти наверняка являются простыми, но их простота строго не доказана. Именно по этой причине многие исследователи называют алгоритм 3.5.6 «тестом на простоту». Вероятно, более правильно было бы вслед за А. Коэном назвать число, прошедшее этот тест, простым числом «промышленного уровня».

Следующий алгоритм можно применять для получения случайных чисел, которые, скорее всего, являются простыми.

**Алгоритм 3.5.7** (генерация простого числа «промышленного уровня»). Нам заданы целые числа $k \geqslant 3$ и $T \geqslant 1$. Этот вероятностный алгоритм генерирует $k$-битовое число (т. е. число из промежутка $\left[2^{k-1}, 2^k\right)$), которое не было признано составным после $T$ итераций алгоритма 3.5.6.

1 [Выбор кандидата]
Выбор случайного нечетного числа $n$ из промежутка $\left(2^{k-1}, 2^k\right)$.
2 [Выполнить тесты на сильно вероятное простое число]
for($1 \leqslant i \leqslant T$) { // $i$ — просто счетчик
Применяем алгоритм 3.5.6 для поиска свидетеля для $n$;
if(свидетель для $n$ найден) goto [Выбор кандидата];
}
return $n$; // $n$ — простое число «промышленного уровня».

Возникает интересный вопрос: какова вероятность того, что сгенерированное алгоритмом 3.5.7 число окажется составным? Обозначим эту вероятность через $P(k, T)$. Может показаться, что ответ на этот вопрос следует непосредственно из теоремы 3.5.4 и $P(k, T) \leqslant 4^{-T}$. Однако это рассуждение ошибочно. Положим $k = 500$, $T = 1$. Согласно теореме о простых числах 1.1.4 вероятность того, что случайное нечетное 500-битовое число окажется простым, примерно равна $1/173$. Поскольку, разумеется, скорее произойдет событие, вероятность которого равна $1/4$, чем то, вероятность которого равна $1/173$, может показаться, что алгоритм 3.5.7 с довольно высокой вероятностью будет порождать составные числа. Однако, на самом деле, теорема 3.5.4 дает оценку для самого плохого случая, а для большинства составных чисел доля свидетелей гораздо больше, чем $3/4$. В статье [Burthe 1996] показано, что и в самом деле $P(k, T) \leqslant 4^{-T}$.

Если число $k$ велико, алгоритм 3.5.7 приводит к хорошим результатам даже при $T = 1$. В статье [Damgård et al. 1993] показано, что $P(k, 1) < k^2 4^{2-\sqrt{k}}$. Для

конкретных значений $k$ в этой статье указаны даже более сильные оценки, например, $P(500,1) < 4^{-28}$. Таким образом, если случайное нечетное 500-битовое число проходит только одну итерацию случайного теста на сильно вероятное простое число, то вероятность того, что это число составное, пренебрежимо мала, так что его можно уверенно считать «простым» почти во всех практических приложениях, за исключением самых чувствительных.

Доказательству теоремы 3.5.4 предпошлем несколько лемм.

**Лемма 3.5.8.** *Пусть $n$ — нечетное составное число, $n-1 = 2^s t$, где число $t$ нечетно. Обозначим через $\nu(n)$ наибольшее целое число, такое, что $2^{\nu(n)}$ делит $p-1$ для всех простых чисел $p$, делящих $n$. Если $n$ — сильно псевдопростое число по основанию $a$, то справедливо сравнение $a^{2^{\nu(n)-1}t} \equiv \pm 1 \pmod n$.*

Доказательство. Если $a^t \equiv 1 \pmod n$, то утверждение леммы очевидно. Предположим, что $a^{2^i t} \equiv -1 \pmod n$ и пусть простое число $p$ делит $n$. Тогда $a^{2^i t} \equiv -1 \pmod p$. Если порядок $a \pmod p$ равен $k$ (т. е. $k$ есть наименьший положительный показатель, для которого $a^k \equiv 1 \pmod p$), то $k$ делит $2^{i+1}t$, но не делит $2^i t$. Следовательно, точная степень 2 в каноническом разложении числа $k$ равна $2^{i+1}$. Но мы также имеем, что $k$ делит $p-1$, так что $2^{i+1} \mid p-1$. Поскольку это выполнено для всех простых чисел $p$, делящих $n$, мы получаем неравенство $i+1 \leqslant \nu(n)$. Таким образом, если $i+1 < \nu(n)$, то $a^{2^{\nu(n)-1}t} \equiv 1 \pmod n$, а если $i+1 = \nu(n)$, то $a^{2^{\nu(n)-1}t} \equiv -1 \pmod n$. □

В следующей лемме нам потребуются обозначения

$$\overline{S}(n) = \left\{ a \pmod n : a^{2^{\nu(n)-1}t} \equiv \pm 1 \pmod n \right\}, \quad \overline{S}(n) = \#\overline{S}(n). \tag{3.6}$$

**Лемма 3.5.9.** *Сохраним введенные выше обозначения (см. лемму 3.5.8 и (3.6)). Пусть $\omega(n)$ обозначает количество различных простых делителей числа $n$. Тогда справедливо равенство*

$$\overline{S}(n) = 2 \cdot 2^{(\nu(n)-1)\omega(n)} \prod_{p \mid n} \text{НОД}(t, p-1).$$

Доказательство. Пусть $m = 2^{\nu(n)-1}t$. Пусть, далее, каноническое разложение на множители числа $n$ имеет вид $p_1^{j_1} p_2^{j_2} \dots p_k^{j_k}$, где $k = \omega(n)$. Как известно, $a^m \equiv 1 \pmod n$ тогда и только тогда, когда $a^m \equiv 1 \pmod{p_i^{j_i}}$ при всех $i = 1, 2, \dots, k$. Для нечетного простого числа $p$ и натурального числа $j$ группа $\mathbf{Z}_{p^j}^*$ приведенных вычетов по модулю $p^j$ является циклической порядка $p^{j-1}(p-1)$, так как по модулю $p^j$ существует первообразный корень. (Последний факт упоминался и в разд. 1.4.3. Доказательство можно найти в большинстве учебников по элементарной теории чисел. См. также теорему 2.2.5.) Следовательно, число решений $a \pmod{p_i^{j_i}}$ сравнения $a^m \equiv 1 \pmod{p_i^{j_i}}$ равно

$$\text{НОД}(m, p_i^{j_i-1}(p_i-1)) = \text{НОД}(m, p_i - 1) = 2^{\nu(n)-1} \cdot \text{НОД}(t, p_i - 1).$$

(Отметим, что первое равенство следует из того, что $m$ делит $n-1$, а следовательно, не кратно $p_i$.) Далее, согласно китайской теореме об остатках, число

решений $a \pmod n$ сравнения $a^m \equiv 1 \pmod n$ равно

$$\prod_{i=1}^{k}\Bigl(2^{\nu(n)-1}\cdot \text{НОД}\,(t, p_i - 1)\Bigr) = 2^{(\nu(n)-1)\omega(n)} \prod_{p \mid n} \text{НОД}\,(t, p-1).$$

Для завершения доказательства мы должны показать, что существует ровно такое же число решений сравнения $a^m \equiv -1 \pmod n$. Заметим, что $a^m \equiv -1 \pmod{p_i^{j_i}}$ тогда и только тогда, когда

$$a^{2m} \equiv 1 \pmod{p_i^{j_i}} \quad \text{и} \quad a^m \not\equiv 1 \pmod{p_i^{j_i}}.$$

Поскольку $2^{\nu(n)}$ делит $p_i - 1$, как и выше мы получаем, что число решений сравнения $a^m \equiv -1 \pmod{p_i^{j_i}}$ равно

$$2^{\nu(n)}\cdot \text{НОД}\,(t, p_i - 1) - 2^{\nu(n)-1}\cdot \text{НОД}\,(t, p_i - 1) = 2^{\nu(n)-1}\cdot \text{НОД}\,(t, p_i - 1).$$

Следовательно, числа решений сравнений $a^m \equiv 1 \pmod n$ и $a^m \equiv -1 \pmod n$ совпадают. Лемма доказана. □

Доказательство теоремы 3.5.4. Согласно лемме 3.5.8 нам достаточно доказать неравенство $\overline{S}(n)/\varphi(n) \leqslant 1/4$ для всех нечетных составных чисел $n$, превосходящих 9. В силу леммы 3.5.9

$$\frac{\varphi(n)}{\overline{S}(n)} = \frac{1}{2}\prod_{p^a \parallel n} p^{a-1}\frac{p-1}{2^{\nu(n)-1}\text{НОД}\,(t, p-1)},$$

где обозначение $p^a \parallel n$ означает, что $p^a$ есть степень простого числа $p$ в каноническом разложении числа $n$ (так называемая точная степень). Каждый множитель $(p-1)/(2^{\nu(n)-1}\text{НОД}\,(t, p-1))$ является четным числом, а значит, $\varphi(n)/\overline{S}(n)$ — целое число. Кроме того, если $\omega(n) \geqslant 3$, то $\varphi(n)/\overline{S}(n) \geqslant 4$. Если же $\omega(n) = 2$ и число $n$ не свободно от квадратов, то произведение различных $p^{a-1}$ как минимум равно 3, так что $\varphi(n)/\overline{S}(n) \geqslant 6$.

Пусть теперь $n = pq$, где $p < q$ — простые числа. Если $2^{\nu(n)+1} \mid q-1$, то $2^{\nu(n)-1}\text{НОД}\,(t, q-1) \leqslant (q-1)/4$ и $\varphi(n)/\overline{S}(n) \geqslant 4$. Следовательно, можно считать, что $2^{\nu(n)} \parallel q-1$. Заметим, что $n - 1 \equiv p - 1 \pmod{q-1}$, а значит, $q-1$ не делит $n-1$. Это означает, что найдется такое нечетное простое число, точная степень которого для числа $q-1$ превосходит эту степень для числа $n-1$, т. е. $2^{\nu(n)-1}\text{НОД}\,(t, q-1) \leqslant (q-1)/6$. Следовательно, и в этом случае $\varphi(n)/\overline{S}(n) \geqslant 6$.

Наконец, пусть $n = p^a$, где $a \geqslant 2$. Тогда $\varphi(n)/\overline{S}(n) = p^{a-1}$, а следовательно, $\varphi(n)/\overline{S}(n) \geqslant 5$, за исключением случая $p^a = 9$. □

### 3.5.1. Наименьший свидетель для $n$

Согласно теореме 3.5.4 нечетное составное число $n$ имеет не менее $3n/4$ свидетелей на отрезке $[1, n-1]$. Обозначим наименьший из этих свидетелей через $W(n)$. Тогда $W(n) \geqslant 2$. На самом деле, почти для всех составных чисел $n$ справедливо равенство $W(n) = 2$ (это непосредственно вытекает из теоремы 3.4.2).

Следующая теорема устанавливает неравенство $W(n) \geqslant 3$ для бесконечного множества нечетных составных чисел $n$.

**Теорема 3.5.10.** *Если простое число $p$ превосходит 5, то $n = (4^p + 1)/5$ — сильно псевдопростое число по основанию 2, так что $W(n) \geqslant 3$.*

Доказательство. Сначала покажем, что число $n$ составное. Поскольку $4^p \equiv (-1)^p \equiv -1 \pmod 5$, число $n$ целое, причем тот факт, что оно составное, следует из разложения

$$4^p + 1 = (2^p - 2^{(p+1)/2} + 1)(2^p + 2^{(p+1)/2} + 1).$$

Заметим, что $2^{2p} \equiv -1 \pmod n$, а значит, если число $m$ нечетно, то $2^{2pm} \equiv -1 \pmod n$. Но $n - 1 = 2^2 t$, где нечетное число $t$ делится на $p$ (в силу малой теоремы Ферма 3.4.1). Таким образом, $2^{2t} \equiv -1 \pmod n$, а следовательно, $n$ — сильно псевдопростое число по основанию 2. □

Возникает естественный вопрос: может ли функция $W(n)$ принимать сколь угодно большие значения? Можно сказать, что этот вопрос имеет решающее значение. Дело в том, что если бы нашлось такое не слишком большое число $B$, что для каждого нечетного составного числа $n$ выполнялось неравенство $W(n) \leqslant B$, то вся проблема проверки простоты стала бы тривиальной: достаточно было бы просто перебрать все числа $a \leqslant B$ и если для каждого из них выполнено условие (3.4), то число $n$ можно было с уверенностью считать простым. Но, к сожалению, такого числа $B$ не существует, а в работе [Alford et al. 1994b] доказан следующий результат.

**Теорема 3.5.11.** *Существует бесконечное множество нечетных составных чисел $n$, для которых*

$$W(n) > (\ln n)^{1/(3 \ln \ln \ln n)}.$$

*Более того, количество таких чисел $n$, не превосходящих $x$, при достаточно больших значениях $x$ превосходит*

$$x^{1/(35 \ln \ln \ln x)}.$$

Хотя универсальной постоянной $B$ не существует, возможно ли найти такую медленно растущую функцию, чтобы она всегда превосходила $W(n)$? На основании результата статьи [Miller 1976] в диссертации [Bach 1985] доказан такой факт.

**Теорема 3.5.12.** *При условии расширенной гипотезы Римана (ERH) для всех нечетных составных чисел $n$ справедливо неравенство $W(n) < 2 \ln^2 n$.*

Доказательство. Пусть $n$ — нечетное составное число. Согласно упр. 3.19, если число $n$ делится на квадрат простого числа, то $W(n) < \ln^2 n$, причем этот результат не зависит ни от какой недоказанной гипотезы. Следовательно, мы можем полагать, что число $n$ свободно от квадратов. Пусть простое число $p$ делит $n$, причем $p - 1 = 2^{s'} t'$ и число $t'$ нечетно. Тогда те же рассуждения, которые были проведены при доказательстве леммы 3.5.8, позволяют установить, что если выполнено условие (3.4), то $(a/p) = -1$ тогда и только тогда, когда $a^{2^{s'-1} t} \equiv -1 \pmod n$. Поскольку число $n$ нечетное, составное

и бесквадратное, оно должно делиться на два различных нечетных простых числа, скажем, $p_1$ и $p_2$. Пусть $p_i - 1 = 2^{s_i}t_i$, где числа $t_i$ нечетны, $i = 1, 2$, причем $s_1 \leqslant s_2$. Положим $\chi_1(m) = (m/p_1p_2)$, $\chi_2(m) = (m/p_2)$, так что $\chi_1$ является характером по модулю $p_1p_2$, а $\chi_2$ — характером по модулю $p_2$. Сначала рассмотрим случай $s_1 = s_2$. При условии справедливости расширенной гипотезы Римана теорема 1.4.5 утверждает существование такого натурального числа $m < 2\ln^2(p_1p_2) \leqslant 2\ln^2 n$, что $\chi_1(m) \neq 1$. Тогда $\chi_1(m) = 0$ или $-1$. Если $\chi_1(m) = 0$, то $m$ кратно $p_1$ или $p_2$, откуда следует, что число $m$ является свидетелем для $n$. Пусть $\chi_1(m) = -1$, так что или $(m/p_1) = 1$, $(m/p_2) = -1$ или наоборот. Без ограничения общности будем считать, что выполнены первые равенства. Тогда, как отмечалось выше, если выполнено условие (3.4), то $m^{2^{s_2-1}t} \equiv -1 \pmod{n}$, что в свою очередь влечет равенство $(m/p_1) = -1$, поскольку $s_1 = s_2$. Полученное противоречие означает, что $m$ является свидетелем для $n$. Перейдем к случаю $s_1 < s_2$. Снова по теореме 1.4.5 существует такое натуральное число $m < 2\ln^2 p_2 < 2\ln^2 n$, что $(m/p_2) = \chi_2(m) \neq 1$. Если $(m/p_2) = 0$, то $m$ делится на $p_2$ и является свидетелем. Если $(m/p_2) = -1$, то, как и выше, если $m$ не является свидетелем, то $m^{2^{s_2-1}t} \equiv -1 \pmod{n}$. Тогда по лемме 3.5.8 получаем $2^{s_2} \mid p_1 - 1$, так что $s_2 \leqslant s_1$ — противоречие. Таким образом, $m$ является свидетелем для $n$. □

Можно задаться вопросом: а что доказано безусловно? Очевидно, что $W(n) \leqslant n^{1/2}$, поскольку наименьший простой делитель нечетного составного числа $n$ является свидетелем для $n$. В статье [Burthe 1997] показано, что $W(n) \leqslant n^{c+o(1)}$ при $n \to \infty$ по множеству нечетных составных чисел, где $c = 1/(6\sqrt{e})$. Хиз-Браун (см. [Balasubramanian and Nagaraj 1997]) недавно заменил это значение на $c = 1/10.82$.

В заключение настоящего раздела приведем детерминистический тест Миллера на простоту. Он основан на теореме 3.5.12 и при условии расширенной гипотезы Римана устанавливает простоту за полиномиальное время.

**Алгоритм 3.5.13** (тест Миллера на простоту). Нам задано нечетное число $n > 1$. Этот алгоритм устанавливает, является ли число $n$ простым (ДА) или нет (НЕТ). Если возвращен ответ «НЕТ», то число $n$ непременно составное. Если же получен ответ «ДА», то или число $n$ простое, или расширенная гипотеза Римана неверна.

1. [Граница для свидетелей]
   $W = \min\{\lfloor 2\ln^2 n \rfloor, n-1\}$;
2. [Тесты на сильно вероятное простое число]
   for($2 \leqslant a \leqslant W$) {
      При помощи алгоритма 3.5.2 проверяем, является ли число $n$ сильно вероятным простым числом по основанию $a$;
      if($n$ не является сильно вероятным простым числом по основанию $a$)
         return НЕТ;
   }
   return ДА;

## 3.6. Псевдопростые числа Люка

Многие идеи предыдущих двух разделов можно обобщить и рассмотреть конечные поля. По традиции понятие псевдопростых чисел Люка вводят на языке рекуррентных последовательностей второго порядка. Но более выгодным является рассмотрение этого понятия с точки зрения теории конечных полей, и не потому, что так сейчас модно, а поскольку в этом случае применяемые методы не будут казаться слишком надуманными, и их можно будет обобщить на случай полей более высоких порядков.

### 3.6.1. Псевдопростые числа Фибоначчи и Люка

Последовательность $0, 1, 1, 2, 3, 5, \ldots$ чисел **Фибоначчи** ($j$-й член которой мы будем обозначать через $u_j$, начиная с $j = 0$) обладает следующим интересным свойством, связанным с делимостью на простые числа.

**Теорема 3.6.1.** *Если $n$ — простое число, то*

$$u_{n-\varepsilon_n} \equiv 0 \pmod{n}, \tag{3.7}$$

*где $\varepsilon_n = 1$ при $n \equiv \pm 1 \pmod 5$, $\varepsilon_n = -1$ при $n \equiv \pm 2 \pmod 5$ и $\varepsilon_n = 0$ при $n \equiv 0 \pmod 5$.*

**Замечание.** Должно быть, читатель опознал функцию $\varepsilon_n$: это символ Лежандра $\left(\frac{n}{5}\right)$ (см. определение 2.3.2).

**Определение 3.6.2.** Будем называть составное число $n$ псевдопростым Фибоначчи, если для него справедливо сравнение (3.7).

Например, наименьшее псевдопростое число **Фибоначчи**, взаимно простое с 10, равно 323.

Тест для псевдопростого числа **Фибоначчи** описан не из праздного любопытства. Как мы увидим ниже, он применим к очень большим числам. На самом деле, проверка, является ли данное число псевдопростым **Фибоначчи**, занимает примерно в два раза больше времени, чем стандартный тест на псевдопростоту, а если это число составное и сравнимо с $\pm 2 \pmod 5$, то одновременное применение теста Фибоначчи и обычного теста на псевдопростоту по основанию 2 позволяет еще больше повысить эффективность. Отметим, что нам не известно ни одного числа $n \equiv \pm 2 \pmod 5$, которое было бы одновременно псевдопростым по основанию 2 и псевдопростым числом **Фибоначчи** (см. упр. 3.41).

Доказательство теоремы 3.6.1 позволяет без дополнительных усилий получить и более общий результат. Последовательность чисел **Фибоначчи** удовлетворяет рекуррентному соотношению $u_j = u_{j-1} + u_{j-2}$ с характеристическим многочленом $x^2 - x - 1$. Рассмотрим более общий случай рекуррентных последовательностей с многочленом $f(x) = x^2 - ax + b$, где $a$ и $b$ — целые числа,

причем $\Delta = a^2 - 4b$ не есть квадрат целого числа. Положим

$$U_j = U_j(a,b) = \frac{x^j - (a-x)^j}{x-(a-x)} \pmod{f(x)},$$
$$V_j = V_j(a,b) = x^j + (a-x)^j \pmod{f(x)}, \tag{3.8}$$

т. е. мы берем остатки от деления соответствующих многочленов на $f(x)$ в кольце $\mathbf{Z}[x]$. Обе последовательности $(U_j)$, $(V_j)$ удовлетворяют рекуррентному соотношению с характеристическим многочленом $x^2 - ax + b$, а именно,

$$U_j = aU_{j-1} - bU_{j-2}, \quad V_j = aV_{j-1} - bV_{j-2},$$

причем в силу (3.8) начальные значения таковы:

$$U_0 = 0,\ U_1 = 1, \quad V_0 = 2,\ V_1 = a.$$

Если это не было хорошо видно из равенств (3.8), то теперь совершенно ясно, что $(U_j)$, $(V_j)$ — *целочисленные* последовательности.

По аналогии с теоремой 3.6.1 справедлив следующий результат. В частности, саму теорему 3.6.1 можно рассматривать как его частный случай при $a = 1$ и $b = -1$.

**Теорема 3.6.3.** *Предположим, что числа $a, b, \Delta$ определены выше и зададим последовательности $(U_j)$, $(V_j)$ равенствами* (3.8). *Тогда, если простое число $p$ таково, что* $\text{НОД}(p, 2b\Delta) = 1$, *то*

$$U_{p-\left(\frac{\Delta}{p}\right)} \equiv 0 \pmod{p}. \tag{3.9}$$

Заметим, что если $\Delta = 5$ и число $p$ нечетно, то $\left(\frac{5}{p}\right) = \left(\frac{p}{5}\right)$, что согласуется с замечанием, сделанным после теоремы 3.6.1. Поскольку в том случае, когда $n$ — нечетное простое число, символ Якоби $\left(\frac{\Delta}{n}\right)$ (см. определение 2.3.3) совпадает с символом Лежандра, можно превратить теорему 3.6.3 в тест на псевдопростоту.

**Определение 3.6.4.** Будем говорить, что составное число $n$, для которого $\text{НОД}(n, 2b\Delta) = 1$, является псевдопростым числом Люка по отношению к трехчлену $x^2 - ax + b$, если справедливо сравнение $U_{n-\left(\frac{\Delta}{n}\right)} \equiv 0 \pmod{n}$.

Поскольку последовательность $(U_j)$ построена при помощи приведения многочленов по модулю $x^2 - ax + b$, а теорема 3.6.3 и определение 3.6.4 связаны с приведением этих последовательностей по модулю $n$, на самом деле мы работаем с элементами кольца $R = \mathbf{Z}_n[x]/(x^2 - ax + b)$. Расшифруем это обозначение. Для начала приведем полный список представителей всех смежных классов, входящих в наше кольцо:

$$\{i + jx : i, j \text{ — целые числа, } 0 \leqslant i, j \leqslant n-1\}.$$

Эти элементы можно складывать как векторы $(\text{mod } n)$ и умножать с использованием равенства $x^2 = ax - b$. Строго говоря,

$$\begin{aligned}(i_1 + j_1 x) + (i_2 + j_2 x) &= i_3 + j_3 x,\\ (i_1 + j_1 x)(i_2 + j_2 x) &= i_4 + j_4 x,\end{aligned}$$

где

$$\begin{aligned} i_3 &= i_1 + i_2 \pmod n, \qquad j_3 = j_1 + j_2 \pmod n, \\ i_4 &= i_1 i_2 - b j_1 j_2 \pmod n, \; j_4 = i_1 j_2 + i_2 j_1 + a j_1 j_2 \pmod n. \end{aligned}$$

Доказательство теоремы 3.6.3. Пусть нечетное простое число $p$ таково, что $\left(\frac{\Delta}{p}\right) = -1$. Тогда $\Delta$ не является квадратом в $\mathbf{Z}_p$, а следовательно, многочлен $x^2 - ax + b$, дискриминант которого равен $\Delta$, неприводим над $\mathbf{Z}_p$. Следовательно, кольцо $R = \mathbf{Z}_p[x]/(x^2 - ax + b)$ является полем из $p^2$ элементов и изоморфно полю $\mathbf{F}_{p^2}$. Подполе $\mathbf{Z}_p$ $(= \mathbf{F}_p)$ состоит из тех смежных классов, которые имеют представителя вида $i + jx$, где $j = 0$.

Определенная в поле $\mathbf{F}_{p^2}$ функция $\sigma$ возведения в $p$-ю степень (известная как автоморфизм Фробениуса) имеет следующие удобные свойства, которые легко установить при помощи бинома Ньютона и малой теоремы Ферма (см. (3.2)): $\sigma(u+v) = \sigma(u) + \sigma(v)$, $\sigma(uv) = \sigma(u)\sigma(v)$, причем $\sigma(u) = u$ тогда и только тогда, когда $u$ принадлежит подполю $\mathbf{Z}_p$.

Мы прибегли к рассмотрению поля $\mathbf{F}_{p^2}$ для того, чтобы у многочлена $x^2 - ax + b$, появились корни, отсутствовавшие в $\mathbf{Z}_p$. Какие же представители $i + jx$ смежных классов им соответствуют? Это сам $x$, а также $a - x$ $(= a + (p-1)x)$. Поскольку $x$ и $a - x$ не принадлежат $\mathbf{Z}_p$ и функция $\sigma$ должна переставлять корни многочлена $f(x) = x^2 - ax + b$, мы получаем:

$$\text{если } \left(\frac{\Delta}{p}\right) = -1, \text{ то } \begin{cases} x^p \equiv a - x \pmod{(f(x), p)}, \\ (a-x)^p \equiv x \pmod{(f(x), p)}. \end{cases} \tag{3.10}$$

Следовательно, $x^{p+1} - (a-x)^{p+1} \equiv x(a-x) - (a-x)x \equiv 0 \pmod{(f(x), p)}$, поэтому в силу (3.8) справедливо сравнение $U_{p+1} \equiv 0 \pmod p$.

Доказательство сравнения (3.9) упрощается в том случае, когда простое число $p$ таково, что $\left(\frac{\Delta}{p}\right) = 1$. Дело в том, что тогда многочлен $x^2 - ax + b$ имеет два корня в $\mathbf{Z}_p$, поэтому кольцо $R = \mathbf{Z}_p[x]/(x^2 - ax + b)$ не является полем, а изоморфно кольцу $\mathbf{Z}_p \times \mathbf{Z}_p$, в котором каждый элемент равен себе в $p$-й степени. Поэтому

$$\text{если } \left(\frac{\Delta}{p}\right) = 1, \text{ то } \begin{cases} x^p \equiv x \pmod{(f(x), p)}, \\ (a-x)^p \equiv a - x \pmod{(f(x), p)}. \end{cases} \tag{3.11}$$

Заметим, далее, что из условия НОД$(p, b) = 1$ следует обратимость в $R$ элементов $x$ и $a - x$, так как $x(a-x) \equiv b \pmod{f(x)}$. Таким образом, в кольце $R$ имеем $x^{p-1} = (a-x)^{p-1} = 1$. Следовательно, в силу (3.8) мы получаем $U_{p-1} \equiv 0 \pmod p$. Тем самым теорема 3.6.3 доказана. □

Как показывает упр. 3.26, при работе с псевдопростыми числами Люка удобно отбрасывать многочлен $x^2 - x + 1$. Аналогичная проблема связана и с многочленом $x^2 + x + 1$, который также приходится отбросить. Однако все остальные многочлены с дискриминантами, не равными квадрату, нам, тем не менее, подходят. (Существуют ровно два неприводимых над полем рациональ-

ных чисел многочлена со старшим коэффициентом единица[1], корни которых являются одновременно и корнями из 1: $x^2 \pm x + 1$.)

## 3.6.2. Тест Грэнтхэма—Фробениуса

Ключевая роль автоморфизма Фробениуса (возведения в $p$-ю степень) в тесте Люка была положена в основу нового теста Грэнтхэма. Тест допускает произвольные многочлены на месте многочлена $x^2 - ax + b$, но даже в случае квадратичных многочленов он сильнее, чем тест Люка. Одним из преимуществ подхода Грэнтхэма является то, что он не привязан к рекуррентным последовательностям. Далее мы описываем принадлежащий ему тест для квадратичных многочленов. Несколько слов относительно общего теста сказано в разд. 3.6.5. Дополнительную информацию о псевдопростых числах Фробениуса см. в работе [Grantham 2001].

При доказательстве теоремы 3.6.3 мы вывели утверждения (3.10) и (3.11), но использовали лишь часть заложенной в них информации, а тест Фробениуса извлекает из них все возможное.

**Определение 3.6.5.** Пусть целые числа $a$ и $b$ таковы, что $\Delta = a^2 - 4b$ не равно квадрату целого числа. Будем говорить, что составное число $n$, для которого $\text{НОД}(n, 2b\Delta) = 1$, является псевдопростым числом Фробениуса по отношению к трехчлену $f(x) = x^2 - ax + b$, если справедливо сравнение

$$x^n \equiv \begin{cases} a - x \pmod{(f(x), n)} & \text{при } \left(\frac{\Delta}{n}\right) = -1, \\ x \pmod{(f(x), n)} & \text{при } \left(\frac{\Delta}{n}\right) = 1. \end{cases} \tag{3.12}$$

На первый взгляд, может показаться, что мы по-прежнему отбрасываем половину утверждений (3.10) и (3.11), но это не так (см. упр. 3.27).

Нетрудно указать критерий для псевдопростых чисел Фробениуса по отношению к квадратному трехчлену в терминах последовательностей Люка $(U_m)$, $(V_m)$.

**Теорема 3.6.6.** *Если целые числа $a$ и $b$ таковы, что $\Delta = a^2 - 4b$ не равно квадрату целого числа, и составное число $n$ таково, что* $\text{НОД}(n, 2b\Delta) = 1$, *то $n$ является псевдопростым Фробениуса по отношению к $x^2 - ax + b$ тогда и только тогда, когда*

$$U_{n-\left(\frac{\Delta}{n}\right)} \equiv 0 \pmod{n} \quad \text{и} \quad V_{n-\left(\frac{\Delta}{n}\right)} \equiv \begin{cases} 2b & \text{при } \left(\frac{\Delta}{n}\right) = -1, \\ 2 & \text{при } \left(\frac{\Delta}{n}\right) = 1. \end{cases}$$

Доказательство. Положим $f(x) = x^2 - ax + b$. Из равенств (3.8) сразу следует сравнение

$$2x^m \equiv (2x - a)U_m + V_m \pmod{(f(x), n)}.$$

Тогда из приведенных в условии теоремы сравнений получим $x^{n+1} \equiv b \pmod{(f(x), n)}$ при $\left(\frac{\Delta}{n}\right) = -1$ и $x^{n-1} \equiv 1 \pmod{(f(x), n)}$ при $\left(\frac{\Delta}{n}\right) = 1$. В последнем случае немедленно заключаем, что $x^n \equiv x \pmod{(f(x), n)}$, а в первом в

[1] Далее такие многочлены будем называть «нормированными». — *Прим. перев.*

силу сравнения $x(a-x) \equiv b \pmod{(f(x), n)}$ получим $x^n \equiv a-x \pmod{(f(x), n)}$. Это означает, что $n$ — псевдопростое число Фробениуса по отношению к $f(x)$.

Пусть теперь $n$ — псевдопростое число Фробениуса по отношению к $f(x)$. Согласно упр. 3.27, $n$ — псевдопростое число Люка по отношению к $f(x)$, т. е. $U_{n-\left(\frac{\Delta}{n}\right)} \equiv 0 \pmod{n}$. Поэтому в силу приведенного выше сравнения получим

$$2x^{n-\left(\frac{\Delta}{n}\right)} \equiv V_{n-\left(\frac{\Delta}{n}\right)} \pmod{(f(x), n)}.$$

Предположим, что $\left(\frac{\Delta}{n}\right) = -1$. Тогда $x^{n+1} \equiv (a-x)x \equiv b \pmod{(f(x), n)}$, а значит, $V_{n+1} \equiv 2b \pmod{n}$. Наконец, пусть $\left(\frac{\Delta}{n}\right) = 1$. Тогда, поскольку элемент $x$ обратим по модулю $(f(x), n)$, мы получаем $x^{n-1} \equiv 1 \pmod{(f(x), n)}$, а следовательно, $V_{n-1} \equiv 2 \pmod{n}$. □

Наименьшее псевдопростое число Фробениуса $n$ по отношению к $x^2 - x - 1$ равно 4181 (девятнадцатое число Фибоначчи), а первым числом, для которого $\left(\frac{5}{n}\right) = -1$, является 5777. Отсюда видно, что не всякое псевдопростое число Люка является псевдопростым числом Фробениуса, т. е. тест Фробениуса более сильный. И в самом деле, тест для псевдопростых чисел Фробениуса может быть чрезвычайно эффективным. Например, для многочлена $x^2+5x+5$ не известно ни одного примера псевдопростого числа Фробениуса $n$, для которого $\left(\frac{5}{n}\right) = -1$, хотя есть предположение, что такие числа существуют (см. упр. 3.42).

### 3.6.3. Реализация теста Люка и квадратичного теста Фробениуса

Оказывается, что существует реализация теста Люка, работающая примерно в два раза дольше обычного теста на псевдопростоту, а тест Фробениуса допускает реализацию, которая работает примерно в три раза дольше обычного теста на псевдопростоту. Однако бесхитростная реализация этих тестов потребует несколько большее время. Так что для получения указанных значений 2 и 3 требуется проявить некоторую смекалку.

Пусть, как и раньше, целые числа $a$ и $b$ таковы, что $\Delta = a^2 - 4b$ не равно квадрату целого числа, а последовательности $(U_j)$ и $(V_j)$ определены равенствами (3.8). Сначала заметим, что достаточно рассматривать только последовательность $(V_j)$, так как, зная $V_m$ и $V_{m+1}$, мы можем немедленно найти $U_m$ из равенства

$$U_m = \Delta^{-1}(2V_{m+1} - aV_m). \tag{3.13}$$

Заметим, далее, что при больших значениях $m$ для вычисления $V_m$ по предшествующим элементам есть следующее простое правило: если $0 \leqslant j \leqslant k$, то

$$V_{j+k} = V_jV_k - b^jV_{k-j}. \tag{3.14}$$

Пусть теперь $b = 1$. Воспользуемся формулой (3.14) при $k = j$ и $k = j+1$:

$$V_{2j} = V_j^2 - 2, \quad V_{2j+1} = V_jV_{j+1} - a \quad (\text{при } b = 1). \tag{3.15}$$

Таким образом, если известны вычеты $V_j \pmod n$ и $V_{j+1} \pmod n$, то при помощи (3.15) мы можем найти или пару $V_{2j} \pmod n$, $V_{2j+1} \pmod n$ или пару $V_{2j+1} \pmod n$, $V_{2j+2} \pmod n$, каждый раз производя два умножения по модулю $n$ и одно сложение по модулю $n$. Начиная с $V_0, V_1$ и последовательно применяя равенства (3.15), можно добраться до любой пары $V_m, V_{m+1}$. Пусть, например, $m$ равно 97. Путь от пары 0, 1 к паре 97, 98 выглядит так:

$$0,1 \to 1,2 \to 3,4 \to 6,7 \to 12,13 \to 24,25 \to 48,49 \to 97,98.$$

Существуют два типа ходов. На первом происходит переход от пары $a, a+1$ к паре $2a, 2a+1$, а на втором — к паре $2a+1, 2a+2$. Простой способ нахождения этой цепочки ходов состоит в обратном движении от заданной пары $m, m+1$. Также нетрудно перевести число $m$ в двоичную систему счисления и двигаться по полученной строке слева направо. При этом нуль будет означать первый тип хода, а единица — второй. Например, двоичная запись числа 97 выглядит так: 1100001. Поэтому мы должны, взяв пару 0,1, сделать два хода второго типа, затем четыре первого и, наконец, один ход второго типа.

Эта цепь называется двоичной цепью Люка. Более подробно эта тема освящена в работах [Montgomery 1992b] и [Bleichenbacher 1996]. Суммируем приведенные выше рассуждения на языке псевдокода.

**Алгоритм 3.6.7** (цепь Люка). **Для последовательности $x_0, x_1, \ldots$ с заданным правилом вычисления $x_{2j}$ через $x_j$ и $x_{2j+1}$ через пару $x_j, x_{j+1}$ этот алгоритм вычисляет пару $(x_n, x_{n+1})$ для данного натурального числа $n$. Двоичная запись числа $n$ имеет вид $(n_{B-1}, \ldots, n_1, n_0)$, где $n_{B-1}$ соответствует старшему разряду. Пусть правила имеют вид $x_{2j} = x_j * x_j$ и $x_{2j+1} = x_j \circ x_{j+1}$. На каждом шаге цикла for() алгоритм находит значения $u = x_j, v = x_{j+1}$ для некоторого неотрицательного числа $j$.**

1. [Инициализация]
   $(u, v) = (x_0, x_1)$;
2. [Цикл]
   for($B > j \geqslant 0$) {
   if($n_j == 1$) $(u, v) = (u \circ v, v * v)$;
   else $(u, v) = (u * u, u \circ v)$;
   }
   return $(u, v)$; // Возвращаем пару $(x_n, x_{n+1})$.

Вернемся к общему случаю многочлена $x^2 - ax + b$ и посмотрим, можно ли ослабить условие $b = 1$. Если $a = cd$, $b = d^2$, то можно воспользоваться равенством

$$V_m(cd, d^2) = d^m V_m(c, 1)$$

и быстро вернуться к случаю $b = 1$. Вообще, если $b$ равно квадрату целого числа, скажем, $b = d^2$, причем НОД$(n, b) = 1$, то

$$V_m(a, d^2) \equiv d^m V_m(ad^{-1}, 1) \pmod n,$$

где $d^{-1}$ обозначает вычет, обратный к $d$ по умножению по модулю $n$. Значит, и здесь вы возвращаемся к случаю $b = 1$. Перейдем к самому общему случаю,

когда число $b$ произвольно. Заметим, что если пробежать последовательность $V_m$ дважды, то мы получим новую последовательность $V_j$, причем будет выполнено равенство

$$V_{2m}(a,b) = V_m(a^2 - 2b, b^2),$$

т. е. число «$b$» для второй последовательности есть полный квадрат! Таким образом, если НОД$(n,b) = 1$ и целое число $A$ таково, что $A \equiv b^{-1}V_2(a,b) \equiv a^2b^{-1} - 2 \pmod n$, то

$$V_{2m}(a,b) \equiv b^m V_m(A,1) \pmod n. \tag{3.16}$$

Аналогичным образом,

$$U_{2m}(a,b) \equiv ab^{m-1}U_m(A,1) \pmod n,$$

поэтому применение равенства (3.13) при $a = A$ и $b = 1$ (так что $\Delta = A^2 - 4$) дает сравнение

$$U_{2m}(a,b) \equiv (a\Delta)^{-1}b^{m+1}\bigl(2V_{m+1}(A,1) - AV_m(A,1)\bigr) \pmod n. \tag{3.17}$$

Для эффективного вычисления пары $V_m(A,1) \pmod n$, $V_{m+1}(A,1) \pmod n$, где число $n$ взаимно просто с $b$ и мы считаем, что $A$ — целое число по модулю $n$, можно применять описанный выше метод двоичных цепей Люка. Таким образом, сравнения (3.16) и (3.17) позволяют найти значения $V_{2m}(a,b)$, $U_{2m}(a,b) \pmod n$. Поэтому, выбирая $2m = n - \left(\frac{\Delta}{n}\right)$, мы можем решить вопрос о том, является ли $n$ псевдопростым числом Люка или псевдопростым числом Фробениуса по отношению к $x^2 - ax + b$.

Подведем итог сказанному следующей теоремой.

**Теорема 3.6.8.** *Если числа $a, b, \Delta, A$ определены выше и составное число $n$ взаимно просто с произведением $2ab\Delta$, то $n$ является псевдопростым числом Люка по отношению к $x^2 - ax + b$ тогда и только тогда, когда выполнено сравнение*

$$AV_{\frac12\left(n-\left(\frac{\Delta}{n}\right)\right)}(A,1) \equiv 2V_{\frac12\left(n-\left(\frac{\Delta}{n}\right)\right)+1}(A,1) \pmod n. \tag{3.18}$$

*Более того, $n$ является псевдопростым числом Фробениуса по отношению к $x^2 - ax + b$ тогда и только тогда, когда выполнены сравнения* (3.18) *и*

$$b^{(n-1)/2}V_{\frac12\left(n-\left(\frac{\Delta}{n}\right)\right)}(A,1) \equiv 2 \pmod n. \tag{3.19}$$

Как отмечалось выше, при $m = \frac12\left(n - \left(\frac{\Delta}{n}\right)\right)$ пара $V_m(A,1)$, $V_{m+1}(A,1)$ может быть найдена по модулю $n$ при помощи $2\lg n$ умножений по модулю $n$ и $\lg n$ сложений по модулю $n$. Половина умножений по модулю $n$ является возведениями в квадрат по модулю $n$. Тест Ферма также требует $\lg n$ возведений в квадрат по модулю $n$ и до $\lg n$ дополнительных умножений по модулю $n$, если использовать алгоритм 2.1.5 для двоичной схемы. Из сравнения (3.18) следует, что тест Люка требует не более, чем в два раза больше времени по сравнению с тестом Ферма. Для применения сравнения (3.19) мы должны также вычислить $b^{(n-1)/2} \pmod n$, а следовательно, время работы теста Фробениуса (для квадратного трехчлена) не более, чем в три раза превосходит это время для теста Ферма.

Как и тесты Ферма, тесты Люка и Фробениуса применяются для исследования чисел на простоту. Приведенный ниже псевдокод соответствует рассуждениям, изложенным в этом разделе.

**Алгоритм 3.6.9** (тест на вероятное простое число Люка). Нам заданы целые числа $n, a, b, \Delta$, причем $\Delta = a^2 - 4b$ не равно квадрату целого числа, $n > 1$ и НОД$(n, 2ab\Delta) = 1$. Этот алгоритм выдает «$n$ — вероятное простое число Люка с параметрами $a, b$», если число $n$ или простое, или псевдопростое Люка по отношению к $x^2 - ax + b$, и «$n$ — составное число» в противном случае.

1. [Вспомогательные параметры]
   $A = a^2 b^{-1} - 2 \bmod n$;
   $m = \left(n - \left(\frac{\Delta}{n}\right)\right)/2$;
2. [Двоичная цепь Люка]
   При помощи алгоритма 3.6.7 вычисляем последние два члена последовательности $(V_0, V_1, \ldots, V_m, V_{m+1})$ с начальными значениями $(V_0, V_1) = (2, A)$ и заданными соотношениями $V_{2j} = V_j^2 - 2 \bmod n$ и $V_{2j+1} = V_j V_{j+1} - A \bmod n$;
3. [Объявление результата]
   if$(AV_m \equiv 2V_{m+1} \pmod n)$ return «$n$ — вероятное простое число Люка с параметрами $a, b$»;
   return «$n$ — составное число»;

Алгоритм теста на вероятное простое число Фробениуса отличается лишь тем, что шаг [Объявление результата] заменяется на

3'. [Тест Люка]
   if$(AV_m \not\equiv 2V_{m+1})$ return «$n$ — составное число»;

и добавляется новый шаг

4. [Тест Фробениуса]
   $B = b^{(n-1)/2} \bmod n$;
   if$(BV_m \equiv 2 \pmod n)$ return «$n$ — вероятное простое число Фробениуса с параметрами $a, b$»;
   return «$n$ — составное число»;

### 3.6.4. Теоретические соображения и более сильные тесты

Если многочлен $x^2 - ax + b$ неприводим над $\mathbf{Z}$ и не равен $x^2 \pm x + 1$, то псевдопростые числа Люка по отношению к $x^2 - ax + b$ редки по сравнению с простыми числами (см. упр. 3.26, где мы исключаем $x^2 \pm x + 1$). Этот результат содержится в статье [Baillie and Wagstaff 1980]. Наилучший результат в этом направлении описан в работе [Gordon and Pomerance 1991]. Поскольку каждое псевдопростое число Фробениуса по отношению к $x^2 - ax + b$ является одновременно и псевдопростым числом Люка по отношению к данному трехчлену, эти числа должны быть еще реже.

Было установлено, что для каждого неприводимого многочлена $x^2 - ax + b$ существует бесконечное множество псевдопростых чисел Люка, и, более того, бесконечное множество псевдопростых чисел Фробениуса. Для псевдопростых чисел Фибоначчи это было доказано в статье [Lehmer 1964], для общего случая псевдопростых чисел Люка — в статье [Erdős et al. 1988], а для псевдопростых чисел Фробениуса — в статье [Grantham 2001]. В доказательстве Грэнтхэма бесконечности множества псевдопростых чисел Фробениуса рассмотрен лишь случай $\left(\frac{\Delta}{n}\right) = 1$. Но существуют такие квадратные трехчлены, например, характеристический многочлен $x^2 - x - 1$ последовательности Фибоначчи, для которых существует бесконечно много псевдопростых чисел Фробениуса $n$, причем $\left(\frac{\Delta}{n}\right) = -1$ (см. [Parberry 1970] and [Rotkiewicz 1973]). Не так давно Роткевич доказал, что для любого трехчлена $x^2 - ax + b$, где $\Delta = a^2 - 4b$ не равно квадрату целого числа, существует бесконечно много псевдопростых чисело Люка $n$, для которых $\left(\frac{\Delta}{n}\right) = -1$.

По аналогии с сильно псевдопростыми числами (см. разд. 3.5) можно рассмотреть такие понятия, как сильно псевдопростое число Люка и сильно псевдопростое число Фробениуса. Пусть нечетное простое число $n$ не делит $b\Delta$. Если $\left(\frac{\Delta}{n}\right) = 1$, то в кольце $R = \mathbf{Z}_n[x]/(f(x))$ существует такой элемент $z$, что $z^2 = 1$ и $z \neq \pm 1$. Например, $f(x) = x^2 - x - 1$, $n = 11$, $z = 3 + 5x$. Однако, если $(x(a-x)^{-1})^{2m} = 1$, то несложные выкладки (см. упр. 3.30) дают равенство $(x(a-x)^{-1})^m = \pm 1$. Согласно (3.10) и (3.11), в кольце $R$ справедливо равенство $(x(a-x)^{-1})^{n-\left(\frac{\Delta}{n}\right)} = 1$. Таким образом, если записать $n - \left(\frac{\Delta}{n}\right) = 2^s t$, где число $t$ нечетно, то мы получим

$$\text{или } (x(a-x)^{-1})^t \equiv 1 \pmod{(f(x), n)},$$
$$\text{или } (x(a-x)^{-1})^{2^i t} \equiv -1 \pmod{(f(x), n)} \text{ для некоторого } i,\ 0 \leqslant i \leqslant s-1.$$

Это означает, что

$$\text{или } U_t \equiv 0 \pmod{n},$$
$$\text{или } V_{2^i t} \equiv 0 \pmod{n} \text{ для некоторого } i,\ 0 \leqslant i \leqslant s-1.$$

Если последнее утверждение имеет место для некоторого нечетного составного числа $n$, взаимно простого с $b\Delta$, мы будем называть $n$ сильно псевдопростым числом Люка по отношению к $x^2 - ax + b$. Нетрудно видеть, что каждое сильно псевдопростое число Люка по отношению к $x^2 - ax + b$ одновременно является и псевдопростым числом Люка по отношению к данному трехчлену.

В статье [Grantham 2001] описан тест для сильно псевдопростых чисел Фробениуса, причем не только для квадратных трехчленов, но и для всех многочленов. Мы рассмотрим лишь случай квадратного трехчлена и $\left(\frac{\Delta}{n}\right) = -1$. Пусть $n^2 - 1 = 2^S T$, где нечетное простое число $n$ не делит $b\Delta$, причем $\left(\frac{\Delta}{n}\right) = -1$. Согласно (3.10) и (3.11), мы получаем $x^{n^2-1} \equiv 1 \pmod{n}$, так что

$$\text{или } x^T \equiv 1 \pmod{n},$$
$$\text{или } x^{2^i T} \equiv -1 \pmod{n} \text{ для некоторого } i,\ 0 \leqslant i \leqslant S-1.$$

Если это выполнено для псевдопростого числа Фробениуса $n$ по отношению

к $x^2 - ax + b$, мы будем называть это число $n$ сильно псевдопростым числом Фробениуса по отношению к $x^2 - ax + b$. (Можно показать, что это условие не означает, что $n$ является псевдопростым числом Фробениуса, поэтому эти сравнения взяты за определение именно сильно псевдопростого числа Фробениуса.) В статье [Grantham 1998] показано, что сильно псевдопростое число Фробениуса $n$ по отношению к $x^2 - ax + b$, для которого $\left(\frac{\Delta}{n}\right) = -1$, одновременно является и сильно псевдопростым числом Люка по отношению к этому трехчлену.

Как и в случае обычного теста Люка, сильный тест Люка допускает реализацию за время, не превышающее удвоенное время работы обычного теста для псевдопростых чисел. В статье [Grantham 1998] показано, что сильный тест Фробениуса допускает реализацию за время, не превышающее утроенное время работы обычного теста для псевдопростых чисел. Интерес к сильно псевдопростым числам Фробениуса обусловлен следующим результатом [Grantham 1998].

**Теорема 3.6.10.** *Пусть составное число $n$ не является полным квадратом и не кратно ни одному простому числу, меньшему* 50000. *Тогда $n$ — сильно псевдопростое число Фробениуса по отношению к не более чем* $1/7710$ *части всех трехчленов вида $x^2 - ax + b$, где целые коэффициенты $a, b$ пробегают отрезок $[1, n]$ с условиями $\left(\frac{a^2-4b}{n}\right) = -1$ и $\left(\frac{b}{n}\right) = 1$.*

Этот результат следует сопоставить с теоремой Монье—Рабина 3.5.4. Согласно теореме 3.5.4, вероятность не распознать составное число за три независимых теста для сильно псевдопростых чисел не превосходит $1/64$. В свою очередь, теорема 3.6.10 утверждает существование такого теста, однократное применение которого имеет вероятность ошибки, не превосходящую $1/7710$. Здесь нельзя не упомянуть тест, описанный в работе [Zhang 2002], который сочетает в себе тест Люка и тест на сильно вероятное простое число, превосходя в большинстве случаев квадратичный тест Фробениуса.

## 3.6.5. Общий тест Фробениуса

В нескольких предыдущих разделах мы рассмотрели тест Грэнтхэма—Фробениуса для квадратных трехчленов. Здесь мы кратко опишем подход к обобщению этого метода на случай произвольного нормированного многочлена из $\mathbf{Z}[x]$.

Итак, пусть $f(x)$ — нормированный многочлен степени $d \geqslant 1$ из $\mathbf{Z}[x]$. Мы не будем предполагать, что этот многочлен непременно неприводим. Пусть нечетное простое число $p$ не делит дискриминант $\operatorname{disc}(f)$ многочлена $f(x)$. (Дискриминант нормированного многочлена $f(x)$ степени $d$ равен $(-1)^{d(d-1)/2}$, умноженному на результант многочлена $f(x)$ и его производной. Результант является определителем $(2d-1) \times (2d-1)$-матрицы, у которой на месте $i, j$ записан коэффициент при $x^{j-i}$ многочлена $f(x)$, если $i = 1, \ldots, d-1$, или коэффициент при $x^{j-(i-d+1)}$ многочлена $f'(x)$, если $i = d, \ldots, 2d-1$, причем, если некоторая степень $x$ отсутствует в записи многочлена, то на соответствующее место мат-

рицы записывается 0.) Как известно, $\operatorname{disc}(f) \neq 0$ тогда и только тогда, когда многочлен $f(x)$ не имеет кратных неприводимых делителей положительной степени. Поэтому из предположения о том, что $p$ не делит $\operatorname{disc}(f)$, автоматически следует отсутствие у многочлена $f$ кратных делителей.

Приводя коэффициенты по модулю $p$, можно рассматривать многочлен $f(x)$ в кольце $\mathbf{F}_p[x]$. Во избежание путаницы обозначим новый многочлен через $\overline{f}(x)$. Пусть многочлены $F_1(x), F_2(x), \ldots, F_d(x)$ кольца $\mathbf{F}_p[x]$ определены равенствами

$$\begin{aligned} F_1(x) &= \text{НОД}\left(x^p - x, \overline{f}(x)\right), \\ F_2(x) &= \text{НОД}\left(x^{p^2} - x, \overline{f}(x)/F_1(x)\right), \end{aligned}$$

$$F_d(x) = \text{НОД}\left(x^{p^d} - x, \overline{f}(x)/(F_1(x)\ldots F_{d-1}(x))\right).$$

Тогда справедливы следующие утверждения:

(1) $i$ делит $\deg(F_i(x))$ при $i = 1, \ldots, d$,
(2) $F_i(x)$ делит $F_i(x^p)$ при $i = 1, \ldots, d$,
(3) если

$$S = \sum_{i \text{ четно}} \frac{1}{i} \deg(F_i(x)),$$

то

$$(-1)^S = \left(\frac{\operatorname{disc}(f)}{p}\right).$$

Утверждение (1) следует из того, что каждый многочлен $F_i(x)$ является произведением неприводимых делителей степени $i$ многочлена $\overline{f}(x)$, поэтому его степень кратна $i$. Утверждение (2) справедливо для всех многочленов кольца $\mathbf{F}_p[x]$. Утверждение (3) не столь очевидно. Для доказательства рассмотрим группу Галуа многочлена $\overline{f}(x)$ над $\mathbf{F}_p$. Автоморфизм Фробениуса (который переводит каждый элемент поля разложения многочлена $\overline{f}(x)$ в его $p$-ю степень), разумеется, переставляет в поле разложения корни многочлена $\overline{f}(x)$. При этом происходит циклическая перестановка корней каждого неприводимого делителя, а следовательно, знак всей перестановки равен $-1$ в степени, равной количеству неприводимых делителей четной степени. Таким образом, знак автоморфизма Фробениуса равен в точности $(-1)^S$. Однако согласно теории Галуа, группа Галуа многочлена с попарно различными корнями состоит исключительно из четных перестановок корней тогда и только тогда, когда дискриминант многочлена является квадратом. Следовательно, знак автоморфизма Фробениуса совпадает с символом Лежандра $\left(\frac{\operatorname{disc}(f)}{p}\right)$. Тем самым третье утверждение доказано.

Идея Грэнтхэма заключается в том, что три перечисленные утверждения можно проверить численно, причем это делается очень легко даже в том случае, когда мы не уверены в простоте числа $p$. Если хотя бы одно из этих утверждений не выполняется, то число $p$ непременно составное. Это и является ос-

новой общего теста Фробениуса. Можно сказать, что $n$ — псевдопростое число Фробениуса по отношению к многочлену $f(x)$, если число $n$ составное, но, тем не менее, тест это не выявляет.

Более подробно по этому вопросу см. статьи [Grantham 1998, 2001].

## 3.7. Подсчет простых чисел

Теорема о простых числах 1.1.4 устанавливает асимптотическую формулу для функции $\pi(x)$, равной количеству простых чисел, не превосходящих $x$. Интересно сравнить этот результат с реальными значениями функции $\pi(x)$, как мы сделали в разд. 1.1.5. Разумеется, равенство

$$\pi\left(10^{21}\right) = 21127269486018731928$$

не было установлено путем непосредственного перебора на компьютере всех простых чисел вплоть до $10^{21}$. Таких простых чисел слишком много. Так каким же образом этот результат был получен? В следующих разделах мы предложим два различных подхода к любопытной проблеме подсчета простых чисел: комбинаторный и аналитический.

### 3.7.1. Комбинаторный метод

Здесь мы опишем элегантный комбинаторный метод, развитый Лагариасом, Миллером и Одлыжко на основе работ Майсселя и Лемера (см. [Lagarias et al. 1985], [Deléglise and Rivat 1996]). Он позволяет вычислять значение $\pi(x)$ с битовой сложностью $O\left(x^{2/3+\epsilon}\right)$, используя $O\left(x^{1/3+\epsilon}\right)$ битов памяти.

Занумеруем все простые числа: $p_1, p_2, p_3, \ldots$, где $p_1 = 2$, $p_2 = 3$, $p_3 = 5$ и т. д. Положим

$$\phi(x, y) = \#\{1 \leqslant n \leqslant x \ : \text{ каждый простой делитель числа } n \text{ превосходит } y\}.$$

Таким образом, значение $\phi(x, p_a)$ равно количеству целых чисел, не вычеркнутых из отрезка $[1, x]$ при применении решета Эратосфена и просеивания через числа $p_1, p_2, \ldots, p_a$. Поскольку в результате просеивания через простые числа $\leqslant \sqrt{x}$ остается лишь 1 и простые числа промежутка $(\sqrt{x}, x]$, мы получаем равенство

$$\pi(x) - \pi\left(\sqrt{x}\right) + 1 = \phi\left(x, \sqrt{x}\right).$$

Это соотношение можно легко применить для вычисления значений $\pi(x)$. При этом потребуется $O(x \ln\ln x)$ операций и, если разбить решето на сегменты, $O\left(x^{1/2} \ln x\right)$ битов памяти. (Для простоты мы будем далее заменять выражения вида $\ln x$ и $\ln\ln x$ на превосходящее их выражение $x^{\epsilon}$. При этом будет ясно, что при помощи определенных усилий можно заменить каждое выражение $x^{\epsilon}$ на некоторую небольшую степень логарифма или даже повторного логарифма.)

Заметим теперь, что решето не только определяет количество простых чисел, но и позволяет найти каждое из них. Поэтому если нас интересует лишь их количество, то мы можем попытаться ускорить этот процесс.

Разобьем множество подсчитываемых функцией $\phi(x,y)$ чисел согласно количеству их простых делителей с учетом кратности. Положим

$$\phi_k(x,y) = \#\{n \leqslant x \qquad n \text{ имеет ровно } k \text{ простых делителей,} \\ \text{причем все они больше } y\}.$$

Таким образом, если $x \geqslant 1$, то $\phi_0(x,y) = 1$, значение $\phi_1(x,y)$ равно количеству простых чисел промежутка $(y,x]$, значение $\phi_2(x,y)$ равно количеству чисел $pq \leqslant x$, где простые числа $p, q$ таковы, что $y < p \leqslant q$ и т. д. Очевидно равенство

$$\phi(x,y) = \phi_0(x,y) + \phi_1(x,y) + \phi_2(x,y) + \ldots.$$

Далее, заметим, что $\phi_k(x,y) = 0$ при $y^k \geqslant x$. Следовательно,

$$\phi\left(x, x^{1/3}\right) = 1 + \pi(x) - \pi\left(x^{1/3}\right) + \phi_2\left(x, x^{1/3}\right). \tag{3.20}$$

Поэтому мы можем найти значение $\pi(x)$, если вычислим $\phi\left(x, x^{1/3}\right)$, $\phi_2\left(x, x^{1/3}\right)$ и $\pi\left(x^{1/3}\right)$.

Разумеется, вычислить $\pi\left(x^{1/3}\right)$ можно посредством одного лишь решета Эратосфена. Рассмотрим теперь способ вычисления следующего слагаемого в равенстве (3.20). Значение $\phi_2\left(x, x^{1/3}\right)$ можно найти при помощи соотношения

$$\phi_2(x, x^{1/3}) = \binom{\pi(x^{1/3})}{2} - \binom{\pi(x^{1/2})}{2} + \sum_{x^{1/3} < p \leqslant x^{1/2}} \pi(x/p), \tag{3.21}$$

где символ $p$ пробегает простые числа. Для доказательства равенства (3.21) заметим сначала, что $\phi_2\left(x, x^{1/3}\right)$ равно количеству пар простых чисел $p, q$, для которых $x^{1/3} < p \leqslant q$ и $pq \leqslant x$. Тогда $p \leqslant x^{1/2}$. Для каждого фиксированного $p$ простое число $q$ пробегает отрезок $[p, x/p]$, а следовательно, число вариантов выбора $q$ равно $\pi(x/p) - \pi(p) + 1$. Таким образом,

$$\begin{aligned} \phi_2(x, x^{1/3}) &= \sum_{x^{1/3} < p \leqslant x^{1/2}} (\pi(x/p) - \pi(p) + 1) \\ &= \sum_{x^{1/3} < p \leqslant x^{1/2}} \pi(x/p) - \sum_{x^{1/3} < p \leqslant x^{1/2}} (\pi(p) - 1). \end{aligned}$$

Последняя сумма равна

$$\begin{aligned} \sum_{\pi(x^{1/3}) < j \leqslant \pi(x^{1/2})} (j-1) &= \sum_{j=1}^{\pi(x^{1/2})} (j-1) - \sum_{j=1}^{\pi(x^{1/3})} (j-1) \\ &= \binom{\pi(x^{1/2})}{2} - \binom{\pi(x^{1/3})}{2}. \end{aligned}$$

Доказательство соотношения (3.21) завершено.

Для того чтобы найти значение $\phi_2\left(x, x^{1/3}\right)$ по формуле (3.21), мы должны вычислить $\pi\left(x^{1/3}\right)$, $\pi\left(x^{1/2}\right)$ и сумму $\pi(x/p)$. Значение $\pi\left(x^{1/3}\right)$ мы уже нашли. Для вычисления $\pi\left(x^{1/2}\right)$ можно снова воспользоваться обычным решетом Эратосфена, но при этом разбить решето на блоки размера примерно

$x^{1/3}$, чтобы уложиться в заявленную оценку объема памяти, требуемой алгоритмом. Заметим, что в формуле (3.21) сумма значений $\pi(x/p)$ вычисляется для $x/p < x^{2/3}$. Поэтому обычное решето Эратосфена позволяет также вычислить сумму $\pi(x/p)$ за совокупное время $O\left(x^{2/3+\epsilon}\right)$, при этом требуется $O\left(x^{1/3+\epsilon}\right)$ битов памяти. В самом деле, пусть $N \approx x^{1/3}$ — удобное число для разбиения решета, т. е. мы будем рассматривать интервалы длины $N$, начиная с $x^{1/2}$. Предположим, что мы уже вычислили $\pi(z)$, и воспользуемся решетом (мы храним простые числа $< x^{1/3}$) на промежутке $[z, z+N)$ для вычисления различных значений $\pi(x/p)$ при $x/p$, принадлежащем этому промежутку, находя также и значение $\pi(z+N)$, которое будет использовано для вычислений на следующем промежутке. Значения $\pi(x/p)$ вычисляются и последовательно складываются, при этом ни одно из них по отдельности не хранится. Для нахождения простых чисел $p$, для которых $x/p$ принадлежит рассматриваемому промежутку, мы должны применить второе решето к промежутку $(x/(z+N), x/z]$, содержащемуся в $\left(x^{1/3}, x^{1/2}\right]$. Длина последнего промежутка не превосходит $N$, поэтому здесь не требуется большой объем памяти, и решето может быть реализовано при помощи сохраненного списка простых чисел, не превосходящих $x^{1/4}$, за время, равное $O\left(x^{1/3+\epsilon}\right)$. При больших значениях $z$ промежутки $(x/(z+N), x/z]$ становятся очень короткими, поэтому время можно немного сэкономить (не изменяя совокупную сложность) путем просеивания промежутка длины $N$ в этом диапазоне, сохраняя результаты и используя их для нескольких различных промежутков в более высоком диапазоне.

Для вычисления $\pi(x)$ по формуле (3.20) нам осталось найти $\phi\left(x, x^{1/3}\right)$. На первый взгляд кажется, что это потребует около $x$ шагов, поскольку потребуется подсчитать количество невычеркнутых чисел отрезка $[1, x]$ после применения решета Эратосфена с просеивающим множеством простых чисел $\leqslant x^{1/3}$. Однако мы сведем задачу вычисления $\phi\left(x, x^{1/3}\right)$ к большому числу меньших задач. Начнем с рекуррентного соотношения

$$\phi(y, p_b) = \phi(y, p_{b-1}) - \phi(y/p_b, p_{b-1}) \tag{3.22}$$

при $b \geqslant 2$. Доказать это равенство не трудно и мы предоставляем это читателю (см. упр. 3.33). Поскольку $\phi(y, 2) = \lfloor (y+1)/2 \rfloor$, мы можем последовательно применять соотношение (3.22), в конечном счете приходя к выражениям вида $\phi(y, 2)$ для некоторых значений $y$. Например,

$$\begin{aligned}
\phi(1000, 7) &= \phi(1000, 5) - \phi(142, 5) \\
&= \phi(1000, 3) - \phi(200, 3) - \phi(142, 3) + \phi(28, 3) \\
&= \phi(1000, 2) - \phi(333, 2) - \phi(200, 2) + \phi(66, 2) \\
&\quad - \phi(142, 2) + \phi(47, 2) + \phi(28, 2) - \phi(9, 2) \\
&= 500 - 167 - 100 + 33 - 71 + 24 + 14 - 5 \\
&= 228.
\end{aligned}$$

Следуя этой схеме, можно выразить каждое $\phi(x, p_a)$ через сумму $2^{a-1}$ слагаемых. На самом деле, полученный результат является просто применением формулы включений—исключений к делителям произведения $p_2 p_3 \ldots p_a$ пер-

вых $a-1$ нечетных простых чисел: справедливо равенство

$$\phi(x,p_a)=\sum_{n\mid p_2p_3\quad p_a}\mu(n)\phi(x/n,2)=\sum_{n\mid p_2p_3\quad p_a}\mu(n)\left\lfloor\frac{x/n+1}{2}\right\rfloor,$$

где $\mu$ обозначает функцию Мёбиуса (см. разд. 1.4.1).

Ясно, что при $a=\pi(x^{1/3})$ количество членов $2^{a-1}$ слишком велико, и более удачным было бы просто просеивание до $x$. Однако нам не нужно рассматривать слагаемые $n>x$, поскольку для них $\phi(x/n,2)=0$. Это «правило отбрасывания» снижает число слагаемых до $O(x)$, так что здесь у обычного просеивания уже нет такого превосходства. Развивая эту идею далее, можно уменьшить постоянную в знаке $O$ до приемлемо малого значения. Поскольку $2\cdot3\cdot5\cdot7\cdot11=2310$, вычислив таблицу значений функции $\phi(x,11)$ при $x=0,1,\ldots,2309$, можно быстро найти любое значение $\phi(x,11)$: оно равно $\varphi(2310)\lfloor x/2310\rfloor+\phi(x \bmod 2310,11)$, где $\varphi$ — функция Эйлера. Заканчивая применять соотношение (3.22) каждый раз, когда значение $b$ станет равным 11 или значение $y/p_b$ окажется меньшим 1, получим

$$\phi(x,p_a)=\sum_{\substack{n\mid p_6p_7\quad p_a\\ n\leqslant x}}\mu(n)\phi(x/n,11).$$

Если $a=\pi\left(x^{1/3}\right)$, то число слагаемых этой суммы асимптотически эквивалентно $cx$, где $c=\rho(3)\zeta(2)^{-1}\prod_{i=1}^{5}p_i/(p_i+1)$, где $\rho$ — функция Дикмана (см. разд. 1.4.5), а $\zeta$ — дзета-функция Римана (как известно, $\zeta(2)=\pi^2/6$). Такой вид постоянной $c$ связан с тем, что число $n$ свободно от квадратов, не имеет простых делителей, превосходящих $x^{1/3}$ или меньших 12. Поскольку $\rho(3)\approx0.0486$, мы получаем $c\approx0.00987$. Снижая значение $a$ до $\pi\left(x^{1/4}\right)$ (при этом помимо $\phi_2\left(x,x^{1/4}\right)$ нам придется вычислить и $\phi_3\left(x,x^{1/4}\right)$), мы одновременно уменьшим и постоянную $c$, поскольку вместо $\rho(3)$ будет множитель $\rho(4)\approx0.00491$, а значит, $c\approx0.000998$. По существу, эти рассуждения и составляют метод Майсселя, впоследствии усовершенствованный Лемером (см. [Lagarias et al. 1985]).

Однако нашей текущей задачей остается снижение битовой сложности до $O\left(x^{2/3+\epsilon}\right)$. Для этого мы воспользуемся еще одним правилом отбрасывания. А именно, мы прекратим применять соотношение (3.22), если дойдем до $\phi(y,p_b)$, где

(1) $p_b=2$ и $y\geqslant x^{2/3}$ или
(2) $y<x^{2/3}$.

Здесь $y$ соответствует некоторому числу $x/n$, где $n\mid p_2p_3\ldots p_a$. Очевидно, что количество слагаемых первого типа не превосходит $x^{1/3}$, поскольку каждое из них соответствует некоторому значению $n<x^{1/3}$. Для оценки числа слагаемых второго типа заметим, что «родителем» члена $\phi(x/n,p_b)$ при нашем построении является или член $\phi(x/n,p_{b+1})$, или член $\phi(x/(n/p_{b+1}),p_{b+1})$. Последнее имеет место лишь тогда, когда $p_{b+1}$ является наименьшим простым делителем числа $n$, причем $n/p_{b+1}\leqslant x^{1/3}$, а первое не реализуется никогда, поскольку в

этом случае уже должно было произойти отбрасывание второго типа. Таким образом, число членов второго типа не превосходит количества пар $m, p_b$, где $m \leqslant x^{1/3}$ и $p_b$ меньше наименьшего простого делителя числа $m$. Число этих пар не превосходит $x^{1/3}\pi(x^{1/3})$, так что количество слагаемых второго типа меньше $x^{2/3}$.

Для целого числа $m > 1$ положим

$$P_{\min}(m) = \text{ наименьший простой делитель числа } m.$$

Тогда в силу нового правила отбрасывания получим

$$\begin{aligned}\phi(x, p_a) \;=\; & \sum_{\substack{m \mid p_2 p_3 \ldots p_a \\ m \leqslant x^{1/3}}} \mu(m) \left\lfloor \frac{x/m+1}{2} \right\rfloor \\ & - \sum_{\substack{m \mid p_2 p_3 \ldots p_a \\ 1 < m \leqslant x^{1/3}}} \mu(m) \sum_{\substack{p_{b+1} < P_{\min}(m) \\ p_{b+1} m > x^{1/3}}} \phi\left(\frac{x}{m p_{b+1}}, p_b\right).\end{aligned} \tag{3.23}$$

Воспользуемся этим равенством при $a = \pi(x^{1/3})$. Нетрудно вычислить первую сумму в (3.23), соответствующую членам первого типа. При помощи решета подготовим таблицу $\mathcal{T}$ нечетных бесквадратных чисел $m \leqslant x^{1/3}$ вместе с их наименьшими простыми делителями (они потребуются в двойной сумме) и значением $\mu(m)$. (Каждая позиция решета соответствует некоторому нечетному числу, не превосходящему $x^{1/3}$, и в начале содержит число 1. При первом попадании в эту позицию мы записываем текущее простое число как наименьший простой делитель числа, соответствующего этой позиции решета. При каждом последующем попадании в эту позицию мы умножаем содержимое на $-1$. Перебрав все простые числа до $x^{1/3}$, мы приписываем каждой из оставшихся неотмеченных позиций соответствующее ей число и изменяем ее содержимое на $-1$. Наконец, мы просеиваем наше множество через квадраты простых чисел $p^2$, где $p \leqslant x^{1/6}$, причем при попадании в каждую позицию ее содержимое заменяем на 0. Итак, числа с ненулевыми значениями содержимого позиций свободны от квадратов, причем это значение содержимого равно значению $\mu$ от соответствующего числа, а записанное число является наименьшим простым делителем.) Создание таблицы $\mathcal{T}$ требует $O(x^{1/3+\varepsilon})$ операций и битов памяти, и на ее основе мы можем вычислить первую сумму в (3.23) за время $O(x^{1/3+\varepsilon})$.

Главная часть рассуждения состоит в вычислении двойной суммы равенства (3.23). Сначала опишем, как можно это сделать за время $O\left(x^{2/3+\varepsilon}\right)$ с использованием такого же числа битов памяти, а затем покажем, как разбиение на сегменты может снизить последнее значение для памяти до $O(x^{1/3+\varepsilon})$. Подготовим таблицу $\mathcal{T}'$ троек $\mu(m), \lfloor x/(mp_{b+1})\rfloor, b$, где число $m$ пробегает все элементы заранее созданной таблицы $\mathcal{T}$, большие 1, а $b$ пробегает все такие числа, что $p_{b+1} < P_{\min}(m)$ и $mp_{b+1} > x^{1/3}$. Отметим, что все числа $\lfloor x/(mp_{b+1})\rfloor$ меньше $x^{2/3}$. Просеем отрезок $\left[1, x^{2/3}\right]$ через все простые числа, не превосходящие $x^{1/3}$. На шаге $b$ мы просеяли через $p_1, p_2, \ldots, p_b$, а значит, можем получить значение $\phi(y, b)$ для всех $y \leqslant x^{2/3}$. Нас интересуют значения $y = \lfloor x/(mp_{b+1})\rfloor$.

Однако одно лишь знание чисел, взаимно простых с произведением $p_1p_2\dots p_b$, не достаточно для знания их количества до $y$, поэтому требуются дополнительные вычисления. Проведение такой работы для каждого $b$ увеличило бы битовую сложность до $O\left(x^{1+\epsilon}\right)$. Эта проблема решается посредством организации двоичной структуры данных. Для $i=0,1,\dots,\lfloor \lg n\rfloor$ рассмотрим промежутки

$$I_{i,j}=\left((j-1)2^i, j2^i\right],$$

где $j$ — натуральное число и $I_{i,j}\subset\left[1,x^{2/3}\right]$. Общее число этих промежутков есть $O\left(x^{2/3}\right)$. Для каждого промежутка $I_{i,j}$ положим

$$A(i,j,b)=\#\{n\in I_{i,j}:\text{НОД}\,(n,p_1p_2\dots p_b)=1\}.$$

Идея состоит в том, чтобы найти все числа $A(i,j,b)$ для фиксированного значения $b$. После этого можно воспользоваться двоичной записью числа $\lfloor x/(mp_{b+1})\rfloor$, сложить все соответствующие значения $A(i,j,b)$ и получить $\phi(\lfloor x/(mp_{b+1})\rfloor, p_b)$.

Итак, покажем, каким образом можно найти числа $A(i,j,b)$, зная предыдущие значения $A(i,j,b-1)$ (начальные значения $A(i,j,0)$ мы полагаем равными $2^i$). Заметим, что при $i=0$ промежуток $I_{0,j}$ содержит лишь целое число $j$, так что $A(0,j,b)$ равно 1, если $j$ взаимно просто с произведением $p_1p_2\dots p_b$, и 0 в противном случае. Для целых чисел $l\leqslant x/p_b$ мы обновляем числа $A(i,j,b)$, соответствующие промежуткам $I_{i,j}$, которые содержат $lp_b$. Число таких промежутков для данного $lp_b$ есть $O(\ln x)$. Если $A(0,j,b-1)=0$, где $j=lp_b$, тогда обновлений не требуется ни для одного интервала. Если $A(0,j,b-1)=1$, где снова $j=lp_b$, мы полагаем каждое соответствующее $A(i,j,b)$ равным $A(i,j,b-1)-1$. Общее число обновлений равно $O\left(x^{2/3}(\ln x)/p_b\right)$, поэтому суммируя по $p_b\leqslant x^{1/3}$, получим оценку $O\left(x^{2/3+\epsilon}\right)$.

Проведенное рассуждение требует $O(x^{2/3+\epsilon})$ битов памяти. Чтобы уменьшить это значение до $O(x^{1/3+\epsilon})$, выберем такое целое число $k$, что $x^{1/3}\leqslant 2^k<2x^{1/3}$, и разобьем отрезок $\left[1,x^{2/3}\right]$ на блоки длины $2^k$, где или последний блок немного короче, или мы немного зашли за $x^{2/3}$. Блок с номером $r$ имеет вид $\left((r-1)2^k, r2^k\right]$, т. е. это промежуток $I_{r,k}$. Когда мы его достигнем, у нас будут храниться значения $\phi\left((r-1)2^k,p_b\right)$ для всех $b\leqslant\pi\left(x^{1/3}\right)$ из предшествующих блоков. Затем мы воспользуемся заранее созданной таблицей $\mathcal{T}$ и найдем тройки $\mu(m),\lfloor x/(mp_{b+1})\rfloor,\ b$, где $\lfloor x/(mp_{b+1})\rfloor$ лежит в $r$-м блоке. При $i\leqslant k$ промежутки $I_{i,j}$ непрерывно заполняют $r$-й блок, и нам не требуется рассматривать бóльшие значения $i$. Дальнейший процесс происходит, как раньше, и мы вычисляем каждое искомое значение $\phi(x/(mp_{b+1}),p_b)$, где $\lfloor x/(mp_{b+1})\rfloor$ лежит в $r$-м блоке, а также находим значения $\phi(r2^k,p_b)$ для каждого $b$, чтобы воспользоваться ими для следующего блока. Вычисленные значения $\phi(x/(mp_{b+1}),p_b)$ не хранятся, а умножаются на $\mu(m)$ и прибавляются к текущей сумме, которая соответствует второму слагаемому правой части равенства (3.23). Время и память, требуемые на выполнение описанных процедур для всех $p_b\leqslant x^{1/3}$ и $r$-го блока, равны $O(x^{1/3+\epsilon})$. Значения $\phi\left(r2^k,p_b\right)$ записываются поверх предшествующих значений $\phi((r-1)2^k,p_b)$, поэтому совокупный объем требуемой памяти

есть $O\left(x^{1/3+\epsilon}\right)$. Общее число блоков не превосходит $x^{1/3}$, так что совокупное время на все вычисления равно $O\left(x^{2/3+\epsilon}\right)$, что и было заявлено.

На практике этот алгоритм можно ускорить разнообразными приемами (см. [Lagarias et al. 1985] и [Deléglise and Rivat 1996]).

## 3.7.2. Аналитический метод

Здесь мы опишем высокоэффективный метод подсчета простых чисел, предложенный в статье [Lagarias and Odlyzko 1987], и рассмотрим пути его усовершенствования. Основная идея состоит в использовании того факта, что дзета-функция Римана, в определенном смысле, содержит информацию о свойствах простых чисел. Начнем с формального логарифмирования произведения Эйлера (1.18):

$$\ln \zeta(s) = \ln \prod_{p\in\mathcal{P}} (1-p^{-s})^{-1} = -\sum_{p\in\mathcal{P}} \ln(1-p^{-s}),$$

так что возникает следующее представление $\ln\zeta(s)$ в виде повторного ряда:

$$\ln \zeta(s) = \sum_{p\in\mathcal{P}} \sum_{m=1}^{\infty} \frac{1}{m p^{sm}}, \tag{3.24}$$

причем все преобразования справедливы (и, в частности, если потребуется, то можно изменить порядок суммирования) в полуплоскости $\mathrm{Re}(s) > 1$, и мы считаем, что $\ln\zeta$ следует понимать как функцию непрерывно меняющегося аргумента. (По сложившейся на сегодня традиции, мы начинаем с положительного вещественного значения $\ln\zeta(2)$ и отслеживаем логарифм по аргументу $\zeta$, двигаясь вдоль контура, идущего вертикально до точки $2+i\,\mathrm{Im}(s)$ и далее до $s$.)

Для того чтобы применить равенство (3.24) при подсчете простых чисел, определим функцию, которая напоминает подсчитывающую простые числа функцию $\pi(x)$ (но с ней не совпадает). Рассмотрим сумму по *степеням* простых чисел, не превосходящим $x$, а именно,

$$\pi^*(x) = \sum_{p\in\mathcal{P},\ m>0} \frac{\theta(x-p^m)}{m}, \tag{3.25}$$

где функция Хевисайда $\theta$ определена равенством

$$\theta(z) = \begin{cases} 1, & \text{если } z>0; \\ 1/2, & \text{если } z=0; \\ 0, & \text{если } z<0. \end{cases}$$

Появление функции $\theta$ связано с тем, что мы хотим подсчитывать лишь те степени простых чисел $p^m$, которые не превосходят $x$, причем, если значение $x$ окажется в точности равным $p^m$, то соответствующее слагаемое должно быть равно $1/(2m)$. Следующий шаг заключается в применении формулы Перрона, согласно которой для неотрицательного действительного числа $x$, натурального числа $n$ и заданного контура $\mathcal{C} = \{s : \mathrm{Re}(s) = \sigma\}$, где $\sigma > 0$ фиксировано, а

$t = \operatorname{Im}(s)$ изменяется, справедливо равенство

$$\frac{1}{2\pi i}\int_{\mathcal{C}}\left(\frac{x}{n}\right)^s\frac{ds}{s} = \theta(x-n). \tag{3.26}$$

Из всего вышесказанного немедленно получаем, что для рассмотренного контура (во избежание особенности функции $\ln\zeta$ мы будем считать, что $\sigma > 1$) имеет место соотношение

$$\pi^*(x) = \frac{1}{2\pi i}\int_{\mathcal{C}} x^s \ln\zeta(s)\frac{ds}{s}. \tag{3.27}$$

Эта формула предоставляет аналитический способ вычисления $\pi(x)$, поскольку, если $x$ не равен степени простого числа, согласно соотношению (3.25) справедливо равенство

$$\pi^*(x) = \pi(x) + \frac{1}{2}\pi\left(x^{1/2}\right) + \frac{1}{3}\pi\left(x^{1/3}\right) + \ldots,$$

причем сумма обрывается, достигнув члена $\pi\left(x^{1/n}\right)/n$, для которого $2^n > x$.

Очевидно, что, по крайней мере теоретически, значение $\pi(x)$ может быть найдено через значение контурного интеграла (3.27) при помощи несложных дополнительных вычислений значений $\pi\left(x^{1/n}\right)$, начиная с $\pi\left(\sqrt{x}\right)$. Кроме того, можно просто последовательно применить контурное интегрирование, поскольку главный член разности $\pi^*(x) - \pi(x)$ равен $\pi^*\left(x^{1/2}\right)/2$ и т. д. Еще один способ восстановления функции $\pi$ по вычисленным значениям $\pi^*$ состоит в применении формулы обращения (снова, если $x$ не равен степени простого числа)

$$\pi(x) = \sum_{n=1}^{\infty}\frac{\mu(n)}{n}\pi^*\left(x^{1/n}\right).$$

Таким образом, этот аналитический подход сводится к численному интегрированию, но на этом пути возникает ряд трудностей. Начнем с того, что необходимо уметь вычислять значения $\zeta$ с достаточной точностью. Кроме того, необходима строгая оценка пределов интегрирования по контуру. Начнем с рассмотрения последней проблемы. Предположим, что у нас есть точный вычислительный метод для расчета значений самой функции $\zeta$. Возьмем $x = 100$, $\sigma = 3/2$. Численное интегрирование при конкретных значениях $T \in \{10, 30, 50, 70, 90\}$ дает

$$\begin{aligned}\pi^*(100) &\approx \operatorname{Re}\frac{100^{3/2}}{\pi}\int_0^T \frac{100^{it}}{3/2+it}\ln\zeta(3/2+it)\,dt\\ &\approx 30.14,\ 29.72,\ 27.89,\ 29.13,\ 28.3,\end{aligned}$$

из чего видна плохая сходимость значений контурного интеграла: истинное значение $\pi^*(100)$ можно непосредственно найти вручную, оно равно $428/15 = 28.5(3)$. Более того, если рассмотреть это значение как функцию от предела интегрирования $T$, можно заметить, что имеют место достаточно хаотичные колебания около истинного значения, причем получить точные оценки остаточного члена, как следовало ожидать, совсем не просто (см. упр. 3.37).

Предложения работы [Lagarias and Odlyzko 1987] позволяют, в принципе, устранить указанные недостатки аналитического подхода. Что касается вычисления значений самой дзета-функции, то для наибольшей скорости обычно применяется формула Римана—Зигеля. В частности, если мнимая часть $t$ переменной $s$ принимает громадные значения, эта формула становится единственным доступным средством (хотя не так давно были разработаны ее интересные варианты, о чем мы упомянули в конце упр. 1.61). Кроме того, существует предложенная [Odlyzko and Schönhage 1988] схема для, так сказать, «параллельных» вычислений значений $\zeta(s)$, скажем, для прогрессии мнимых частей переменной $s$. Именно этот тип одновременных вычислений и необходим для численного интегрирования. Среди современных сборников, содержащих варианты формулы Римана—Зигеля и другие вычислительные подходы, отметим статью [Borwein et al. 2000], а также упомянутые в ней работы. В работе [Crandall 1998] можно найти описание разнообразных быстрых алгоритмов для одновременных вычислений при различных множествах значений аргумента. Важный способ ускорения вычисления значений дзета-функции заключается в применении быстрого преобразования Фурье, а также методов вычисления значений многочлена и методов Ньютона для одновременного вычисления значений $\zeta(s)$ на данном множестве значений аргумента $s$. В настоящей книге мы приводим достаточно указаний (см. упр. 1.61) для того, чтобы можно было по меньшей мере начать с вычисления одного значения $\zeta(\sigma + it)$ при помощи не более чем $O\left(t^{1/2+\epsilon}\right)$ битовых операций.

Что касается проблемы, связанной с плохой сходимостью контурных интегралов, то остроумный прием заключается в рассмотрении гладкой (можно сказать, «адиабатической») функции отключения, которая улучшает сходимость (измененных) контурных интегралов. Это явление схоже с явлением быстрого убывания спектра гладких функций в анализе Фурье. Итак, нам потребуется тождество Лагариаса—Одлыжко (здесь и далее мы будем предполагать, что $x$ не равен степени простого числа):

$$\pi^*(x) = \frac{1}{2\pi i}\int_{\mathcal{C}} F(s,x)\ln\zeta(s)ds + \sum_{p\in\mathcal{P},\ m>0}\frac{\theta(x-p^m)-c(p^m,x)}{m}, \qquad (3.28)$$

где $c, F$ образуют пару преобразования Меллина:

$$c(u,x) = \frac{1}{2\pi i}\int_{\mathcal{C}} F(s,x)u^{-s}\,ds,$$

$$F(s,x) = \int_0^\infty c(u,x)u^{s-1}\,du.$$

Чтобы понять суть этого метода, возьмем функцию отключения $c(u,x)$ равной $\theta(x-u)$. Тогда $F(s,x) = x^s/s$, последняя сумма в (3.28) равна нулю, и мы получаем исходное аналитическое представление (3.27) для $\pi^*$. Однако давайте теперь рассмотрим класс непрерывных функций отключения $c(u,x)$, которые равны 1 при $u \in [0, x-y)$, монотонно убывают (к нулю) при $u \in (x-y, x]$ и равны нулю при $u > x$. Для повышения вычислительной эффективности мы

впоследствии выберем $y$ порядка $\sqrt{x}$. Далее, комбинируя выписанные выше равенства, мы получаем

$$\begin{aligned}\pi(x) &= \frac{1}{2\pi i}\int_C F(s,x)\ln\zeta(s)\,ds \\ &\quad - \sum_{p\in\mathcal{P},\ m>1}\frac{\theta(x-p^m)}{m} + \sum_{p\in\mathcal{P},\ m>0}\frac{\theta(x-p^m)-c(p^m,x)}{m}.\end{aligned} \tag{3.29}$$

Отметим, что вычисление последней суммы трудностей не вызывает, поскольку она содержит лишь $O(\sqrt{x})$ членов. Предпоследняя сумма, отражающая разность между $\pi(x)$ и $\pi^*(x)$, также состоит из $O(\sqrt{x})$ слагаемых.

Выберем конкретное гладкое угасание, а именно, при $u\in(x-y,x]$ положим

$$c(u,x) = 3\frac{(x-u)^2}{y^2} - 2\frac{(x-u)^3}{y^3}.$$

Заметим, что $c(x-y,x)=1$ и $c(x,x)=0$, что и требовалось для непрерывных функций $c$ из описанного класса. Преобразование Меллина функции $c$ имеет вид

$$\begin{aligned}&\frac{y^3}{6}F(s,x) = \\ &\frac{-2x^{s+3}+(s+3)x^{s+2}y+(x-y)^s(2x^3+(s-3)x^2y-2sxy^2+(s+1)y^3)}{s(s+1)(s+2)(s+3)}.\end{aligned} \tag{3.30}$$

Несмотря на некоторую громоздкость, это выражение позволяет более эффективно подсчитывать простые числа. В частности, знаменатель последней дроби есть $O(t^4)$, что обнадеживает. В качестве примера рассмотрим результаты численного интегрирования подобно тому, как это делалось выше. Выбирая $x=100$, $y=10$, для того же множества пределов интегрирования $T\in\{10,30,50,70,90\}$ мы получим

$$\pi(100)\approx 25.3,\ 26.1,\ 25.27,\ 24.9398,\ 24.9942.$$

Эти данные вполне удовлетворительны, поскольку $\pi(100)=25$ (отметим, однако, что при малых значениях $T$ по-прежнему наблюдается несколько хаотичное поведение). Необходимо отметить, что Лагариас и Одлыжко предложили гораздо более общую параметрическую форму пары Меллина $c, F$ и указали способы оптимизации параметров. При этом вычисление значения $\pi(x)$ их методом требует $O\left(x^{1/2+\epsilon}\right)$ битовых операций и $O\left(x^{1/4+\epsilon}\right)$ битов памяти, причем в случае ограниченности объема памяти можно заменить показатели на $3/5+\varepsilon$ и $\varepsilon$ соответственно.

На момент написания этой книги авторам не был известен ни один практический результат, полученный аналитическим методом, который мог бы соперничать с огромными достижениями описанных ранее комбинаторных методов. Однако, скорее всего, выход из сложившейся ситуации является лишь делом ближайшего времени. В частности, в частном разговоре [Galway 1998] было заявлено, что значения $\pi(10^n)$ при $n=13$, и, быть может, 14, можно вычислить

при помощи некоторой функции отключения $c$ и (лишь) стандартных арифметических операций с плавающей точкой двойной точности для численного интегрирования. Наверное, применение аналитического метода для получения современных результатов, скажем, при $x = 10^{21}$ или еще большем, потребует не менее чем 100-битовой точности. Эта точность зависит от того, насколько хороши оценки остатков для контурных интегралов. Функции Гэлвея представляют собой удачный выбор пары Меллина, и оказываются более эффективными, чем функции отключения, приводящие к функции $F$ вида (3.30). Положим

$$c(u,x) = \frac{1}{2}\operatorname{erfc}\left(\frac{\ln\frac{u}{x}}{2a(x)}\right),$$

где erfc — обычная функция ошибок:

$$\operatorname{erfc}(z) = \frac{2}{\sqrt{\pi}}\int_z^\infty e^{-t^2}\,dt\,,$$

а функция $a$ введена для повышения эффективности. Функция $c$ производит гладкое отключение при $u \sim x$, при этом его скорость зависит от выбора $a$. Двойственная по Меллину функция оказывается равной в точности

$$F(s,x) = \frac{x^s}{s}e^{s^2a(x)^2}. \tag{3.31}$$

При $s = \sigma + it$ замечательным (для вычислительных целей) угасанием в $F$ является $e^{-t^2a^2}$. При этом численные данные дают еще более хороший результат. Например, известно, что формула (3.29) с функцией угасания $a(x) = (2x)^{-1/2}$, $\sigma = 3/2$ и пределами интегрирования $T \in \{20, 40, 60, 80, 100, 120\}$, примененная для $x = 1000$, позволяет получить, соответственно, значения

$$\pi(1000) \approx 170.6,\ 169.5,\ 170.1,\ 167.75,\ 167.97,\ 167.998.$$

Это превосходно согласуется с истинным значением $\pi(1000) = 168$ и, более того, при этом хаотическая манера сходимости на качественном уровне проявляется не столь явно.

Кстати говоря, несмотря на то, что мы пользовались свойствами функции $\zeta(s)$ справа от критической полосы, существуют методы подсчета простых чисел, основанные на ее свойствах внутри полосы (см. упр. 3.50).

## 3.8. Упражнения

**3.1.** Возвращаясь к рассуждениям в начале этой главы, обозначим через $S_B(n)$ сумму цифр числа $n$ по основанию $B$. Для некоторых значений $B$ можно проследить интересные свойства. Пусть $B = 7$. Найдите наименьшее простое число $p$, для которого число $S_7(p)$ составное. (Величина этого простого числа должна Вас удивить!) Затем найдите все возможные составные значения $S_7(p)$ для простых чисел $p < 16000000$ (этих значений *очень* мало!). Возникают два естественных вопроса, ответы на которые авторам не известны. Пусть задано основание $B$. Существует ли бесконечно много простых чисел $p$, для которых

значение $S_B(p)$ является простым числом? А составным числом? Очевидно, что хотя бы на один из этих вопросов ответ должен быть положительным!

**3.2.** Иногда исследования в других областях знаний приводят к теоретико-числовым задачам. Рассмотрим удивительную находку из статьи [Golomb 1956], которая основана на хитроумной комбинаторике (и даже некоторых ярких «зрительных» образах) и позволяет доказать малую теорему Ферма 3.4.1.

Для данного числа $p$ требуется составить ожерелья из $p$ бусин. Каждое из ожерелий может состоять из бусин $n$ различных цветов, причем имеется ограничение: ни одно ожерелье не может быть одноцветным.

(1) Докажите, что существует в точности $n^p - n$ распрямленных (т. е. еще не свернутых) таких ожерелий.
(2) Докажите, что после сворачивания число различных ожерелий станет равным $(n^p - n)/p$.
(3) Докажите малую теорему Ферма: $n^p \equiv n \pmod p$.
(4) В каком месте Вы воспользовались простотой числа $p$?

**3.3.** Докажите, что если $n > 1$ и НОД $(a^n - a, n) = 1$ для некоторого натурального числа $a$, то число $n$ не только составное, но и не равно степени простого числа.

**3.4.** Для каждого числа $B \geqslant 2$ обозначим через $d_B$ асимптотическую плотность множества таких натуральных чисел, которые имеют простой делитель, превосходящий $B$ и состоящий исключительно из простых чисел, не превосходящих $B$. То есть, если $N(x, B)$ обозначает количество таких натуральных чисел, не превосходящих $x$, то $d_B = \lim_{x\to\infty} N(x, B)/x$.

(1) Покажите, что
$$d_B = 1 - \prod_{p\leqslant B}\left(1 - \frac{1}{p}\right)\cdot\sum_{m=1}^{B}\frac{1}{m},$$
где произведение берется по простым числам.
(2) Найдите наименьшее значение $B$, для которого $d_B > d_7$.
(3) При помощи теоремы Мертенса 1.4.2 покажите, что
$$\lim_{B\to\infty} d_B = 1 - e^{-\gamma} \approx 0.43854,$$
где $\gamma$ — постоянная Эйлера.
(4) В статье [Rosser and Schoenfeld 1962] показано, что если $x \geqslant 285$, то $e^\gamma \ln x \prod_{p\leqslant x}(1 - 1/p)$ заключено между $1 - 1/(2\ln^2 x)$ и $1 + 1/(2\ln^2 x)$. Воспользуйтесь этим результатом и докажите, что $0.25 \leqslant d_B < e^{-\gamma}$ при всех $B \geqslant 2$.

**3.5.** Пусть $c$ — положительное действительное число. Рассмотрим множество натуральных чисел $n$, наибольший простой делитель которых не превосходит $n^c$. Докажите, что если асимптотическая плотность этого множества равна $1/2$, то $c = 1/(2\sqrt{e})$. Любопытное междисциплинарное упоминание см. в статье [Knuth and Trabb Pardo 1976].

Теперь рассмотрим множество натуральных чисел $n$, для которых *второй* из наибольших простых делителей (если он существует) не превосходит $n^c$. Докажите, что если асимптотическая плотность этого множества равна $1/2$, то $c$ является решением уравнения

$$I(c) = \int_c^{1/2} \frac{\ln(1-u) - \ln u}{u}\, du = \frac{1}{2}.$$

Найдите отсюда численное значение $c$. Укажем интересный современный подход к этому вопросу. Сначала покажите, что для этого интеграла справедливо точное равенство

$$I(c) = \frac{1}{12}\left(-\pi^2 + 6\ln^2 c + 12\mathrm{Li}_2(c)\right),$$

где появляется стандартный полилогарифм $\mathrm{Li}_2(c) = c/1^2 + c^2/2^2 + c^3/3^2 + \ldots$. Затем воспользуйтесь любым современным пакетом, умеющим вычислять $\mathrm{Li}_2$ с высокой точностью, реализуйте метод Ньютона и тем самым избавьтесь от *самого* численного интегрирования. В частности, Вы должны уметь получать такие результаты, как, например,

$$c \approx 0.23043660131599974571471085700604655750807 54\ldots,$$

предполагаемые верными с подразумеваемой точностью.

Вот еще одно любопытное направление деятельности. Разработайте быстрый алгоритм, который по заданному значению $c$ подсчитывал бы количество целых чисел $n \in [1, x]$, для которых второй из наибольших простых делителей не превосходит $n^c$ (давайте не подсчитывать те числа $n$, у которых менее двух простых делителей). Для приведенного выше высокоточного значения $c$ существует ровно 548 таких чисел $n \in [1, 1000]$, тогда как теоретически должно было быть 500. Проследите изменение доли таких чисел при бóльших значениях $x$.

**3.6.** Усовершенствуйте базовый алгоритм для решета Эратосфена 3.2.1. Для начала уменьшите объем требуемой памяти (и увеличьте скорость), заметив, что каждое простое число $p > 3$ удовлетворяет сравнению $p \equiv \pm 1 \pmod 6$. Кроме того, можно также воспользоваться и другим модулем, большим 6.

**3.7.** При помощи критерия Корсельта (теорема 3.4.6) найдите вручную или на ЭВМ несколько чисел Кармайкла.

**3.8.** Докажите, что каждое составное число Ферма $F_n = 2^{2^n} + 1$ является псевдопростым числом Ферма по основанию 2. Может ли составное число Ферма быть псевдопростым числом Ферма по основанию 3? (Авторам не известен ни один подобный пример, причем доказательство отсутствия таких чисел также неизвестно.)

**3.9.** В этом упражнении мы рассмотрим грубые эвристические оценки, относящиеся к статистике, присущей определенным действиям с псевдопростыми числами. Знаменитая команда вычислителей и теоретиков супругов Лемер была первой из тех, кто еще в начале XX в. начал изучать тесты на простоту (а также огромное количество других вещей) при помощи ручной вычислительной техники. В частности, они установили простоту числа $(10^{23} - 1)/9$

при помощи механического вычислительного устройства у себя дома, как они рассказывали, работая понемногу каждый день на протяжении многих месяцев. Они готовили еду и трудились над доказательством простоты по очереди. Разумеется, позднее Лемеры при помощи ЭВМ смогли исследовать гораздо бóльшие числа.

Итак, упражнение заключается в следующем. Прокомментируйте статистическую закономерность, которая стоит за ответом Д. Лемера (1969) на вопрос одного студента: «Профессор Лемер, скажите, случалось ли Вам хотя бы раз за все время Ваших исследований в области простых чисел ошибиться, приняв псевдопростое число (т. е. составное число, прошедшее тест Ферма по основанию 2) за простое?», на который Лемер ответил чрезвычайно кратко: «Лишь раз». Итак, вот вопрос: каков статистический смысл этого ответа? Насколько часты псевдопростые числа по основанию 2 в окрестности $10^n$? Кроме того, по-видимому, нас не ввели бы в заблуждение, скажем, те псевдопростые числа по основанию 2, которые делятся на 3, так что переформулируйте вопрос для тех псевдопростых чисел по основанию 2, которые не имеют «малых» простых делителей. По этому вопросу см. статью [Damgård et al. 1993].

**3.10.** Заметим, что наименьшее допустимое значение $p$ для применения формулы в доказательстве теоремы 3.4.4 при $a = 2$ равно 5, и, как было указано, эта формула дает $n = 341$, наименьшее псевдопростое число по основанию 2. Если $a = 3$, то наименьшее допустимое значение $p$ равно 3, и эта формула дает $n = 91$, первое псевдопростое число по основанию 3. Покажите, что для бóльших значений $a$ такая закономерность нарушается и, более того, подобное не произойдет больше никогда.

**3.11.** Покажите, что если $n$ — число Кармайкла, то оно нечетно и имеет не менее трех простых делителей.

**3.12.** Покажите, что составное число $n$ является числом Кармайкла тогда и только тогда, когда $a^{n-1} \equiv 1 \pmod{n}$ для всех натуральных чисел $a$, взаимно простых с $n$.

**3.13.** [Beeger] Покажите, что если $p$ — фиксированное простое число, то существует не более чем конечное множество чисел Кармайкла, у которых второй по величине простой делитель равен $p$.

**3.14.** Для каждого натурального числа $n$ положим

$$\mathcal{F}(n) = \left\{a \pmod{n} : a^{n-1} \equiv 1 \pmod{n}\right\}.$$

(1) Покажите, что $\mathcal{F}(n)$ является подгруппой группы $\mathbf{Z}_n^*$ всех приведенных вычетов по модулю $n$, причем это собственная подгруппа тогда и только тогда, когда $n$ — составное число и не число Кармайкла.

(2) [Monier, Baillie–Wagstaff] Положим $F(n) = \#\mathcal{F}(n)$. Покажите, что

$$F(n) = \prod_{p \mid n} \text{НОД}\,(p-1, n-1).$$

(3) Обозначим через $F_0(n)$ число вычетов $a \pmod{n}$, таких, что $a^n \equiv a \pmod{n}$. Найдите формулу для $F_0(n)$, как это сделано в п. (2). Пока-

жите, что если $F_0(n) < n$, то $F_0(n) \leqslant \frac{2}{3}n$. Покажите, что если $n \neq 6$ и $F_0(n) < n$, то $F_0(n) \leqslant \frac{3}{5}n$. (Неизвестно, существует ли бесконечно много таких чисел $n$, что $F_0(n) = \frac{3}{5}n$, а также неизвестно, существует ли такое число $\varepsilon > 0$, что для бесконечно многих чисел $n$ справедливо двойное неравенство $\varepsilon n < F_0(n) < n$.)

Отметим, что доказан следующий факт: если $h(n)$ — произвольная функция, стремящаяся к бесконечности, то множество таких чисел $n$, для которых $F(n) < \ln^{h(n)} n$, имеет асимптотическую плотность 1 [Erdős and Pomerance 1986].

**3.15.** [Monier] Докажите, что в обозначениях лемм 3.5.8 и 3.5.9 для функции $S(n)$ (см. (3.5)) справедливо равенство

$$S(n) = \left(1 + \frac{2^{\nu(n)\omega(n)} - 1}{2^{\omega(n)} - 1}\right) \prod_{p \mid n} \text{НОД}\,(t, p-1).$$

**3.16.** [Haglund] Пусть $n$ — нечетное составное число. Докажите, что $\overline{\mathcal{S}}(n)$ является подгруппой группы $\mathbf{Z}_n^*$, порожденной множеством $\mathcal{S}(n)$.

**3.17.** [Gerlach] Пусть $n$ — нечетное составное число. Докажите, что $\mathcal{S}(n) = \overline{\mathcal{S}}(n)$ тогда и только тогда, когда $n$ или равно степени простого числа, или делится на простое число, сравнимое с 3 (mod 4). Выведите отсюда, что множество нечетных составных чисел $n$, для которых $\mathcal{S}(n)$ *не* является подгруппой группы $\mathbf{Z}_n^*$, бесконечно, но имеет асимптотическую плотность нуль (см. упр. 1.10, 1.91 и 5.16).

**3.18.** Пусть нам даны нечетное число $n$ и такое число $a$, не кратное $n$, что $n$ является псевдопростым числом по основанию $a$, но не является сильно псевдопростым числом по этому основанию. Опишите алгоритм, который по поданным на входе таким числам возвращает нетривиальное разложение на множители числа $n$ за полиномиальное время.

**3.19.** [Lenstra, Granville] Покажите, что если нечетное число $n$ кратно квадрату некоторого простого числа, то $W(n)$ (наименьший свидетель для $n$) меньше $\ln^2 n$. (Указание: воспользуйтесь (1.45).) См. также упр. 4.28.

**3.20.** Опишите вероятностный алгоритм, возвращающий нетривиальное разложение на множители числа Кармайкла за полиномиальное время в среднем.

**3.21.** Будем говорить, что нечетное составное число $n$ является псевдопростым числом Эйлера по основанию $a$, если числа $a$ и $n$ взаимно просты и справедливо сравнение

$$a^{(n-1)/2} \equiv \left(\frac{a}{n}\right) \pmod{n}, \tag{3.32}$$

где $\left(\frac{a}{n}\right)$ — символ Якоби (см. определение 2.3.3). Критерий Эйлера (см. теорему 2.3.4) утверждает, что нечетные простые числа $n$ удовлетворяют сравнению (3.32). Покажите, что если $n$ — сильно псевдопростое число по основанию $a$, то оно является и псевдопростым числом Эйлера по основанию $a$, а если $n$ — псевдопростое число Эйлера по основанию $a$, то оно является и псевдопростым числом по основанию $a$.

**3.22.** [Lehmer, Solovay–Strassen] Пусть $n$ — нечетное составное число. Докажите, что множество вычетов $a \pmod n$, для которых $n$ является псевдопростым числом Эйлера, образует собственную подгруппу группы $\mathbf{Z}_n^*$. Выведите отсюда, что количество таких оснований $a$ не превосходит $\varphi(n)/2$.

**3.23.** Следуя методам алгоритма 3.5.6, разработайте вероятностный тест на составность, опирающийся на упр. 3.22. (Этот тест обычно называют тестом Соловея—Штрассена на простоту.) На основе упр. 3.21 покажите, что этот алгоритм слабее алгоритма 3.5.6.

**3.24.** [Lenstra, Robinson] Докажите, что если число $n$ нечетно и существует такое число $b$, что $b^{(n-1)/2} \equiv -1 \pmod n$, то каждое число $a$, удовлетворяющее сравнению $a^{(n-1)/2} \equiv \pm 1 \pmod n$, удовлетворяет и сравнению $a^{(n-1)/2} \equiv \left(\frac{a}{n}\right) \pmod n$. Выведите отсюда и упр. 3.22, что если $n$ — нечетное составное число, причем $a^{(n-1)/2} \equiv \pm 1 \pmod n$ для всех $a$, взаимно простых с $n$, то на самом деле $a^{(n-1)/2} \equiv 1 \pmod n$ для всех $a$, взаимно простых с $n$. Это число $n$ обязано быть числом Кармайкла (см. упр. 3.12). (Из доказательства бесконечности множества чисел Кармайкла вытекает существование бесконечного множества таких нечетных составных чисел $n$, что $a^{(n-1)/2} \equiv \pm 1 \pmod n$ для всех $a$, взаимно простых с $n$. Наименьшим из таких чисел является «номер такси» Рамануджана: 1729.)

**3.25.** Покажите, что существуют ровно семь псевдопростых чисел Фибоначчи, меньших 323.

**3.26.** Покажите, что каждое составное число, взаимно простое с 6, является псевдопростым числом Люка по отношению к $x^2 - x + 1$.

**3.27.** Покажите, что если справедливо (3.12), то

$$(a-x)^n \equiv \begin{cases} x \pmod{(f(x), n)}, & \text{если } \left(\frac{\Delta}{n}\right) = -1, \\ a - x \pmod{(f(x), n)}, & \text{если } \left(\frac{\Delta}{n}\right) = 1. \end{cases}$$

В частности, убедитесь, что псевдопростое число Фробениуса по отношению к $f(x) = x^2 - ax + b$ одновременно является и псевдопростым числом Люка по отношению к $f(x)$.

**3.28.** Покажите, что определение псевдопростого числа Фробениуса в разд. 3.6.5 для многочлена $f(x) = x^2 - ax + b$ сводится к определению из разд. 3.6.2.

**3.29.** Покажите, что если натуральное число $n$ нечетно и взаимно просто с натуральным числом $a$, то $n$ является псевдопростым числом Ферма по основанию $a$ тогда и только тогда, когда $n$ является псевдопростым числом Фробениуса по отношению к двучлену $f(x) = x - a$.

**3.30.** Пусть целые числа $a, b$ таковы, что $\Delta = a^2 - 4b$ не равно квадрату целого числа. Положим $f(x) = x^2 - ax + b$ и $R = \mathbf{Z}_n[x]/(f(x))$. Пусть, далее, нечетное простое число $n$ не делит $b\Delta$. Докажите, что если $(x(a-x)^{-1})^{2m} = 1$ в кольце $R$, то $(x(a-x)^{-1})^m = \pm 1$ в кольце $R$.

**3.31.** Покажите, что псевдопростое число Фробениуса по отношению к трехчлену $x^2 - ax + b$ одновременно является и псевдопростым числом Эйлера по отношению к $b$ (см. упр. 3.21).

**3.32.** Докажите все тождества из разд. 3.6.3.

**3.33.** Докажите рекуррентное соотношение (3.22).

**3.34.** Покажите, что если $a = \pi\left(x^{1/3}\right)$, то число слагаемых двойной суммы в (3.23) есть $O\left(x^{2/3}/\ln^2 x\right)$.

**3.35.** Покажите, что при помощи $M$ компьютеров, где $M < x^{1/3}$, с объемом памяти $O\left(x^{1/3+\epsilon}\right)$, алгоритм для подсчета простых чисел из разд. 3.7 можно ускорить в $M$ раз.

**3.36.** Покажите, что вместо аналитического соотношения (3.27) для получения модифицированной функции $\pi^*(x)$ можно было при желании воспользоваться «дзета-функцией по простым»

$$\mathcal{P}(s) = \sum_{p \in \mathcal{P}} \frac{1}{p^s},$$

подставив ее интеграл вместо $\ln \zeta$, вследствие чего результат в левой части (3.27) для нецелого $x$ совпал бы с самой функцией $\pi$. Затем покажите, что это наблюдение не лишено смысла и способно приносить пользу, выведя при $\operatorname{Re} s > 1$ соотношение

$$\mathcal{P}(s) = \sum_{n=1}^{\infty} \frac{\mu(n)}{n} \ln \zeta(ns)$$

и описав в количественном выражении относительную легкость вычисления значений $\zeta(ns)$ для больших целых чисел $n$.

**3.37.** Получите теоретические оценки величины действительной части интеграла

$$\int_T^{\infty} \frac{e^{it\alpha}}{\beta + it}\, dt,$$

где $T, \alpha, \beta$ — положительные действительные числа, и оцените вклад части интеграла в соотношении (3.27) по множеству $\operatorname{Im}(s) > T$. Затем укажите, насколько большим должно быть значение $T$, чтобы вычисленное значение $\pi^*(x)$ отличалось от истинного значения не более чем на заданное число $\varepsilon > 0$ (см. упр. 3.47 и 3.39 для ознакомления с аналогичными оценками в гораздо более эффективных методах).

**3.38.** Рассмотрим конкретный выбор функции отключения Лагариаса—Одлыжко $c(u, x)$, а именно, прямолинейное соединение значений 1 и 0. Положим $y = \sqrt{x}$ и $c = 1, (x-u)/y, 0$ при $u \leqslant x-y, u \in (x-y, x], u > x$ соответственно. Покажите, что двойственная по Меллину функция равна

$$F(s, x) = \frac{1}{y} \frac{x^{s+1} - (x-y)^{s+1}}{s(s+1)}.$$

Затем получите оценку (как в упр. 3.37) значений $T$ для вычисления $\pi(x)$ с заданной точностью при помощи соотношения

$$\pi^*(x) \approx \operatorname{Re} \int_0^T F(s,x) \ln \zeta(s)\, dt.$$

Найдите несколько точных численных значений $\pi(x)$ при помощи именно этой функции отключения $c$.

**3.39.** В связи с рассмотренной в тексте функцией Гэлвея (см. (3.31)) проведите строгое обоснование того утверждения, что хотя дзета-функция Римана в некотором смысле содержит, если хотите, «все секреты простых чисел», но для подсчета всех простых чисел, не превосходящих $x$, нам требуется уметь вычислять $\zeta$ лишь до высоты «примерно» $x^{1/2}$.

**3.40.** Посредством интегрирования по частям покажите, что функция $F$, определенная равенством (3.31), и в самом деле является преобразованием Меллина заданной функции $c$.

## 3.9. Проблемы для исследования

**3.41.** Найдите число $n \equiv \pm 2 \pmod 5$, которое одновременно являлось бы и псевдопростым числом по основанию 2, и псевдопростым числом Фибоначчи. Померанс, Селфридж и Вагштаф предлагают \$620 за первый такой пример. (Должно быть предъявлено разложение на простые множители.) Денежный приз выплатят все трое, но не в равных долях: Селфридж предлагает \$500, Вагштаф — \$100, а Померанс — \$20. Однако они также готовы выплатить \$620 за доказательство отсутствия такого числа $n$, при этом Померанс и Селфридж поменяются ролями.

**3.42.** Найдите составное число $n$ (вместе с разложением на простые множители), которое являлось бы псевдопростым числом Фробениуса для $x^2 + 5x + 5$ и удовлетворяло равенству $\left(\frac{5}{n}\right) = -1$. Дж. Грэнтхэм установил приз в \$6.20 за первый пример.

**3.43.** Рассмотрим функцию наименьшего свидетеля $W(n)$, определенную на множестве нечетных составных чисел $n$. Докажите относительно нетрудное утверждение, что $W(n)$ никогда не является точной степенью. Какими еще свойствами должны обладать значения функции $W(n)$? Если существует такое число $n$, что $W(n) = k$, то обозначим наименьшее из таких чисел $n$ через $n_k$. Д. Блейхенбахер установил (см. также статьи [Jaeschke 1993], [Zhang and Tang 2003] и упр. 4.34), что

$$\begin{aligned}
n_2 &= 9 & n_{12} &> 10^{16} \\
n_3 &= 2047 & n_{13} &= 2152302898747 \\
n_5 &= 1373653 & n_{14} &= 1478868544880821 \\
n_6 &= 134670080641 & n_{17} &= 3474749660383 \\
n_7 &= 25326001 & n_{19} &= 4498414682539051 \\
n_{10} &= 307768373641 & n_{23} &= 341550071728321. \\
n_{11} &= 3215031751
\end{aligned}$$

С. Ли показал, что $W(n) = 12$ при

$$n = 1502401849747176241,$$

поэтому мы знаем, что число $n_{12}$ существует. Найдите $n_{12}$ и продолжите эту таблицу. Благодаря вычислениям Д. Блейхенбахера, мы знаем, что все остальные существующие значения $n_k$ должны превосходить $10^{16}$.

**3.44.** Изучите возможную альтернативу обычному алгоритму пробных делений 3.1.1, рассмотрев операции вычисления НОД для разложения на множители числа $N$ (это может показаться несколько экстравагантным). Например, можно вычислять факториалы числа $B$ и находить НОД$(B!, N)$ с целью нахождения очередного делителя. Опишите этот алгоритм полностью так, чтобы он возвращал полное разложение на простые множители. Эта задача не так проста: в частности, заметим, что делитель $k^2$ числа $N$, для которого $k < B$, может быть не найден при помощи одного факториала.

Затем рассмотрите вопросы, связанные со сложностью. Можно ли повысить эффективность, используя в качестве $\{B_i\}$ произведения последовательных простых чисел, т. е. частичных «приориалов» (см. упр. 1.6), и рассматривая НОД$(B_i, N)$?

**3.45.** Пусть $f(N)$ обозначает время, которое требуется для распознавания простоты самого «плохого» числа из промежутка $[N, N + N^{1/4}]$. После просеивания через простые числа, меньшие $N^{1/4}$, останутся лишь те числа этого промежутка, у которых нет простых делителей, меньших $N^{1/4}$. Количество таких чисел равно $O(N^{1/4}/\ln N)$. Таким образом, все простые числа рассматриваемого промежутка могут быть найдены за время, равное $O(N^{1/4} f(N)/\ln N) + O(N^{1/4} \ln\ln N)$. Можно ли улучшить этот результат до $o(N^{1/4} f(N)/\ln N)$ или $O(N^{1/4} \ln\ln N)$?

**3.46.** Как обсуждалось выше, обычное решето Эратосфена можно разбить на сегменты так, чтобы на всем пути вычислений, за исключением итогового списка простых чисел, требовался объем памяти $O(N^{1/2})$. Причем можно реализовать этот процесс без увеличения числа битовых операций $O(N \ln\ln N)$. Можно ли создать таблицу простых чисел, не превосходящих $N$, за $o(N)$ битовых операций, использовав лишь объем памяти $O(N^{1/2})$? Алгоритм 3.2.2 удовлетворяет такому ограничению по времени, но не по объему памяти. (В статье [Atkin and Bernstein 2004] эта проблема почти решена.)

**3.47.** Следуя формальным рассуждениям разд. 3.7.2, получите интегральное условие на $x, \Delta$, связанное с дзета-функцией Римана, чтобы на промежутке $[x, x + \Delta]$ *не* существовало простых чисел. Опишите применение такого критерия для численного (но строгого) доказательства существования или отсутствия простых чисел в этом интервале для данных значений $x$, $\Delta$. Разумеется, было бы интересно получить новые *теоретические* подходы к анализу длин промежутков между соседними простыми числами.

**3.48.** Пусть T — вероятностный тест, устанавливающий составность числа $n$ с вероятностью $p(n)$. (Это если $n$ — составное число, а если на вход подано простое число, то тест T сообщает о неудаче при доказательстве его составности.)

Существует ли такой тест T, что $p(n) \to 1$ при $n$, стремящемся к бесконечности по множеству составных чисел, и такой, что время работы теста T для числа $n$ не превосходит времени работы $k$ тестов на псевдопростоту для $n$, где $k$ — некоторое фиксированное число?

**3.49.** Определим функцию $K(n)$ на множестве бесквадратных натуральных чисел $n$, взаимно простых с 12, следующим образом:

$$\begin{aligned} K(n) &= \#\{(u,v) : u > v > 0;\ n = u^2 + v^2\}, \text{ если } n \equiv 1, 5 \pmod{12}, \\ K(n) &= \#\{(u,v) : u > 0,\ v > 0;\ n = 3u^2 + v^2\}, \text{ если } n \equiv 7 \pmod{12}, \\ K(n) &= \#\{(u,v) : u > v > 0;\ n = 3u^2 - v^2\}, \text{ если } n \equiv 11 \pmod{12}. \end{aligned}$$

В работе [Atkin and Bernstein 2004] доказано, что число $n$ — простое тогда и только тогда, когда $K(n)$ нечетно. Докажите эту теорему, быть может, применив тот факт (или аналогичный ему), что количество представлений любого натурального числа $n$ в виде суммы двух квадратов

$$r_2(n) = 4 \sum_{d \mid n,\ d \text{ нечетно}} (-1)^{(d-1)/2},$$

причем мы подсчитываем и те представления $n = u^2 + v^2$, где числа $u$ или $v$ отрицательны (для проверки укажем, что $r_2(25) = 12$).

Возникает проблема для исследований: разработайте эффективное решето для простых чисел на промежутке, используя теорему Аткина—Бернштейна и проверяя четность значения $K$ для многих чисел $n$ одновременно (см. работу [Galway 2000]).

Другой вопрос состоит в том, чтобы создать эффективное решето (или даже тест на простоту) при помощи альтернативных описаний $r_2(n)$, например, использующих разнообразные связи с дзета-функцией Римана (см. [Titchmarsh 1986]).

Еще один исследовательский вопрос звучит так: какова сложность «подсчета» всех узлов решетки (в трех указанных областях), находящихся внутри заданного «радиуса» $\sqrt{n}$, и поиска чисел представлений $K(n)$ через величину скачка при данном радиусе. На первый взгляд этот метод может показаться довольно грубым, но существуют *эффективные* формулы, возникающие при рассмотрении знаменитой проблемы круга (проблемы Гаусса) (сколько точек решетки лежит внутри круга заданного радиуса?), позволяющие удивительно быстро подсчитать точное число точек. Кроме того, докажите еще одну теорему из теории решеток о том, что если бесквадратное число $n \equiv 1 \pmod 4$, то $n$ является простым тогда и только тогда, когда $r_2(n) = 8$. Простой численный эксперимент для начинающих, доказывающий простоту числа $n = 13$ путем подсчета точек по аналитическим формулам Бесселя, описан в книге [Crandall 1994b, p. 68].

**3.50.** В заключение этой главы мы затронем аналитический метод подсчета простых чисел, основанный на определенных свойствах дзета-функции Римана. В соответствии с упр. 1.60, рассмотрим следующее направление исследо-

ваний, в котором будет использоваться информация о дзета-функции внутри критической полосы (а не справа от нее).

Начнем с формулы Римана—Мангольдта для функции $\pi^*$, определенной соотношением (3.25), близко напоминающую равенство (1.23):

$$\pi^*(x) = \mathrm{li}_0(x) - \sum_\rho \mathrm{li}_0(x^\rho) - \ln 2 + \int_x^\infty \frac{dt}{t(t^2-1)\ln t}$$

(здесь следует напомнить о мерах вычислительной предосторожности из упр. 1.36, связанных с использованием Ei для получения достоверных результатов). Символом $\rho$ обозначены нули дзета-функции Римана в критической полосе, причем соответствующую сумму можно заменить на удвоенную сумму по действительным частям этих нулей.

Проблема для исследования такова: найдите быстрый алгоритм, позволяющий на основании набора нулей дзета-функции Римана в критической полосе вычислять $\pi(x)$ с высокой точностью. Ясно, что, имея в распоряжении малое число нулей, можно вычислить $\pi(x)$ как целозначную лестницу, по меньшей мере, до некоторой границы, зависящей от количества этих нулей. Более сложная проблема состоит в том, как по заданному $x$ определить, насколько высоко следует подняться по критической полосе, собирая значения $\rho$, для получения такой численной аппроксимации (назовем ее $\pi_a(x)$), чтобы для каждого натурального $n \in [2, x]$ выполнялось *строгое* равенство $\pi(n) = \lfloor \pi_a(n + 1/2) \rfloor$. Разумеется, теоретические соображения подсказывают, что для этого потребуется не менее $O(\sqrt{x})$ значений $\rho$, но мысль заключается в том, что нам хотелось бы получить аналитически точную функцию $\pi_a(x)$ на некотором промежутке изменения переменной $x$.

Подсчет количества простых чисел, основанный на привлечении нулей дзета-функции Римана в критической полосе, описан в работах [Riesel and Göhl 1970] и [Borwein et al. 2000].

Глава 4

# ДОКАЗАТЕЛЬСТВО ПРОСТОТЫ ЧИСЕЛ

В главе 3 мы рассматривали быстрые вероятностные методы распознавания составных чисел. Если проход такого вероятностного теста не распознает число как составное, то либо оно простое, либо нам не повезло. Исходя из того, что нам не может постоянно не везти, после некоторого количества проходов мы приходим к убеждению, что число является простым. Однако, сколько бы раз мы не проводили такой тест, мы не получим *доказательства* простоты числа. У нас будет только предположение, подтвержденное вычислительными экспериментами. Эта глава посвящена тому, как доказать простоту числа. Заметим, что доказательство простоты при помощи эллиптических кривых обсуждается в разд. 7.6.

## 4.1. $(n-1)$-тест

Простоту небольших чисел можно доказать при помощи пробного деления, но для больших чисел существуют более быстрые методы (пробным делением можно проверить числа, порядок которых не больше, чем $10^{12}$, хотя эта величина зависит от быстродействия компьютера). Один из более быстрых методов основан на той же теореме, что и самый простой тест на псевдопростоту, а именно на малой теореме Ферма (теорема 3.4.1). Известный как $(n-1)$-тест, этот метод немного своеобразен тем, что вместо разложения на множители числа $n$ мы ищем делители числа $n-1$.

### 4.1.1. Теорема Люка и тест Пепена

Начнем с идеи, выдвинутой Люка в 1876 г.

**Теорема 4.1.1** (теорема Люка). *Пусть $a, n$ — целые числа, $n > 1$; если*

$$\begin{aligned} &a^{n-1} \equiv 1 \pmod{n}, \\ &\text{но } a^{(n-1)/q} \not\equiv 1 \pmod{n} \text{ для любого простого числа } q \mid n-1, \end{aligned} \tag{4.1}$$

*то число $n$ является простым.*

Доказательство. Из первого условия в (4.1) следует, что порядок элемента $a$ в группе $\mathbf{Z}_n^*$ является делителем числа $n-1$. А из второго условия следует, что порядок элемента $a$ не может быть делителем числа $n-1$, меньшим $n-1$, т. е. порядок элемента $a$ равен $n-1$. По теореме Эйлера (см. (2.2)) порядок элемента $a$ является делителем $\varphi(n)$, следовательно, $n-1 \leqslant \varphi(n)$. Допустим, что $n$ является составным числом и имеет простой делитель $p$. Тогда оба числа $p$ и

$n$ являются целыми, лежат на отрезке $\{1, 2, \ldots, n\}$ и не взаимно просты с числом $n$. По определению функции Эйлера $\varphi(n)$ (оно дано после равенств (1.5)) $\varphi(n) \leqslant n-2$. Получаем противоречие с неравенством $n-1 \leqslant \varphi(n)$, следовательно, число $n$ является простым. □

**Замечание.** Такую формулировку теоремы 4.1.1 на самом деле дал Лемер. Люка доказал эту теорему, когда $q$ пробегает все (не обязательно простые) делители числа $n-1$.

Условие (4.1) теоремы Люка не является бессодержательным для простых чисел. Число $a$ с таким свойством называется первообразным корнем. И кроме того, для любого простого $n$ первообразный корень существует. Т. е. если $n$ является простым числом, то мультипликативная группа $\mathbf{Z}_n^*$ будет циклической, см. теорему 2.2.5. На самом деле каждое простое число $n > 200560490131$ имеет не менее $n/(2\ln\ln n)$ первообразных корней в промежутке $\{1, 2, \ldots, n-1\}$, см. упр. 4.1. (Замечание: простое число 200560490131 равно произведению первых 11 простых чисел, увеличенному на 1.)

Как следствие получаем, что если число $n > 200560490131$ является простым, то число $a$, удовлетворяющее условию (4.1), довольно просто найти при помощи следующего вероятностного алгоритма: будем брать случайные целые числа $a$ на отрезке $1 \leqslant a \leqslant n-1$ и проверять условие (4.1). Среднее число попыток для нахождения нужного нам числа не превосходит $2\ln\ln n$.

Хотя мы и не знаем детерминированного алгоритма для нахождения первообразного корня за полиномиальное время, основная проблема при использовании теоремы Люка в качестве теста на простоту состоит *не* в нахождении первообразного корня $a$, а в разложении на множители числа $n-1$. Разложение на множители многих, но не всех, чисел является очень сложной вычислительной задачей. Например, предположим, что мы ищем простые числа, которые на 1 больше степени 2. Как показано в теореме 1.3.4, такие простые числа имеют вид $F_k = 2^{2^k}+1$. Эти числа называются числами Ферма в честь Пьера Ферма, предполагавшего, что все эти числа простые.

В 1877 г. Пепен привел критерий простоты чисел Ферма. Следующая теорема представляет собой немного измененный критерий Пепена.

**Теорема 4.1.2** (тест Пепена). *При $k \geqslant 1$ число $F_k = 2^{2^k}+1$ является простым тогда и только тогда, когда $3^{(F_k-1)/2} \equiv -1 \pmod{F_k}$.*

Доказательство. Предположим, что сравнение верно. Тогда условие (4.1) выполняется при $n = F_k$, $a = 3$, следовательно, $F_k$ является простым числом по теореме Люка 4.1.1. Обратно, пусть $F_k$ — простое число. Так как $2^k$ — четное число, то $2^{2^k} \equiv 1 \pmod 3$, следовательно, $F_k \equiv 2 \pmod 3$. Но $F_k \equiv 1 \pmod 4$, поэтому символ Лежандра $\left(\frac{3}{F_k}\right)$ равен $-1$. Следовательно, 3 не является квадратичным вычетом $(\mathrm{mod}\ F_k)$. Необходимое сравнение следует из критерия Эйлера (2.6). □

На самом деле Пепен привел критерий с числом 5 вместо числа 3 (и при $k \geqslant 2$). Прот и Люка отметили, что можно также использовать число 3. Подробнее об этом см. [Williams 1998] и упр. 4.5.

На момент выхода этой книги самое большое число Ферма $F_k$, для которого применялся тест Пепена, — это $F_{24}$. Как обсуждалось в разд. 1.3.2, это число является составным, и на самом деле все числа Ферма, большие $F_4$, с известным характером (простое или составное) являются составными.

### 4.1.2. Частичное разложение на множители

Так как самым трудным шагом в применении теоремы Люка 4.1.1 в качестве теста на простоту является полное разложение числа $n-1$ на простые сомножители, вполне естественно возникает вопрос: можно ли что-либо сказать о простоте числа $n$, используя частичное разложение числа $n-1$? Пусть, например,

$$\begin{aligned} &n-1=FR \\ &\textit{и у числа } F \textit{ известно полное разложение на простые сомножители.} \end{aligned} \tag{4.2}$$

Если число $F$ достаточно велико как функция от числа $n$, мы можем получить доказательство простоты числа $n$ по тому же принципу, что и в (4.1), если, конечно, число $n$ окажется простым. Наш первый результат в этом направлении позволяет получить некоторую информацию о разложении на множители числа $n$.

**Теорема 4.1.3** (Поклингтон). *Пусть выполняется условие* (4.2), *и число* $a$ *таково, что*

$$\begin{aligned} &a^{n-1}\equiv 1 \pmod{n} \\ &\textit{и } \text{НОД}\,(a^{(n-1)/q}-1,n)=1 \textit{ для всех простых чисел } q|F. \end{aligned} \tag{4.3}$$

*Тогда каждый простой делитель числа* $n$ *сравним с* $1 \pmod{F}$.

Доказательство. Пусть число $p$ является простым делителем числа $n$. Из первой части условия (4.3) получаем, что порядок элемента $a^R$ в мультипликативной группе $\mathbf{Z}_p^*$ является делителем числа $(n-1)/R=F$. Из второй части условия (4.3) получаем, что порядок не может быть делителем числа $F$, меньшим $F$. Следовательно, порядок числа $a^R$ равен $F$. Следовательно, число $F$ делит порядок группы $\mathbf{Z}_p^*$, который равен $p-1$. Этот факт завершает доказательство теоремы. □

**Следствие 4.1.4.** *Если выполняются условия* (4.2) *и* (4.3), *и кроме того,* $F\geqslant\sqrt{n}$, *то число* $n$ *является простым.*

Доказательство. Теорема 4.1.3 утверждает, что все простые делители числа $n$ сравнимы с $1 \pmod{F}$, следовательно, все простые делители числа $n$ больше числа $F$. Но $F\geqslant\sqrt{n}$, следовательно, все простые делители числа $n$ больше, чем $\sqrt{n}$, следовательно, число $n$ должно быть простым. □

Следующая теорема показывает, что число $F$ может быть и меньше, чем $\sqrt{n}$.

**Теорема 4.1.5** (Бриллхарт, Лемер, Селфридж). *Пусть выполнены условия* (4.2) *и* (4.3), *и пусть* $n^{1/3}\leqslant F<n^{1/2}$. *Рассмотрим представление числа*

*$n$ в $F$-ичной системе счисления, т. е. $n = c_2F^2 + c_1F + 1$, где $c_1$, $c_2$ — целые числа из отрезка $[0, F-1]$. Тогда число $n$ является простым тогда и только тогда, когда число $c_1^2 - 4c_2$ не является полным квадратом.*

Доказательство. Так как $n \equiv 1 \pmod F$, то у числа $n$ цифра, соответствующая единицам в $F$-ичной системе счисления, равна 1. Следовательно, представление числа $n$ в $F$-ичной системе счисления имеет указанный вид $c_2F^2+c_1F+1$. Предположим, что число $n$ является составным. По теореме 4.1.3 все простые делители числа $n$ сравнимы с 1 $\pmod F$, следовательно, превосходят $n^{1/3}$. Отсюда следует, что число $n$ имеет ровно два простых делителя:

$$n = pq, \quad p = aF + 1, \quad q = bF + 1, \quad a \leqslant b.$$

Следовательно,

$$c_2F^2 + c_1F + 1 = n = (aF + 1)(bF + 1) = abF^2 + (a + b)F + 1.$$

Наша цель — доказать выполнение равенств: $c_2 = ab$ и $c_1 = a+b$, откуда будет следовать, что число $c_1^2 - 4c_2$ является полным квадратом.

Сначала заметим, что $F^3 \geqslant n > abF^2$, следовательно, $ab \leqslant F - 1$. Отсюда имеем, что либо $a + b \leqslant F - 1$, либо $a = 1, b = F - 1$. В последнем случае получаем $n = (F + 1)((F - 1)F + 1) = F^3 + 1$, что противоречит условию $F \geqslant n^{1/3}$. Следовательно, $ab$ и $a + b$ являются положительными целыми числами, меньшими $F$. Из единственности представления числа в $F$-ичной системе счисления получаем искомые равенства $c_2 = ab$ и $c_1 = a + b$.

Теперь обратно, предположим, что число $c_1^2 - 4c_2$ является полным квадратом $u^2$. Тогда

$$n = \left(\frac{c_1 + u}{2}F + 1\right)\left(\frac{c_1 - u}{2}F + 1\right).$$

Так как $c_1 \equiv u \pmod 2$, то обе скобки в правой части равенства являются целыми числами. Остается только заметить, что мы получили нетривиальное разложение на множители числа $n$. Это следует из того, что $c_2 > 0$ и $|u| < c_1$. Следовательно, число $n$ является составным. □

Для применения теоремы 4.1.5 в качестве теста на простоту нам надо уметь быстро распознавать, является ли целое число $c_1^2 - 4c_2$ из условия теоремы полным квадратом. Это можно сделать при помощи алгоритма 9.2.11.

Следующий результат показывает, что число $F$ можно еще уменьшить.

**Теорема 4.1.6** (Конягин и Померанс). *Пусть $n \geqslant 214$, выполнены условия* (4.2) *и* (4.3), *и пусть $n^{3/10} \leqslant F < n^{1/3}$. Пусть представление числа $n$ в $F$-ичной системе счисления имеет вид $c_3F^3 + c_2F^2 + c_1F + 1$, обозначим $c_4 = c_3F + c_2$. Тогда число $n$ является простым тогда и только тогда, когда выполнены следующие условия:*

*(1) число $(c_1 + tF)^2 + 4t - 4c_4$ не является полным квадратом при $t = 0, 1, 2, 3, 4, 5$;*

*(2) если $u/v$ — непрерывная дробь, приближающая $c_1/F$, такая что $v$ максимально при условии $v < F^2/\sqrt{n}$, и если $d = \lfloor c_4v/F + 1/2 \rfloor$, то многочлен $vx^3 + (uF - c_1v)x^2 + (c_4v - dF + u)x - d \in \mathbf{Z}[x]$ не имеет целых*

*корней $a$, таких, что число $aF+1$ является нетривиальным делителем числа $n$.*

ДОКАЗАТЕЛЬСТВО. Так как любой простой делитель числа $n$ сравним с 1 (mod $F$) (по теореме 4.1.3), получаем, что число $n$ является составным тогда и только тогда, когда существуют положительные целые числа $a_1 \leqslant a_2$, такие, что $n = (a_1F+1)(a_2F+1)$. Предположим, что число $n$ является составным и выполнены условия (1) и (2). Начнем с установления некоторых тождеств и неравенств. Мы имеем

$$n = c_4F^2 + c_1F + 1 = a_1a_2F^2 + (a_1+a_2)F + 1,$$

и существует некоторое целое число $t \geqslant 0$, такое, что

$$a_1a_2 = c_4 - t, \quad a_1 + a_2 = c_1 + tF. \tag{4.4}$$

Так как условие (1) выполнено, то $t \geqslant 6$. Следовательно,

$$a_2 \geqslant \frac{a_1+a_2}{2} \geqslant \frac{c_1+6F}{2} \geqslant 3F,$$

и

$$a_1 < \frac{n}{a_2F^2} \leqslant \frac{n}{3F^3}. \tag{4.5}$$

Из равенств (4.4) получаем, что

$$t \leqslant \frac{a_1+a_2}{F} \leqslant \frac{a_1a_2+1}{F} < \frac{c_4}{F} < \frac{n}{F^3}. \tag{4.6}$$

Из равенств (4.4) следует также, что

$$a_1c_1 + a_1tF = a_1^2 + c_4 - t. \tag{4.7}$$

Из условия (2) и равенства (4.7) получаем, что

$$\begin{aligned} a_1u + a_1tv - \frac{c_4v}{F} &= a_1v\left(\frac{u}{v} - \frac{c_1}{F}\right) + (a_1c_1 + a_1tF)\frac{v}{F} - \frac{c_4v}{F} \\ &= a_1v\left(\frac{u}{v} - \frac{c_1}{F}\right) + (a_1^2 + c_4 - t)\frac{v}{F} - \frac{c_4v}{F} \\ &= a_1v\left(\frac{u}{v} - \frac{c_1}{F}\right) + (a_1^2 - t)\frac{v}{F}. \end{aligned} \tag{4.8}$$

Из неравенств (4.5), (4.6) и $t \geqslant 6$ следует, что

$$|a_1^2 - t| < \max\{a_1^2, t\} \leqslant \max\left\{\frac{1}{9}\left(\frac{n}{F^3}\right)^2, \frac{n}{F^3}\right\} \leqslant \frac{1}{6}\left(\frac{n}{F^3}\right)^2. \tag{4.9}$$

Сначала предположим, что $u/v = c_1/F$. Тогда из равенств (4.8) и неравенств (4.9) будет следовать, что

$$\left|a_1u + a_1tv - \frac{c_4v}{F}\right| = |a_1^2 - t|\frac{v}{F} < \frac{1}{6}\left(\frac{n}{F^3}\right)^2\frac{v}{F} < \frac{n^2}{6F^7}\cdot\frac{F^2}{\sqrt{n}} = \frac{n^{3/2}}{6F^5} \leqslant \frac{1}{6}. \tag{4.10}$$

Если $u/v \neq c_1/F$, то пусть $u'/v'$ — следующая непрерывная дробь, приближающая $c_1/F$ после $u/v$, так что

$$v < \frac{F^2}{\sqrt{n}} \leqslant v', \quad \left|\frac{u}{v} - \frac{c_1}{F}\right| \leqslant \frac{1}{vv'} \leqslant \frac{\sqrt{n}}{vF^2}.$$

Тогда из неравенств (4.5) и тождеств (4.8) аналогично вычислениям в формуле (4.10) получаем

$$\left|a_1u + a_1tv - \frac{c_4v}{F}\right| \leqslant a_1v\frac{\sqrt{n}}{vF^2} + \frac{1}{6} < \frac{n^{3/2}}{3F^5} + \frac{1}{6} \leqslant \frac{1}{2}.$$

Пусть число $d = a_1u + a_1tv$ таково, что $|d - c_4v/F| < 1/2$, следовательно, число $d = \lfloor c_4v/F + 1/2 \rfloor$. Умножив обе части равенства (4.7) на $a_1v$, получим

$$va_1^3 - c_1va_1^2 - a_1^2tvF - a_1tv + c_4a_1v = 0,$$

учитывая равенство $-a_1tv = ua_1 - d$, получим

$$va_1^3 + (uF - c_1v)a_1^2 + (c_4v - dF + u)a_1 - d = 0.$$

Следовательно, условие (2) не выполняется; таким образом, если число $n$ является составным, то либо условие (1), либо условие (2) не выполняются.

Теперь предположим, что число $n$ является простым. Если $t \in \{0,1,2,3,4,5\}$ и $(c_1 + tF)^2 - 4c_4 + 4t = u^2$, где $u$ — целое число, то

$$\begin{aligned} n &= (c_4 - t)F^2 + (c_1 + tF)F + 1 \\ &= \left(\frac{c_1 + tF + u}{2}F + 1\right)\left(\frac{c_1 + tF - u}{2}F + 1\right). \end{aligned}$$

Так как число $n$ является простым, то это разложение на множители должно быть тривиальным, т. е.

$$c_1 + tF - |u| = 0,$$

откуда следует, что $c_4 = t$. Но $c_4 \geqslant F \geqslant n^{3/10} \geqslant 214^{3/10} > 5 \geqslant t$, получаем противоречие. Следовательно, если условие (1) не выполняется, то число $n$ является составным. Очевидно, что если число $n$ является простым, то условие (2) выполняется. □

Так же как в случае с теоремой 4.1.5, если мы хотим использовать теорему 4.1.6 как тест на простоту числа, мы должны использовать алгоритм 9.2.11 для распознавания полных квадратов. Также нам придется использовать метод Ньютона или метод деления пополам для нахождения целых корней кубического многочлена в условии (2) теоремы. Мы объединили результаты теорем 4.1.3–4.1.6 в следующем алгоритме.

**Алгоритм 4.1.7** ($(n-1)$-тест). Пусть нам дано целое число $n \geqslant 214$, и пусть выполнено условие (4.2) при числе $F \geqslant n^{3/10}$. Данный вероятностный алгоритм пытается определить, является число $n$ простым (ДА) или составным (НЕТ).

1. [Тест Поклингтона]
   Выберем произвольное $a \in [2, n-2]$;
   if($a^{n-1} \not\equiv 1 \pmod n$) return НЕТ; // Число $n$ является составным.
   for(по простым $q|F$) {
   $g = \text{НОД}\left((a^{(n-1)/q} \bmod n) - 1, n\right)$;
   if($1 < g < n$) return НЕТ;
   if($g == n$) goto [тест Поклингтона]
   } // Конец цикла означает выполнение соотношения (4.3).

2. [Тест первого уровня]
   if($F \geqslant n^{1/2}$) return ДА;
3. [Тест второго уровня]
   if($n^{1/3} \leqslant F < n^{1/2}$) {
   Получить представление числа $n$ в $F$-ичной системе счисления: $n = c_2F^2 + c_1F + 1$;
   if($c_1^2 - 4c_2$ не является полным квадратом) return ДА;
   return НЕТ;
   }
4. [Тест третьего уровня]
   if($n^{3/10} \leqslant F < n^{1/3}$) {
   Если выполнены условия (1) и (2) теоремы 4.1.6, return ДА;
   return НЕТ;
   }

Хотя алгоритм 4.1.7 и является вероятностным, любой его ответ ДА (число $n$ — простое) или НЕТ (число $n$ — составное) является строгим.

### 4.1.3. Краткие сертификаты

Целью тестов на простоту является нахождение короткого доказательства простоты для простых чисел $p$. Но откуда мы знаем, что такое короткое доказательство существует? Любой поиск обречен на неудачу, если для числа $p$ не существует короткого доказательства простоты. Сейчас мы покажем, что для любого простого числа $p$ существует короткое доказательство его простоты; В. Пратт назвал это «кратким сертификатом».

На самом деле всегда существует короткое доказательство простоты, основанное на теореме Люка 4.1.1. Это может показаться очевидным, так как если нам удалось как-то найти полное разложение числа $p-1$ на простые сомножители и первообразный корень $a$, то условия (4.1) легко проверяются.

Однако для завершения доказательства нам надо показать, что мы действительно получили полное разложение числа $p-1$ на простые сомножители, т. е. что числа $q$, появляющиеся в условии (4.1), действительно являются простыми. Здесь мы должны использовать наш метод еще раз, но таким образом нам придется рассматривать несколько случаев. Сущность доказательства состоит в том, чтобы показать, что даже при самом худшем варианте мы получим не очень много случаев для рассмотрения.

Целесообразно сделать небольшую, но очень полезную модификацию теоремы Люка 4.1.1. Идея состоит в том, чтобы обрабатывать простое число $q = 2$ отдельно от остальных простых чисел $q$, делящих число $p-1$. На самом деле мы знаем, с чем должно быть сравнимо число $a^{(p-1)/2} \pmod p$, если это не 1, а именно, с $-1$. Если $a^{(p-1)/2} \equiv -1 \pmod p$, то нам не надо проверять сравнение $a^{p-1} \equiv 1 \pmod p$. Далее, если число $q$ является нечетным простым делителем числа $p-1$, положим $m = a^{(p-1)/2q}$. Если $m^q \equiv -1 \pmod p$ и $m^2 \equiv 1 \pmod p$, то $m \equiv -1 \pmod p$ (независимо от того, является число $p$ простым

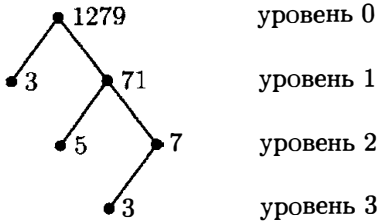
или составным). Следовательно, чтобы показать, что $a^{(p-1)/q} \not\equiv 1 \pmod p$, достаточно показать, что $a^{(p-1)/2q} \not\equiv -1 \pmod p$. Таким образом, мы получаем следующий результат.

**Теорема 4.1.8.** *Пусть $p > 1$ — нечетное число, и*

$$\begin{cases} a^{(p-1)/2} \equiv -1 \pmod p, \\ a^{(p-1)/2q} \not\equiv -1 \pmod p \text{ для любого нечетного простого числа } q | p-1. \end{cases} \tag{4.11}$$

*Тогда число $p$ является простым. Обратно, если число $p$ является нечетным простым числом, то каждый первообразный корень $a$ числа $p$ удовлетворяет условиям* (4.11).

Сейчас мы опишем так называемое «дерево Люка». Это корневое дерево с нечетными простыми числами в вершинах; число $p$ является корневым (уровень 0). На каждом положительном уровне $k$ простое число $r$ на уровне $k$ соединено с простым числом $q$ на уровне $k-1$ тогда и только тогда, когда $r | q-1$. Ниже приведен пример дерева Люка для простого числа $p = 1279$:




Обозначим через $M(p)$ количество модульных умножений (на целые числа, не превосходящие $p$), необходимых для доказательства простоты числа $p$ с использованием теоремы 4.1.8 для прохождения дерева Люка для числа $p$ и двоичных цепей для возведения в степень (см. алгоритм 2.1.5).

Например, пусть $p = 1279$:

$$\begin{aligned} 7^{1278/2} \equiv -1 \pmod{1279} \quad &, \quad 7^{1278/6} \equiv 504 \pmod{1279}, \\ &\quad 7^{1278/142} \equiv 1157 \pmod{1279}, \\ 2^{2/2} \equiv -1 \pmod 3 \quad &, \\ 7^{70/2} \equiv -1 \pmod{71} \quad &, \quad 7^{70/10} \equiv 14 \pmod{71}, \\ &\quad 7^{70/14} \equiv 51 \pmod{71}, \\ 2^{4/2} \equiv -1 \pmod 5 \quad &, \\ 3^{6/2} \equiv -1 \pmod 7 \quad &, \quad 3^{6/6} \equiv 3 \pmod 7, \\ 2^{2/2} \equiv -1 \pmod 3 & \end{aligned}$$

Если мы будем использовать двоичное представление для всех экспонент, мы

получим следующее количество модульных умножений:

| | |
|---|---|
| 1278/2 | 16 |
| 1278/6 | 11 |
| 1278/142 | 4 |
| 2/2 | 0 |
| 70/2 | 7 |
| 70/10 | 4 |
| 70/14 | 3 |
| 4/2 | 1 |
| 6/2 | 2 |
| 6/6 | 0 |
| 2/2 | 0. |

Следовательно, используя двоичное представление, нам потребуется 48 модульных умножений, т. е. $M(1279) = 48$.

Следующий результат по существу взят из работы [Pratt 1975]:

**Теорема 4.1.9.** *Пусть $p$ — нечетное простое число, тогда $M(p) < 2\lg^2 p$*[1].

ДОКАЗАТЕЛЬСТВО. Обозначим через $N(p)$ количество (не обязательно различных) нечетных простых чисел в дереве Люка для числа $p$. Сначала покажем, что $N(p) < \lg p$. Это неравенство верно для числа $p = 3$. Предположим, что неравенство выполняется для всех нечетных простых чисел, меньших числа $p$. Если число $p - 1$ является степенью числа 2, то $N(p) = 1 < \lg p$. Если $p - 1$ раскладывается в произведение нечетных простых сомножителей $q_1, \ldots, q_k$, то по предположению индукции

$$N(p) = 1 + \sum_{i=1}^{k} N(q_i) < 1 + \sum_{i=1}^{k} \lg q_i = 1 + \lg(q_1 \ldots q_k) \leqslant 1 + \lg\left(\frac{p-1}{2}\right) < \lg p.$$

Поэтому неравенство $N(p) < \lg p$ всегда выполнено.

Если $r$ — одно из нечетных простых чисел, находящихся в дереве Люка для числа $p$, и $r < p$, то существует также другое простое число $q$, находящееся в дереве Люка, такое, что $r | q - 1$ и $q \leqslant p$. Мы показали, что, с одной стороны, существует число $a$, такое, что $a^{(q-1)/2r} \not\equiv -1 \pmod q$, и, с другой стороны, существует число $b$, такое, что $b^{(r-1)/2} \equiv -1 \pmod r$. Отметим также, что количество модульных умножений при использовании двоичного представления для числа $m$ не превосходит числа $2\lg m$. Следовательно, количество модульных умножений в двух сравнениях, написанных выше, не превосходит числа

$$2\lg\left(\frac{q-1}{2r}\right) + 2\lg\left(\frac{r-1}{2}\right) < 2\lg q - 4 < 2\lg p.$$

[1] Под lg понимается логарифм по основанию 2. — *Прим. ред.*

Получаем, что

$$M(p) < 2\lg\left(\frac{p-1}{2}\right) + (N(p)-1)2\lg p < 2\lg p + (\lg p - 1)2\lg p = 2\lg^2 p.$$

Доказательство закончено. □

Используя более эффективные цепи сложений, можно уменьшить коэффициент 2. На данный момент неизвестно, существует ли число $c > 0$, такое, что для бесконечного числа простых чисел $p$ доказательство простоты числа $p$ при помощи дерева Люка требует не менее $c\lg^2 p$ модульных умножений. Мы также не знаем, существует ли бесконечное число простых чисел $p$, таких, что $M(p) = o(\lg^2 p)$. Однако известно, что по теореме 7.6.1 (см. [Pomerance 1987a]), в принципе, существует *некоторое* доказательство простоты любого простого $p$, использующее только $O(\lg p)$ модульных умножений. Из дерева Люка мы получаем существование короткого доказательства простоты числа, но найти его мы не можем.

## 4.2. $(n+1)$-тест

Основная проблема при использовании $(n-1)$-теста для доказательства простоты числа $n$ состоит в нахождении достаточно большого, полностью разложенного на множители делителя числа $n-1$. Для некоторых чисел $n$ это не проблема, например, для чисел Ферма, для которых можно использовать тест Пепена. Для некоторых чисел, например, для чисел Мерсенна $M_p = 2^p - 1$, очевидно разложение на множители числа $n+1$. Естественно возникает вопрос, можем ли мы как-нибудь использовать эту информацию для проверки чисел на простоту? Оказывается, можем.

### 4.2.1. Тест Люка—Лемера

Пусть $a, b \in \mathbf{Z}$, рассмотрим

$$f(x) = x^2 - ax + b,\ \Delta = a^2 - 4b. \tag{4.12}$$

Напомним последовательности Люка $(U_k), (V_k)$, которые мы уже обсуждали в разд. 3.6.1:

$$U_k = \frac{x^k - (a-x)^k}{x - (a-x)} \pmod{f(x)}, \quad V_k = x^k + (a-x)^k \pmod{f(x)}. \tag{4.13}$$

Напомним также: многочлены $U_k, V_k$ не могут иметь ненулевую положительную степень, поэтому являются целыми числами.

**Определение 4.2.1.** Пусть натуральное число $n$ удовлетворяет условию $\text{НОД}(n, 2b\Delta) = 1$. Рангом появления числа $n$ (обозначается $r_f(n)$) называется наименьшее натуральное число $r$, такое, что $U_r \equiv 0 \pmod{n}$.

Это понятие иногда называют «рангом явления» («rank of apparition»), но по словам Рибенбойма это произошло из-за неправильного перевода на ан-

глийский язык французского слова *apparition*. В понятии ранга появления нет ничего призрачного!

Из определения (4.13) видно, что последовательность $(U_k)$ является «последовательностью, сохраняющей делимость», т. е. если $k|j$, то и $U_k|U_j$. (Здесь мы допускаем возможность того, что $U_k = U_j = 0$.) Отсюда следует, что если $\text{НОД}(n, 2b\Delta) = 1$, то $U_j \equiv 0 \pmod n$ тогда и только тогда, когда $j \equiv 0 \pmod{r_f(n)}$. Таким образом, на основе теоремы 3.6.3 мы получаем следующий результат:

**Теорема 4.2.2.** *Пусть $f, \Delta$ обозначают то же, что и в* (4.12), *и пусть $p$ — простое число, не делящее $2b\Delta$. Тогда $r_f(p)|p - \left(\frac{\Delta}{p}\right)$.*

(Символ Лежандра $\left(\frac{\cdot}{p}\right)$ введен в определении 2.3.2.)

По аналогии с теоремой 4.1.3 мы получаем следующий результат:

**Теорема 4.2.3** (Моррисон). *Пусть $f, \Delta$ обозначают то же, что и в* (4.12), *и пусть $n$ — натуральное число, удовлетворяющее условиям* $\text{НОД}(n, 2b) = 1$, $\left(\frac{\Delta}{n}\right) = -1$. *Если число $F$ является делителем числа $n+1$, и*

$$U_{n+1} \equiv 0 \pmod n, \quad \text{НОД}(U_{(n+1)/q}, n) = 1 \text{ для всех простых чисел } q|F, \tag{4.14}$$

*то любое простое число $p$, делящее число $n$, удовлетворяет условию $p \equiv \left(\frac{\Delta}{p}\right) \pmod F$. В частности, если число $F > \sqrt{n}+1$, и выполняются условия* (4.14), *то число $n$ является простым.*

(Символ Якоби $\left(\frac{\cdot}{n}\right)$ введен в определении 2.3.3.)

ДОКАЗАТЕЛЬСТВО. Пусть число $p$ — простой делитель числа $n$. Тогда из условий (4.14) следует, что число $F$ делит ранг появления $r_f(p)$. Следовательно, по теореме 4.2.2 имеем $p \equiv \left(\frac{\Delta}{p}\right) \pmod F$. Так как $F > \sqrt{n}+1$, любой простой делитель $p$ числа $n$ удовлетворяет неравенству $p \geqslant F - 1 > \sqrt{n}$, следовательно, число $n$ является простым. □

Для использования теоремы 4.2.3 в качестве теста на простоту нам необходимо найти подходящий многочлен $f$ в (4.12). Так же как в алгоритме 4.1.7, где число $a$ выбиралось случайным образом, мы можем и здесь выбрать числа $a, b$ в (4.12) случайным образом. Как показывает следующий результат, количество выборов, необходимое для нахождения подходящей пары чисел $a, b$, невелико.

**Теорема 4.2.4.** *Пусть $f, \Delta$ обозначают то же, что и в* (4.12). *Пусть также $p$ — нечетное простое число, и $N$ — количество пар чисел $a, b \in \{0, 1, \ldots, p\text{-}1\}$, для которых $\left(\frac{\Delta}{p}\right) = -1$ и $r_f(p) = p+1$. Тогда $N = \frac{1}{2}(p-1)\varphi(p+1)$.*

Оставим доказательство этой теоремы в качестве упражнения 4.12. Результат теоремы 4.2.4 состоит в следующем: нам дано нечетное простое число $n$, мы случайно выбираем пару чисел $a, b$ из отрезка $\{0, 1, \ldots, n-1\}$, причем $a$ и $b$ оба не равны нулю. Тогда ожидаемое количество выборов пары $a, b$, пока не будет найдена пара, удовлетворяющая условию $r_f(n) = n+1$, равно $2(n+1)/\varphi(n+1)$. Если число $n > 892271479$, то ожидаемое количество выборов меньше, чем $4 \ln \ln n$ (см. упр. 4.16).

Также можно применить тест на простоту, использующий последовательность $V$ из определения (4.13).

**Теорема 4.2.5.** *Пусть $f, \Delta$ обозначают то же, что и в* (4.12), *и пусть $n$ — натуральное число, удовлетворяющее условиям* $\text{НОД}(n, 2b) = 1$ *и* $\left(\frac{\Delta}{n}\right) = -1$. *Если число $F$ является четным делителем числа $n+1$, и*

$$V_{F/2} \equiv 0 \pmod{n},\ \text{НОД}(V_{F/2q}, n) = 1 \text{ для любого нечетного простого } q|F, \tag{4.15}$$

*то каждое простое число $p$, делящее число $n$, удовлетворяет условию $p \equiv \left(\frac{\Delta}{p}\right) \pmod{F}$. В частности, если $F > \sqrt{n}+1$, то число $n$ является простым.*

Доказательство. Предположим, что $p$ — нечетное простое число, делящее $U_m$ и $V_m$. Тогда из определения (4.13) следует, что $x^m \equiv (a-x)^m \pmod{(f(x), p)}$ и $x^m \equiv -(a-x)^m \pmod{f(x), p}$. Следовательно, $x^m \equiv 0 \pmod{(f(x), p)}$. Тогда $b^m \equiv (x(a-x))^m \equiv 0 \pmod{(f(x), p)}$, т. е. число $p$ делит число $b$. Так как число $n$ взаимно просто с числом $2b$, и $U_{2m} = U_m V_m$, мы имеем

$$\text{НОД}(U_{2m}, n) = \text{НОД}(U_m, n) \cdot \text{НОД}(V_m, n).$$

Из первого условия (4.15) следует, что $U_F \equiv 0 \pmod{n}$ и $\text{НОД}(U_{F/2}, n) = 1$. Предположим теперь, что число $q$ является нечетным простым делителем числа $F$. Мы имеем, что число $U_{F/q} = U_{F/2q} V_{F/2q}$ взаимно просто с числом $n$. На самом деле число $U_{F/2q}$ делит число $U_{F/2}$, т. е. $\text{НОД}(U_{F/2q}, n) = 1$. Из второго условия (4.15) мы получаем, что $\text{НОД}(U_{F/q}, n) = 1$. Следовательно, $r_f(p) = F$; этого равенства достаточно для вывода результата теоремы. □

Так же как $(n-1)$-тест хорошо подходит для чисел Ферма, $(n+1)$-тест очень быстро работает на числах Мерсенна.

**Теорема 4.2.6** (тест Люка—Лемера для чисел Мерсенна). *Рассмотрим последовательность чисел $(v_k)$, где $k = 0, 1, \ldots$, определяемую рекуррентно из следующих условий: $v_0 = 4$ и $v_{k+1} = v_k^2 - 2$. Пусть $p$ — нечетное простое число. Число $M_p = 2^p - 1$ является простым тогда и только тогда, когда $v_{p-2} \equiv 0 \pmod{M_p}$.*

Доказательство. Положим $f(x) = x^2 - 4x + 1$, тогда $\Delta = 12$. Так как $M_p \equiv 3 \pmod 4$ и $M_p \equiv 1 \pmod 3$, то $\left(\frac{\Delta}{M_p}\right) = -1$. Применим теорему 4.2.5 с числом $F = 2^{p-1} = (M_p + 1)/2$. Условия (4.15) сводятся к одному условию $V_{2^{p-2}} \equiv 0 \pmod{M_p}$. Но

$$V_{2m} \equiv x^{2m} + (4-x)^{2m} = (x^m + (4-x)^m)^2 - 2x^m(4-x)^m \equiv V_m^2 - 2 \pmod{f(x)},$$

так как $x(4-x) \equiv 1 \pmod{f(x)}$ (см. (3.15)). Имеем также, что $V_1 = 4$. Следовательно, $V_{2^k} = v_k$, и из теоремы 4.2.5 следует, что если $v_{p-2} \equiv 0 \pmod{M_p}$, то число $M_p$ является простым.

Предположим, что число $M = M_p$ является простым. Так как $\left(\frac{\Delta}{M}\right) = -1$, то $\mathbf{Z}[x]/(f(x), M)$ изоморфно конечному полю $\mathbf{F}_{M^2}$. Следовательно, возведение числа в степень $M$ является автоморфизмом, и $x^M \equiv 4 - x \pmod{(f(x), M)}$ (см. доказательство теоремы 3.6.3). Вычислим выражение $(x-1)^{M+1}$ двумя способами. Первый: так как $(x-1)^2 \equiv 2x \pmod{(f(x), M)}$, и по критерию

Эйлера $2^{(M-1)/2} \equiv (2/M) = 1 \pmod{M}$, то

$$\begin{aligned}(x-1)^{M+1} &\equiv (2x)^{(M+1)/2} = 2\cdot 2^{(M-1)/2}x^{(M+1)/2}\\ &\equiv 2x^{(M+1)/2} \pmod{(f(x), M)}.\end{aligned}$$

Второй:

$$\begin{aligned}(x-1)^{M+1} = (x-1)(x-1)^M &\equiv (x-1)(x^M-1) \equiv (x-1)(3-x)\\ &\equiv -2 \pmod{(f(x), M)}.\end{aligned}$$

Следовательно, $x^{(M+1)/2} \equiv -1 \pmod{(f(x), M)}$, т. е. $x^{2^{p-1}} \equiv -1 \pmod{(f(x), M)}$. Используя наш автоморфизм, мы также получим, что $(4-x)^{2^{p-1}} \equiv -1 \pmod{(f(x), M)}$, поэтому $U_{2^{p-1}} \equiv 0 \pmod{M}$. Если $U_{2^{p-2}} \equiv 0 \pmod{M}$, то $x^{2^{p-2}} \equiv (4-x)^{2^{p-2}} \pmod{(f(x), M)}$, следовательно,

$$-1 \equiv x^{2^{p-1}} \equiv x^{2^{p-2}}(4-x)^{2^{p-2}} \equiv (x(4-x))^{2^{p-2}} \equiv 1^{2^{p-2}} \equiv 1 \pmod{(f(x), M)},$$

получили противоречие. Так как $U_{2^{p-1}} = U_{2^{p-2}}V_{2^{p-2}}$, мы имеем $V_{2^{p-2}} \equiv 0 \pmod{M}$. Но мы уже показали, что $V_{2^{p-2}} = v_{p-2}$, следовательно, доказательство закончено. □

**Алгоритм 4.2.7** (тест Люка—Лемера для чисел Мерсенна). Нам дано нечетное простое число $p$. Этот алгоритм определяет, является число $2^p - 1$ простым (ДА) или составным (НЕТ).

1. [Инициализация]
   $v = 4$;
2. [Вычисление последовательности Люка—Лемера]
   for($k \in [1, p-2]$) $v = (v^2 - 2) \bmod (2^p - 1)$; // $k$ — счетчик.
3. [Проверка остатка]
   if($v == 0$) return ДА; // $2^p - 1$ является простым.
   return НЕТ; // $2^p - 1$ является составным.

Указанный тест Люка—Лемера для чисел Мерсенна довольно успешно применяется, что также упоминается в гл. 1 и в обсуждении алгоритма 9.5.19. Не только потому, что тест поразительно простой, но также и потому, что существуют эффективные методы вычисления $p-2$ квадратов на шаге [Вычисление последовательности Люка—Лемера].

### 4.2.2. Улучшенный $(n+1)$-тест и объединенный $(n^2-1)$-тест

Для использования $(n-1)$-теста нам необходимо найти большой, полностью разложенный на множители делитель числа $n-1$; точно так же, основная сложность применения $(n+1)$-теста для всех чисел состоит в нахождении большого, полностью разложенного на множители делителя числа $n+1$. В этом разделе мы улучшим теорему 4.2.3 и получим результат, похожий на теорему 4.1.5. Т. е. нам будет достаточно, чтобы полностью разложенный на множители делитель числа $n+1$ превосходил кубический корень. (Используя идеи теоремы 4.1.6,

этот результат можно улучшить до степени $3/10$.) Затем мы покажем, как можно использовать в одном тесте полностью разложенные на множители делители двух чисел $n-1$ и $n+1$, т. е. полностью разложенные на множители делители числа $n^2-1$.

**Теорема 4.2.8.** *Пусть $f, \Delta$ обозначают то же, что и в* (4.12). *Пусть также $n$ — натуральное число, удовлетворяющее условиям* $\text{НОД}(n, 2b) = 1$ *и* $\left(\frac{\Delta}{n}\right) = -1$. *Предположим, что $n+1 = FR$, где $F > n^{1/3}+1$ и выполняется условие* (4.14). *Запишем число $R$ в $F$-ичной системе счисления, т. е. $R = r_1F + r_0$, где $0 \leqslant r_i \leqslant F-1$. Число $n$ является простым тогда и только тогда, когда многочлены $x^2 + r_0x - r_1$ и $x^2 + (r_0+F)x - r_1 - 1$ не имеют натуральных корней.*

Заметим, что если $R < F$, то $r_1 = 0$, и ни один из указанных многочленов не имеет натуральных корней. Следовательно, теорема 4.2.8 содержит последнее утверждение теоремы 4.2.3.

Доказательство. Из теоремы 4.2.3 следует, что любой простой делитель $p$ числа $n$ удовлетворяет условию $p \equiv \left(\frac{\Delta}{p}\right) \pmod F$. Следовательно, если число $n$ является составным, то оно имеет всего два простых делителя: $n = pq$. Действительно, если число $n$ имеет 3 или более простых делителей, то оно превосходит число $(F-1)$; получаем противоречие. Так как $-1 = \left(\frac{\Delta}{n}\right) = \left(\frac{\Delta}{p}\right)\left(\frac{\Delta}{q}\right)$, то без ограничения общности мы можем считать, что $\left(\frac{\Delta}{p}\right) = 1$, $\left(\frac{\Delta}{q}\right) = -1$. Следовательно, существуют натуральные числа $c, d$, такие, что $p = cF+1$, $q = dF-1$. Так как $(F^2+1)(F-1) > n$, и $(F+1)(F^2-1) > n$, то $1 \leqslant c, d \leqslant F-1$. Заметим, что

$$r_1F + r_0 = R = \frac{n+1}{F} = cdF + d - c,$$

следовательно, $d - c \equiv r_0 \pmod F$. Получаем, что $d = c + r_0$ или $d = c + r_0 - F$, т. е. $d = c + r_0 - iF$, где $i = 0$ или $1$. Таким образом,

$$r_1F + r_0 = c(c + r_0 - iF)F + r_0 - iF,$$

следовательно, $r_1 = c(c + r_0 - iF) - i$, откуда получаем

$$c^2 + (r_0 - iF)c - r_1 - i = 0.$$

Но тогда многочлен $x^2 + (r_0 - iF) - r_1 - i$ имеет натуральный корень при каком-то $i = 0, 1$. В одну сторону теорема доказана.

Предположим теперь, что многочлен $x^2 + (r_0 - iF) - r_1 - i$ имеет натуральный корень $c$ при каком-то $i = 0, 1$. Производя предыдущие вычисления в обратном порядке, мы получаем, что число $cF+1$ является делителем числа $n$. Но $n \equiv -1 \pmod F$, следовательно, число $n$ является составным, так как по предположению $F > 2$. □

Можно еще более улучшить $(n+1)$-тест, так что нам будет достаточен делитель $F \geqslant n^{3/10}$. Доказательство этого факта полностью аналогично теореме 4.1.6, и мы оставим его в качестве упражнения 4.15.

**Теорема 4.2.9.** *Пусть $n \geqslant 214$, и выполнены условия теоремы* 4.2.8, *за тем лишь исключением, что $n^{3/10} \leqslant F \leqslant n^{1/3}+1$. Назовем $F$-ичным разложением*

*числа $n+1$ выражение вида $c_3F^3+c_2F^2+c_1F$ и положим $c_4 = c_3F+c_2$. Число $n$ является простым тогда и только тогда, когда выполняются следующие условия:*

*(1) число $(c_1+tF)^2-4t+4c_4$ не является полным квадратом при целом $t$, $|t| \leqslant 5$;*

*(2) пусть $u/v$ — непрерывная дробь, приближающая число $c_1/F$, с максимальным знаменателем $v$, удовлетворяющим неравенству $v < F^2/\sqrt{n}$. Пусть также $d = \lfloor c_4v/F + 1/2 \rfloor$. Многочлен $vx^3-(uF-c_1v)x^2-(c_4v-dF+u)x+d$ не имеет целого корня $a$, такого, что число $aF+1$ является нетривиальным делителем числа $n$. Многочлен $vx^3+(uF-c_1v)x^2-(c_4v+dF+u)x+d$ не имеет целого корня $b$, такого, что число $bF-1$ является нетривиальным делителем числа $n$.*

Следующая теорема позволяет объединить частичное разложение на множители чисел $n-1$ и $n+1$ для доказательства простоты числа $n$.

**Теорема 4.2.10** (Бриллхарт, Лемер, Селфридж). *Пусть $n$ — натуральное число, $F_1|n-1$, и для некоторого целого числа $a_1$ и числа $F = F_1$ выполняется условие* (4.3). *Пусть $f, \Delta$ обозначают то же, что и в* (4.12), НОД$(n, 2b) = 1$, $\left(\frac{\Delta}{n}\right) = -1$, $F_2|n+1$, *и для числа $F = F_2$ выполняется условие* (4.14). *Пусть число $F$ равно наименьшему общему кратному чисел $F_1, F_2$. Тогда каждый простой делитель числа $n$ сравним либо с 1, либо с $n \pmod F$. В частности, если $F > \sqrt{n}$, и число $n \bmod F$ не является нетривиальным делителем числа $n$, то число $n$ — простое.*

Заметим, что если оба числа $F_1, F_2$ — четные, то $F = \frac{1}{2}F_1F_2$. Если же хотя бы одно из этих чисел нечетное, то $F = F_1F_2$.

Доказательство. Пусть $p$ — простой делитель числа $n$. Из теоремы 4.1.3 следует, что $p \equiv 1 \pmod{F_1}$. Из теоремы 4.2.3 следует, что $p \equiv \left(\frac{\Delta}{p}\right) \pmod{F_2}$. Если $\left(\frac{\Delta}{p}\right) = 1$, то $p \equiv 1 \pmod F$. Если $\left(\frac{\Delta}{p}\right) = -1$, то $p \equiv n \pmod F$. Последнее утверждение теоремы очевидно. □

## 4.2.3. Делители в классах вычетов

Что делать, если в теореме 4.2.10 мы получим $F < n^{1/2}$? Теорема будет полезна, если мы сможем быстро находить простые делители числа $n$, сравнимые с 1 или с $n \pmod F$. Следующий алгоритм (см. [Lenstra 1984]) позволяет быстро находить такие делители, если число $F/n^{1/3}$ не слишком мало.

**Алгоритм 4.2.11** (делители в классах вычетов). **Нам даны натуральные числа $n, r, s$, удовлетворяющие условиям $r < s < n$ и НОД$(r, s) = 1$. Этот алгоритм находит все делители числа $n$, сравнимые с $r \pmod s$.**

1. **[Инициализация]**
   $r^* = r^{-1} \bmod s$;
   $r' = nr^* \bmod s$;
   $(a_0, a_1) = (s, r'r^* \bmod s)$;

$(b_0, b_1) = (0, 1)$;
$(c_0, c_1) = (0, r^*(n - rr')/s \bmod s)$;

2. [Евклидовы цепи]
Рассмотрим евклидовы последовательности $(a_i), (q_i)$, где $a_i = a_{i-2} - q_i a_{i-1}$ и $0 \leqslant a_i < a_{i-1}$ при четном $i$, $0 < a_i \leqslant a_{i-1}$ при нечетном $i$. Оканчиваются последовательности при $a_t = 0$, где $t$ — четно;
Рассмотрим последовательности $(b_i), (c_i)$, где $i = 0, 1, \ldots, t$, строящиеся по правилам: $b_i = b_{i-2} - q_i b_{i-1}$ и $c_i = c_{i-2} - q_i c_{i-1}$;

3. [Цикл]
for($0 \leqslant i \leqslant t$) {
Решим следующую систему уравнений (относительно переменных $x, y$) для каждого целого числа $c \equiv c_i \pmod s$, удовлетворяющего условиям $|c| < s$, если $i$ четно, и $2a_i b_i < c < a_i b_i + n/s^2$, если $i$ нечетно:

$$xa_i + yb_i = c, \quad (xs + r)(ys + r') = n; \tag{4.16}$$

Если пара целых неотрицательных чисел $(x, y)$ является решением системы, то число $xs + r$ является делителем числа $n$, сравнимым с $r \pmod s$;
}

Теоретическое обоснование этого алгоритма дает следующая теорема:

**Теорема 4.2.12** (Ленстра). *Алгоритм* 4.2.11 *создает список всех делителей числа $n$, сравнимых с $r \pmod s$. Более того, если $s \geqslant n^{1/3}$, алгоритм работает за $O(\ln n)$ арифметических операций с целыми числами порядка $O(n)$ и $O(\ln n)$ оценок целой части квадратного корня из числа порядка $O(n^7)$.*

Доказательство. Сначала укажем некоторые свойства последовательностей $(a_i), (b_i)$. Мы имеем

$$a_i > 0 \text{ при } 0 \leqslant i < t, \ a_t = 0. \tag{4.17}$$

Также верно равенство

$$b_{i+1}a_i - a_{i+1}b_i = (-1)^i s \text{ при } 0 \leqslant i < t. \tag{4.18}$$

На самом деле, при $i = 0$ соотношение (4.18) выполнено. Далее по индукции, если $0 < i < t$, и соотношение выполнено при $i - 1$, то

$$\begin{aligned} b_{i+1}a_i - a_{i+1}b_i &= (b_{i-1} - q_{i+1}b_i)a_i - (a_{i-1} - q_{i+1}a_i)b_i \\ &= b_{i-1}a_i - a_{i-1}b_i \\ &= (-1)^i s, \end{aligned}$$

т. е. соотношение (4.18) выполнено при всех $0 \leqslant i < t$.

Заметим также, что

$$b_0 = 0, \ b_i < 0 \text{ при четном } i \neq 0, \ b_i > 0 \text{ при нечетном } i. \tag{4.19}$$

При $i = 0, 1$ соотношение (4.19) выполняется. Далее по индукции, из $b_i = b_{i-2} - q_i b_{i-1}$ и $q_i > 0$ соотношение выполняется для всех $i$, если оно выполняется для $i - 1, i - 2$.

Предположим теперь, что число $xs + r$ ($x \geqslant 0$) является делителем числа $n$. Покажем, что алгоритм найдет его. Существует целое число $y \geqslant 0$, такое, что $n = (xs + r)(ys + r')$. Имеем

$$xa_i + yb_i \equiv c_i \pmod{s} \text{ при } 0 \leqslant i \leqslant t. \tag{4.20}$$

Равенство (4.20) очевидно выполняется при $i = 0$, из $n = (xs + r)(ys + r')$ и определения числа $c_1$ оно выполняется при $i = 1$, и для больших значений $i$ оно следует из индуктивного определения последовательностей $(a_i), (b_i), (c_i)$.

Таким образом, достаточно показать, что существует либо некоторое четное значение числа $i$, удовлетворяющее условию $|xa_i + yb_i| < s$, либо существует некоторое нечетное значение числа $i$, удовлетворяющее условию $2a_ib_i < xa_i + yb_i < a_ib_i + n/s^2$. Отсюда и из сравнения (4.20) будет следовать, что число $xa_i + yb_i$ является одним из чисел $c$, вычисленных на шаге [Цикл] алгоритма 4.2.11. Следовательно, на шаге [Цикл] мы получим пару чисел $x, y$.

Мы имеем $xa_0 + yb_0 = xa_0 \geqslant 0$ и $xa_t + yb_t = yb_t \leqslant 0$, поэтому существует некоторое значение числа $i$, удовлетворяющее условиям

$$xa_i + yb_i \geqslant 0, \quad xa_{i+2} + yb_{i+2} \leqslant 0.$$

Если хотя бы одна из этих величин по модулю меньше, чем $s$, то теорема доказана. Поэтому мы предположим, что $xa_i + yb_i \geqslant s$, и $xa_{i+2} + yb_{i+2} \leqslant -s$. Тогда из условий (4.17), (4.18), (4.19) мы имеем:

$$xa_i \geqslant xa_i + yb_i \geqslant s = b_{i+1}a_i - a_{i+1}b_i \geqslant b_{i+1}a_i,$$

откуда следует, что $x \geqslant b_{i+1}$. Также мы имеем

$$yb_{i+2} \leqslant xa_{i+2} + yb_{i+2} \leqslant -s = b_{i+2}a_{i+1} - a_{i+2}b_{i+1} < b_{i+2}a_{i+1},$$

поэтому $y > a_{i+1}$. Следовательно,

$$xa_{i+1} + yb_{i+1} > 2a_{i+1}b_{i+1},$$

и из $(x - b_{i+1})(y - a_{i+1}) > 0$ мы получим

$$xa_{i+1} + yb_{i+1} \leqslant xy + a_{i+1}b_{i+1} < a_{i+1}b_{i+1} + \frac{n}{s^2}.$$

Это завершает доказательство корректности алгоритма.

Предположение о количестве арифметических операций следует из теоремы 2.1.3 и алгоритма 2.1.4. Из этих результатов следует, что количество арифметических операций для вычисления $r^*$ можно оценить как $t = O(\ln n)$. Более того, если $s \geqslant n^{1/3}$, то для каждого значения числа $i$ существует не более двух значений числа $c$, для которых надо решать систему уравнений (4.16). Решение такой системы, как мы покажем позже, требует $O(1)$ арифметических операций и извлечение квадратного корня. Следовательно, всего требуется $O(\ln n)$ арифметических операций и извлечений квадратного корня.

Осталось только определить размер целых чисел, для которых нам надо найти целую часть квадратного корня. Заметим, что пара чисел $x, y$ является решением системы (4.16) тогда и только тогда, когда числа $u = a_i(xs + r)$, $v = b_i(ys + r')$ являются корнями квадратного трехчлена

$$T^2 - (cs + a_ir + b_ir')T + a_ib_in.$$

Чтобы этот квадратный трехчлен имел целые корни, необходимо, чтобы его дискриминант

$$\Delta = (cs + a_i r + b_i r')^2 - 4a_i b_i$$

был полным квадратом. Покажем, что $\Delta = O(s^7) = O(n^7)$. Для этого обозначим $B = \max\{|b_i|\}$ и покажем, что $B < s^{5/2}$. Отсюда, так как все числа $c, a_i, r, r'$ ограничены по модулю числом $2s$, мы получим, что $\Delta = O(s^7)$. (Чтобы показать, что $|c| < 2s$, заметим, что $|c| < s$ при четных $i$; при нечетном $i$ интервал $(2a_i b_i, a_i b_i + n/s^2)$ должен содержать целые числа, следовательно, $0 < a_i b_i < n/s^2 \leqslant s$.)

Для получения оценки числа $B$ заметим, что

$$|b_i| = |b_{i-2}| + q_i |b_{i-1}| \text{ при } i = 2, \ldots, t,$$

следовательно,

$$B = |b_t| < \prod_{i=2}^{t} (1 + q_i) < 2^t \prod_{i=2}^{t} q_i.$$

Но $a_{i-2} \geqslant q_i a_{i-1}$ при $i = 2, \ldots, t$, поэтому

$$s = a_0 \geqslant \prod_{i=2}^{t} q_i.$$

Отсюда мы делаем вывод, что $B < 2^t s$. Из теоремы 2.1.3 мы получаем, что $t < \ln s / \ln((1+\sqrt{5})/2)$, следовательно, $2^t < s^{3/2}$. Отсюда следует утверждение теоремы. □

**Замечание.** Извлечение целочисленного квадратного корня можно выполнить с помощью алгоритма 9.2.11. Если $s < n^{1/3}$, то алгоритм 4.2.11 тоже работает, но количество шагов с извлечением квадратного корня составляет $O(n^{1/3} s^{-1} \ln n)$.

Заметим, что если число $F$ в теореме 4.2.10 таково, что число $F/n^{1/3}$ не слишком мало, то мы можем использовать эту теорему и алгоритм 4.2.11 в качестве быстрого теста на простоту. В общем случае мы можем использовать алгоритм 4.2.11 в качестве теста на простоту, если мы знаем, что каждый простой делитель числа $n$ сравним с $r_i \pmod{s}$ при некотором значении $i \in [1, k]$, причем НОД$(r_i, s) = 1$, $0 < r_i < s$ и $s \geqslant n^{1/3}$. Тогда, $k$ раз используя алгоритм 4.2.11, мы либо найдем нетривиальный делитель числа $n$, либо докажем, что число $n$ является простым. Однако, если $s \geqslant \sqrt{n}$, то нам не надо использовать алгоритм 4.2.11. Действительно, если ни одно из целых чисел $r_i$ не является делителем числа $n$, то любой простой делитель числа $n$ превосходит $\sqrt{n}$, следовательно, число $n$ является простым.

Можно также использовать результат, полученный в статье [Coppersmith 1997] (и [Coppersmith et al. 2000]), чтобы улучшить алгоритм 4.2.11 и найти все делители числа $n$, сравнимые с $r \pmod{s}$, причем числа $r, s$ взаимно просты и $s > n^{1/4+\varepsilon}$. В статье Копперсмита используется быстрый метод нахождения приведенного базиса решетки, разработанный А. Ленстрой, Х. Ленстрой

и Л. Ловасом. Этот метод нахождения приведенного базиса решетки часто используется на практике, и вполне возможно, что алгоритм Копперсмита также можно использовать на практике. В самом деле, Ховгрэйв-Грэхэм сообщил нам, что алгоритм имеет практическое значение для модулей $s > n^{0.29}$. Теоретически этот алгоритм является детерминированным и полиномиальным, но количество арифметических операций зависит от выбора числа $\varepsilon$; чем меньше число $\varepsilon$, тем больше арифметических операций требует алгоритм. Интересное доказательство простоты было проделано в конце 2004 г. при помощи этого гибридного метода: Ренце сообщает, что 37511-е число Фибоначчи, состоящее из 7839 цифр, является простым. Относительно простых чисел Фибоначчи также см. упр. 4.37.

Остается открытым вопрос, можно ли придумать эффективный алгоритм, который находит делители числа $n$, сравнимые с $r \pmod s$, где модуль $s$ порядка $n^{1/4}$ или меньше.

Остается открытым также следующий вопрос. Пусть число $D(n,s,r)$ обозначает количество делителей числа $n$, сравнимых с $r \pmod s$. Нам дано число $\alpha > 0$. Интересен вопрос, является ли число $D(n,s,r)$ ограниченным по всем тройкам чисел $n, s, r$, таким, что $\text{НОД}(r,s) = 1$ и $s \geqslant n^\alpha$. Известно, что это так для всех $\alpha > 1/4$, но при $\alpha = 1/4$ вопрос остается открытым (см. [Lenstra 1984]).

## 4.3. Проверка чисел на простоту при помощи конечного поля

Этот раздел носит в основном теоретический характер и не направлен на получение практического теста на простоту числа. Описанный здесь алгоритм имеет хорошую оценку числа арифметических операций. Но существуют более сложные алгоритмы, которые на практике работают быстрее. Некоторые из них мы рассмотрим позже в этой главе.

Предыдущие разделы и, в частности, теорема 4.2.10 и алгоритм 4.2.11 показывают, что если мы знаем полностью разложенный на множители делитель $F$ числа $n^2 - 1$, такой, что $F \geqslant n^{1/3}$, то мы можем определить, является число $n$ простым или составным. Для себя заметим: если $F_1 = \text{НОД}(F, n-1)$, и $F_2 = \text{НОД}(F, n+1)$, то $\text{НОК}(F_1, F_2) \geqslant \frac{1}{2}F$, поэтому число «$F$» в теореме 4.2.10 не меньше, чем $\frac{1}{2}n^{1/3}$. В этом разделе мы обсудим метод Ленстры [Lenstra 1985], работающий в случае, если мы знаем полностью разложенный на множители делитель $F$ числа $n^I - 1$ при некоторой натуральной степени $I$. Этот метод эффективен, если число $F \geqslant n^{1/3}$, и степень $I$ не очень большая.

Прежде чем обсуждать алгоритм, дадим некоторые используемые нами определения из алгебры. Пусть $n > 1$ — натуральное число, рассмотрим кольцо многочленов $\mathbf{Z}_n[x]$ относительно переменной $x$ с коэффициентами, являющимися классами вычетов по модулю $n$. Идеалом кольца $\mathbf{Z}_n[x]$ называется непустое подмножество, замкнутое относительно сложения и умножения на все элементы кольца $\mathbf{Z}_n[x]$. Например, пусть $f, g \in \mathbf{Z}_n[x]$, множество всех много-

членов вида $af$, где многочлен $a \in \mathbf{Z}_n[x]$, является идеалом. Множество всех многочленов вида $af+bg$, где многочлены $a, b \in \mathbf{Z}_n[x]$, также является идеалом. В первом примере мы имеем *главный идеал* (с порождающим многочленом $f$). Идеал во втором примере может как быть главным, так и не быть главным. Пусть, например, $n = 15, f(x) = 3x+1, g(x) = x^2+4x$. Тогда идеал, порожденный многочленами $f$ и $g$, совпадает со всем кольцом $\mathbf{Z}_{15}[x]$ и является главным с порождающим многочленом 1. (Чтобы показать, что 1 лежит в этом идеале, заметим, что $f^2 - 9g = 1$.)

**Определение 4.3.1.** Будем говорить, что многочлены $f, g \in \mathbf{Z}_n[x]$ *взаимно просты*, если порожденный ими идеал совпадает со всем кольцом $\mathbf{Z}_n[x]$. Таким образом, существуют многочлены $a, b \in \mathbf{Z}_n[x]$, такие, что $af + bg = 1$.

Нетрудно показать, что любой идеал кольца $\mathbf{Z}_n[x]$ является главным тогда и только тогда, когда число $n$ является простым (см. упр. 4.19). Алгоритм, приведенный ниже, является по сути всего лишь улучшенной версией алгоритма Евклида (алгоритм 2.2.1). Он либо находит нормированный порождающий многочлен для главного идеала, порожденного двумя многочленами $f, g \in \mathbf{Z}_n[x]$, либо находит нетривиальное разложение числа $n$ на множители. Если главный идеал, порожденный многочленом $h \in \mathbf{Z}_n[x]$ (многочлен $h$ — нормированный), совпадает с идеалом, порожденным многочленами $f$ и $g$, то мы будем писать $h = \text{НОД}(f, g)$. Таким образом, многочлены $f, g$ взаимно просты в кольце $\mathbf{Z}_n[x]$ тогда и только тогда, когда $\text{НОД}(f, g) = 1$.

**Алгоритм 4.3.2** (нахождение порождающего многочлена). Нам даны натуральное число $n > 1$ и многочлены $f, g \in \mathbf{Z}_n[x]$, причем многочлен $g$ — нормированный. Этот алгоритм находит либо нетривиальное разложение числа $n$ на множители, либо нормированный многочлен $h \in \mathbf{Z}_n[x]$, такой, что $h = \text{НОД}(f, g)$. Т. е. идеал, порожденный многочленами $f$ и $g$, совпадает с идеалом, порожденным многочленом $h$. Мы предполагаем, что либо $f = 0$, либо $\deg f \leqslant \deg g$.

1. [Проверка многочлена на равенство нулю]
   if($f == 0$) return $g$;
2. [Шаг алгоритма Евклида]
   Пусть число $c$ равно старшему коэффициенту многочлена $f$;
   Попробуем найти $c^* \equiv c^{-1} \pmod n$ по алгоритму 2.1.4. Если этот алгоритм дает нам нетривиальное разложение числа $n$, выдаем в качестве ответа это разложение;
   $f = c^* f$; // Умножение коэффициентов по модулю $n$; // многочлен $f$ нормирован.
   $r = g \bmod f$; // Деление с остатком возможно, // так как многочлен $f$ нормирован.
   $(f, g) = (r, f)$;
   goto [Проверка многочлена на равенство нулю];

Следующая теорема является основой теста на простоту при помощи конечного поля.

**Теорема 4.3.3** (Ленстра). *Пусть $n > 1, I, F$ — натуральные числа, и $F | n^I - 1$. Предположим, что многочлены $f, g \in \mathbf{Z}_n[x]$ удовлетворяют условиям:*

*(1) многочлен* $g^{n^I-1}-1$ *кратен многочлену* $f$ *в кольце* $\mathbf{Z}_n[x]$,

*(2) многочлены* $g^{(n^I-1)/q}-1$ *и* $f$ *взаимно просты в кольце* $\mathbf{Z}_n[x]$ *для всех простых чисел* $q|F$,

*(3) каждый из* $I$ *элементарных симметрических многочленов относительно* $g, g^n, \ldots, g^{n^{I-1}}$ *сравним* $(\mathrm{mod}\ f)$ *с элементом из* $\mathbf{Z}_n$.

*Тогда каждый простой делитель* $p$ *числа* $n$ *сравним с* $n^j$ $(\mathrm{mod}\ F)$ *при некотором* $j \in [0, I-1]$.

Отметим, что если условия теоремы 4.3.3 выполнены, и число $n$ не имеет делителей в классах вычетов $n^j$ $(\mathrm{mod}\ F)$ при $j = 0, 1, \ldots, I-1$, то число $n$ является простым. Эта идея лежит в основе теста Ленстры на простоту в конечном поле, который мы опишем после доказательства теоремы.

ДОКАЗАТЕЛЬСТВО. Пусть число $p$ является простым делителем числа $n$. Рассмотрим многочлен $f$ как элемент кольца $\mathbf{Z}_p[x]$; пусть многочлен $f_1 \in \mathbf{Z}_p[x]$ — неприводимый делитель, так что поле $\mathbf{Z}_p[x]/(f_1) = K$ является конечномерным расширением поля $\mathbf{Z}_p$. Обозначим через $\bar{g}$ образ многочлена $g$ в поле $K$. Из условий (1) и (2) следует, что $\bar{g}^{n^I-1} = 1$ и $\bar{g}^{(n^I-1)/q} \neq 1$ для всех простых чисел $q|F$. Поэтому порядок многочлена $\bar{g}$ в группе $K^*$ (мультипликативная группа поля $K$) кратен числу $F$. Из условия (3) следует, что многочлен $h(T) = (T-\bar{g})(T-\bar{g}^n)\ldots(T-\bar{g}^{n^{I-1}}) \in K[T]$ на самом деле лежит в $\mathbf{Z}_p[T]$. Для любого многочлена из $\mathbf{Z}_p[T]$ имеем, что если элемент $\alpha$ является его корнем, то элемент $\alpha^p$ также является его корнем. Следовательно, $h(\bar{g}^p) = 0$. Но у нас есть разложение многочлена $h(T)$ на множители, и мы видим, что его корни — это элементы $\bar{g}, \bar{g}^n, \ldots, \bar{g}^{n^{I-1}}$. Поэтому при некотором $j = 0, 1, \ldots, I-1$ должно выполняться равенство $\bar{g}^p \equiv \bar{g}^{n^j}$. Так как порядок элемента $\bar{g}$ кратен числу $F$, мы имеем, что $p \equiv n^j$ $(\mathrm{mod}\ F)$. Доказательство закончено. □

Вполне естественно, что сразу возникает несколько вопросов. Если число $n$ является простым, найдутся ли многочлены $f, g$, описанные в теореме 4.3.3? Если многочлены $f, g$ существуют, легко ли их найти? Можно ли быстро проверить выполнение условий (1), (2), (3) теоремы 4.3.3?

Ответ на первый вопрос не составляет труда. Если число $n$ является простым, то любой неприводимый многочлен $f \in \mathbf{Z}_n[x]$ степени $\deg f = I$ и не кратный $f$ многочлен $g \in \mathbf{Z}_n[x]$ удовлетворяют условиям (1) и (3). Действительно, если $f$ — неприводимый многочлен степени $I$, то поле $K = \mathbf{Z}_n[x]/(f)$ будет конечным полем порядка $n^I$. Поэтому утверждение (1) представляет собой просто теорему Лагранжа (если любой элемент группы возвести в степень, равную порядку группы, то мы получим единичный элемент) для мультипликативной группы $K^*$. Чтобы доказать выполнение условия (3), вспомним, что группа Галуа поля $K$ порождается автоморфизмом Фробениуса — возведением в степень $n$. Поэтому группа Галуа состоит из $I$ функций из $K$ в $K$. Причем $j$-я функция элемент $\alpha \in K$ переводит в элемент $\alpha^{n^j}$ при $j = 0, 1, \ldots, I-1$. Каждая из этих функций оставляет на месте выражение, симметричное относительно $g, g^n, \ldots, g^{n^{I-1}}$, так что такое выражение должно лежать в неподвижном поле $\mathbf{Z}_n$. Это и есть утверждение (3).

Не любой многочлен $g$, такой, что $g \not\equiv 0 \pmod f$, удовлетворяет условию (2). Но группа $K^*$ является циклической, и любой порождающий элемент удовлетворяет условию (2). Однако существует довольно много порождающих элементов, и поиск хотя бы одного не должен занять много времени. В частности, если многочлен $g$ выбирается случайно среди ненулевых многочленов из $\mathbf{Z}_n[x]$ степени, меньшей, чем $I$, то вероятность того, что этот многочлен удовлетворяет условию (2), не меньше $\varphi(n^I-1)/(n^I-1)$ (при условии, что число $n$ является простым, и многочлен $f$ является неприводимым многочленом степени $I$). Поэтому для нахождения подходящего многочлена $g$ надо сделать $O(\ln\ln(n^I))$ случайных выборов.

Остается вопрос выбора многочлена $f$. На самом деле в группе $\mathbf{Z}_n[x]$ не просто существуют неприводимые многочлены степени $I$, а их даже довольно много (см. равенство (2.5)). Вероятность того, что случайно выбранный многочлен степени $I$ будет неприводимым, равна примерно $1/I$. При помощи алгоритма 2.2.9 мы можем быстро проверить, является ли данный многочлен неприводимым.

Теперь мы воплотим все вышесказанное в следующем алгоритме.

**Алгоритм 4.3.4** (тест на простоту при помощи конечного поля). Нам даны натуральные числа $n, I, F$, причем $F|n^I-1$, $F \geqslant n^{1/3}$. Нам дано также полное разложение на простые сомножители числа $F$. Этот вероятностный алгоритм определяет, является число $n$ простым или составным.

1. [Нахождение неприводимого многочлена степени $I$]
   При помощи алгоритма 2.2.9 и алгоритма 4.3.2 (для нахождения НОД многочленов) пробуем найти произвольный нормированный многочлен $f$ из $\mathbf{Z}_n[x]$ степени $I$, который был бы неприводим, если бы число $n$ было простым. Будем продолжать проверять произвольные многочлены, пока алгоритм 2.2.9 не ответит ДА, или пока его шаг нахождения НОД не найдет нетривиальное разложение числа $n$ на множители. В последнем случае возвратить ответ «число $n$ является составным»;
   // Многочлен $f$ неприводим, если число $n$ — простое.
2. [Нахождение порождающего элемента]
   Выбираем случайным образом нормированный многочлен $g \in \mathbf{Z}_n[x]$ степени $\deg g < I$;
   if($1 \neq g^{n^I-1} \bmod f$) return «число $n$ является составным»;
   for(по простым $q|F$) {
   Пытаемся вычислить $\text{НОД}\,(g^{(n^I-1)/q}-1, f)$ при помощи алгоритма 4.3.2. Если при этом будет найдено нетривиальное разложение на множители числа $n$, то возвратить ответ «число $n$ является составным»;
   if($\text{НОД}\,(g^{(n^I-1)/q}-1, f) \neq 1$) goto [Нахождение порождающего элемента];
   }
3. [Проверка симметрических выражений]

Рассмотрим многочлен $(T-g)(T-g^n)\dots(T-g^{n^{I-1}}) = T^I + c_{I-1}T^{I-1} + \dots + c_0$ из $\mathbf{Z}_n[x,T]/(f(x))$;
// Коэффициенты $c_j$ лежат в $\mathbf{Z}_n[x]$ и взяты по модулю многочлена $f$.
for($0 \leqslant j < I$) if($\deg c_j > 0$) return «число $n$ является составным»;

4. [Нахождение делителя]
for($1 \leqslant j < I$) {
Ищем делители числа $n$, которые $\equiv n^j \pmod F$ (при помощи алгоритма 4.2.11). Если найден нетривиальный делитель числа $n$, то возвратить ответ «число $n$ является составным»;
}
return «число $n$ является простым»;

Если число $n$ простое, то алгоритму 4.3.4 для доказательства простоты необходимо будет сделать $O(I^c + \ln^c n)$ ($c$ — некоторая положительная константа) арифметических операций с целыми числами размера $n$. (Мы не оцениваем время работы алгоритма для составных чисел $n$.)

Остается вопрос, как по данному простому числу $n$ подобрать числа $I, F$. Эти числа должны удовлетворять следующим условиям: число $F$ должно быть достаточно большим, а именно, $F \geqslant n^{1/3}$; мы должны знать разложение на простые множители числа $F$; $F | n^I - 1$; число $I$ не должно быть очень большим (так как иначе время работы алгоритма сильно увеличивается). Для некоторых чисел $n$ мы можем положить $I = 1$ или 2; алгоритмы с таким выбором числа $I$ мы обсуждали в предыдущих разделах этой главы. Нас интересует ответ на вопрос, как в общем случае найти числа $I, F$, удовлетворяющие нужным условиям.

Интересно, что мы без особого труда можем отобрать некоторые малые простые числа, входящие в $n^I - 1$. Например, пусть $I = 12$. Тогда, если число $n$ взаимно просто с $65520 = 2^4 \cdot 3^2 \cdot 5 \cdot 7 \cdot 13$, то число $n^I - 1$ делится на 65520. Из теоремы Эйлера (см. соотношение (2.2)) следует такой факт: если число $q$ — степень простого числа, число $q$ взаимно просто с числом $n$, $\varphi(q) | I$, то $q | n^I - 1$. (Если число $q$ является степенью числа 2, большей, чем 4, то достаточно будет того, что $\frac{1}{2}\varphi(q) | I$.) На самом деле, несколько небольших простых сомножителей вместе могут дать нам один достаточно большой делитель числа $n^I - 1$. Например, пусть $I = 7! = 5040$. Тогда, если число $n$ не делится ни на одно простое число до 2521, то число $n^{5040} - 1$ делится на

$$153219867888544432846626127356636113800104312257712000 =$$
$$2^6 \cdot 3^3 \cdot 5^2 \cdot 7^2 \cdot 11 \cdot 13 \cdot 17 \cdot 19 \cdot 29 \cdot 31 \cdot 37 \cdot 41 \cdot 43 \cdot 61 \cdot 71 \cdot 73 \cdot$$
$$113 \cdot 127 \cdot 181 \cdot 211 \cdot 241 \cdot 281 \cdot 337 \cdot 421 \cdot 631 \cdot 1009 \cdot 2521.$$

Поэтому в алгоритме 4.3.4 для простых чисел $n$, меньших $3.5 \cdot 10^{156}$ (и больших 2521), можно положить $I = 5040$.

Из примера с числом $I = 5040$ можно предположить, что в общем случае выбор числа $I$ для получения достаточного большого делителя числа $n^I - 1$ представляет собой небольшую функцию от $n$. На самом деле мы имеем следующую теорему (см. [Adleman et al. 1983]). Для доказательства теоремы требуются сложные инструменты аналитической теории чисел.

**Теорема 4.3.5.** *Пусть функция $I(x)$ обозначает наименьшее натуральное бесквадратное число $I$, такое, что произведение простых чисел $p$, удовлетворяющих условию $p-1|I$, превосходит число $x$. Тогда существует число $c$, такое, что $I(x) < (\ln x)^{c \ln\ln\ln x}$ для любого числа $x > 16$.*

Мы написали условие $x > 16$ для того, чтобы тройной логарифм был положителен. Требование бесквадратности числа $I$ не является необходимым для выполнения теоремы. Но мы включили его, потому что оно понадобится для алгоритма следующего раздела.

**Следствие 4.3.6.** *Существует положительное число $c'$, такое, что алгоритм* 4.3.4 *требует не более $(\ln n)^{c' \ln\ln\ln n}$ арифметических операций для доказательства простоты числа $n$.*

Так как тройной логарифм растет очень медленно, количество операций можно оценить почти как $\ln^{O(1)} n$, т. е. алгоритм является почти полиномиальным.

## 4.4. Суммы Гаусса и Якоби

В статье Адлемана, Померанса и Румели [Adleman et al. 1983] был опубликован тест на простоту числа $n$, требующий $(\ln n)^{c \ln\ln\ln n}$ арифметических операций ($c$ — некоторая положительная константа). Обоснование этого теста опирается на теорему 4.3.5 и арифметические свойства сумм Якоби. В работе были представлены два варианта этого теста: вероятностный и детерминированный. Вероятностный тест более прост и практичен в использовании, чем детерминированный. Как и некоторые другие вероятностные тесты, с которыми мы имели дело в этой главе, описанный этими авторами APR-тест выдает результат «число — простое» только в том случае, когда оно действительно является простым. Единственное, что в этом тесте не может быть определено точно, так это время его работы.

Вскоре после этого были сделаны улучшения этого теста двух типов. Улучшения первого типа дали нам более практичную версию теста, второго — менее практичную, но более простую. В следующем разделе мы будем обсуждать одно из улучшений второго типа, а именно детерминированный тест на простоту Х. Ленстры [Lenstra 1981], основанный на суммах Гаусса.

### 4.4.1. Тест, основанный на суммах Гаусса

В разд. 2.3.1 мы вводили понятие сумм Гаусса для квадратичных характеров. Здесь мы будем рассматривать суммы Гаусса для произвольных характеров Дирихле. Если у нас есть простое число $q$ с первообразным корнем $g$ и комплексное число $\zeta$, такое, что $\zeta^{q-1} = 1$, то мы можем построить характер $\chi$ по модулю $q$ по следующей формуле: $\chi(g^m) = \zeta^m$ для любого целого числа $m$ ($\chi(u) = 0$, если число $u$ кратно числу $q$). (См. разд. 1.4.3 для получения подробной информации о характерах.) Мы также можем построить сумму Гаусса

$\tau(\chi)$. Обозначим $\zeta_n = e^{2\pi\imath/n}$ (примитивный корень $n$-й степени из 1), определим

$$\tau(\chi) = \sum_{m=1}^{q-1} \chi(m)\zeta_q^m = \sum_{k=1}^{q-1} \chi(g^k)\zeta_q^{g^k} = \sum_{k=1}^{q-1} \zeta^k \zeta_q^{g^k}.$$

Будучи характером по модулю $q$, $\chi$ имеет порядок, делящий $q-1$. Пусть $p$ является простым делителем числа $q-1$, и мы хотим, чтобы порядок $\chi$ был равен в точности $p$. Мы можем явным образом построить такой характер $\chi_{p,q}$. Пусть $g = g_q$ является наименьшим положительным первообразным корнем для числа $q$, и пусть $\chi_{p,q}(g_q^k) = \zeta_p^k$ для каждого целого числа $k$. Из предыдущего абзаца следует, что мы тем самым определили характер по модулю $q$, поскольку $\zeta_p^{q-1} = 1$. И поскольку $\chi_{p,q}(m)^p = 1$ для каждого ненулевого вычета $m \bmod q$, и $\chi_{p,q}(g_q) \neq 1$, мы получаем, что $\chi_{p,q}$ имеет порядок $p$. Пусть

$$G(p,q) = \tau(\chi_{p,q}) = \sum_{m=1}^{q-1} \chi_{p,q}(m)\zeta_q^m = \sum_{k=1}^{q-1} \zeta_p^k \zeta_q^{g_q^k} = \sum_{k=1}^{q-1} \zeta_p^{k \bmod p} \zeta_q^{g_q^k \bmod q}.$$

(Это определение в случае $p = 2$ эквивалентно определению 2.3.6 и используется в упр. 4.20.)

Нас интересуют арифметические свойства сумм Гаусса $G(p,q)$, хотя, казалось бы, какие могут быть арифметические свойства у суммы нескольких комплексных чисел! Сумма Гаусса $G(p,q)$ является элементом кольца $\mathbf{Z}[\zeta_p, \zeta_q]$. Элементы этого кольца можно однозначно представить в виде двойной суммы $\sum_{j=0}^{p-2}\sum_{k=0}^{q-2} a_{j,k}\zeta_p^j\zeta_q^k$, где коэффициенты $a_{j,k} \in \mathbf{Z}$. Теперь мы можем ввести понятие сравнимости по модулю $n$ в кольце $\mathbf{Z}[\zeta_p, \zeta_q]$, а именно, два элемента кольца сравнимы по модулю $n$ тогда и только тогда, когда соответствующие им целые коэффициенты $a_{j,k}$ сравнимы по модулю $n$. Также заметим, что если число $\alpha$ принадлежит кольцу $\mathbf{Z}[\zeta_p, \zeta_q]$, то и его комплексное сопряжение $\overline{\alpha}$ тоже принадлежит этому кольцу.

При реальных вычислениях в кольце очень важно уметь работать с $\zeta_p, \zeta_q$ в их символьном представлении. Как и в случае последовательностей Люка, когда мы работаем с символьным представлением корней квадратичных многочленов, мы рассматриваем $\zeta_p, \zeta_q$ как символы, скажем, $x, y$, которые подчиняются законам

$$x^{p-1} + x^{p-2} + \ldots + 1 = 0, \quad y^{q-1} + y^{q-2} + \ldots + 1 = 0.$$

При этом мы можем обойтись без использования комплексных чисел с плавающей точкой.

Мы начнем изучение сумм Гаусса со следующей леммы:

**Лемма 4.4.1.** *Пусть $p, q$ — простые числа, и $p \mid q-1$. Тогда $G(p,q)\overline{G(p,q)} = q$.*

ДОКАЗАТЕЛЬСТВО. Пусть $\chi = \chi_{p,q}$. Мы имеем следующее равенство:

$$G(p,q)\overline{G(p,q)} = \sum_{m_1=1}^{q-1}\sum_{m_2=1}^{q-1} \chi(m_1)\overline{\chi(m_2)}\zeta_q^{m_1-m_2}.$$

Обозначим через $m_2^{-1}$ обратный к $m_2$ (по умножению) элемент по модулю $q$,

т. е. $\overline{\chi(m_2)} = \chi(m_2^{-1})$. Заметим, что если $m_1 m_2^{-1} \equiv a \pmod q$, то $\chi(m_1)\overline{\chi(m_2)} = \chi(a)$ и $m_1 - m_2 \equiv (a-1)m_2 \pmod q$. Следовательно,

$$G(p,q)\overline{G(p,q)} = \sum_{a=1}^{q-1} \chi(a) \sum_{m=1}^{q-1} \zeta_q^{(a-1)m}.$$

Внутренняя сумма равна $q-1$, если $a=1$, и равна $-1$, если $a>1$. Следовательно,

$$G(p,q)\overline{G(p,q)} = q - 1 - \sum_{a=2}^{q-1} \chi(a) = q - \sum_{a=1}^{q-1} \chi(a).$$

И наконец, по равенству (1.28) последняя сумма равна 0, что завершает доказательство леммы. □

Следующий результат показывает возможность применения сумм Гаусса для проверки чисел на простоту. Этот результат можно рассматривать как аналог малой теоремы Ферма.

**Лемма 4.4.2.** *Пусть $p, q, n$ — простые числа, $p \mid q-1$, и $\text{НОД}(pq, n) = 1$. Тогда*

$$G(p,q)^{n^{p-1}-1} \equiv \chi_{p,q}(n) \pmod n.$$

Доказательство. Пусть $\chi = \chi_{p,q}$. Так как число $n$ является простым, то из обобщенной формулы бинома Ньютона следует, что

$$G(p,q)^{n^{p-1}} = \left(\sum_{m=1}^{q-1} \chi(m)\zeta_q^m\right)^{n^{p-1}} \equiv \sum_{m=1}^{q-1} \chi(m)^{n^{p-1}} \zeta_q^{mn^{p-1}} \pmod n.$$

По малой теореме Ферма имеем, что $n^{p-1} \equiv 1 \pmod p$, поэтому $\chi(m)^{n^{p-1}} = \chi(m)$. Пусть $n^{-1}$ обозначает обратный (по умножению) к $n$ элемент по модулю $q$. Имеем

$$\begin{aligned}\sum_{m=1}^{q-1} \chi(m)^{n^{p-1}} \zeta_q^{mn^{p-1}} &= \sum_{m=1}^{q-1} \chi(m)\zeta_q^{mn^{p-1}} = \sum_{m=1}^{q-1} \chi(n^{-(p-1)})\chi(mn^{p-1})\zeta_q^{mn^{p-1}} \\ &= \chi(n) \sum_{m=1}^{q-1} \chi(mn^{p-1})\zeta_q^{mn^{p-1}} = \chi(n)G(p,q),\end{aligned}$$

где предпоследнее равенство использует тот факт, что $\chi(n^p) = \chi(n)^p = 1$. А последнее равенство следует из того, что числа $mn^{p-1}$ пробегают те же классы вычетов (mod $q$), что и числа $m$. Следовательно,

$$G(p,q)^{n^{p-1}} \equiv \chi(n)G(p,q) \pmod n.$$

Пусть $q^{-1}$ обозначает обратный (по умножению) к $q$ элемент по модулю $n$. Умножим обе части последнего сравнения на $q^{-1}\overline{G(p,q)}$. Лемма 4.4.1 дает нам требуемый результат. □

Следующая лемма позволяет в некоторых случаях заменить сравнение на равенство.

**Лемма 4.4.3.** *Пусть $m, n$ — натуральные числа, число $m$ не делится на число $n$, и $\zeta_m^j \equiv \zeta_m^k \pmod n$. Тогда $\zeta_m^j = \zeta_m^k$.*

Доказательство. Умножив обе части сравнения на $\zeta_m^{-k}$, мы получим, что $\zeta_m^j \equiv 1 \pmod n$. Заметим, что $\prod_{l=1}^{m-1}(x - \zeta_m^l) = (x^m - 1)/(x - 1)$, поэтому $\prod_{l=1}^{m-1}(1 - \zeta_m^l) = m$. Следовательно, в последнем произведении ни один из множителей не равен нулю по модулю $n$, отсюда следует результат леммы. □

**Определение 4.4.4.** *Пусть $p, q$ — различные простые. Если $\alpha \in \mathbf{Z}[\zeta_p, \zeta_q] \setminus \{0\}$, где $\alpha = \sum_{i=0}^{p-2}\sum_{k=0}^{q-2} a_{ik}\zeta_p^i\zeta_q^k$, то обозначим через $c(\alpha)$ наибольший общий делитель коэффициентов $a_{ik}$. Кроме того, пусть $c(0) = 0$.*

Теперь мы можем описать детерминированный тест на простоту, основанный на суммах Гаусса.

**Алгоритм 4.4.5** (тест, основанный на суммах Гаусса)**.** Нам дано целое число $n > 1$. Данный детерминированный алгоритм распознает, является ли число $n$ простым, и отвечает «число $n$ является простым» или «число $n$ является составным» в соответствующих случаях.

1. [Инициализация]
   $I = -2$;
2. [Приготовления]
   $I = I + 4$;
   Находим полное разложение на простые множители числа $I$ при помощи пробного деления, и если $I$ — не бесквадратное, перейти к шагу [Приготовления];
   Положим число $F$ равным произведению простых чисел $q$, таких, что $q - 1 | I$, но если $F^2 \leqslant n$, перейти к шагу [Приготовления];
   // Мы имеем $I, F$ — бесквадратные, и $F > \sqrt{n}$.
   Если число $n$ является простым делителем числа $IF$, возвратить ответ «число $n$ является простым»;
   Если НОД $(n, IF) > 1$, возвратить ответ «число $n$ является составным»;
   for(по простым числам $q | F$) находим наименьший первообразный корень $g_q$ для числа $q$;
3. [Вычисление для проверки свойства вероятного простого числа]
   for(по простым числам $p | I$) разложим на множители число $n^{p-1} - 1 = p^{s_p}u_p$, где простое число $p$ не делит число $u_p$;
   for(по простым числам $p, q$, таким, что $p | I,\ q | F,\ p | q - 1$) {
   Находим первое натуральное число $w(p, q) \leqslant s_p$, такое, что
   $$G(p, q)^{p^{w(p,q)}u_p} \equiv \zeta_p^j \pmod n \text{ для некоторого целого числа } j,$$
   если такого числа $w(p, q)$ нет, то возвратить ответ «число $n$ является составным»;
   } // Вычислить в символьном виде в кольце $\mathbf{Z}[\zeta_p, \zeta_q]$ (см. в тексте).
4. [Поиск наибольшего порядка]
   for(по простым числам $p | I$) положим число $w(p)$ равным максимуму среди чисел $w(p, q)$ по всем простым числам $q | F$, таким, что $p | q - 1$, и

положим число $q_0(p)$ равным наименьшему простому числу $q$, такому, что $w(p) = w(p, q)$;

for(по простым числам $p, q$, таким, что $p|I,\ q|F,\ p|q-1$) находим целое число $l(p,q) \in [0, p-1]$, такое, что $G(p,q)^{p^{w(p)}u_p} \equiv \zeta_p^{l(p,q)} \pmod n$;

5. [Проверка взаимной простоты]

for(по простым $p$, таким, что $p|I$) {

$H = G(p, q_0(p))^{p^{w(p)-1}u_p} \bmod n$;

for($0 \leqslant j \leqslant p-1$) {

if(НОД $\left(n, c(H - \zeta_p^j)\right) > 1$) return «число $n$ является составным»;

} // Обозначение из определения 4.4.4.

}

6. [Поиск делителя]

$l(2) = 0$;

for(по простым числам $q|F$) используем китайскую теорему об остатках (см. теорему 2.1.6) для построения целого числа $l(q)$, такого, что

$$l(q) \equiv l(p,q) \pmod p \text{ для всех простых чисел } p|q-1;$$

Используем китайскую теорему об остатках для нахождения целого числа $l$, такого, что

$$l \equiv g_q^{l(q)} \pmod q \text{ для всех простых чисел } q|F;$$

for($1 \leqslant j < I$) если число $l^j \bmod F$ является нетривиальным делителем числа $n$, то возвратить ответ «число $n$ является составным»;

return «число $n$ является простым»;

**Замечание.** Мы можем опустить условие $F \geqslant \sqrt{n}$ и использовать в шаге поиска делителя алгоритм 4.2.11. Этот алгоритм останется достаточно быстрым, если $F \geqslant n^{1/3}$.

**Теорема 4.4.6.** *Алгоритм* 4.4.5 *корректно распознает простые и составные числа. Алгоритм требует не более* $(\ln n)^{c \ln\ln\ln n}$ *арифметических операций, где* $c$ *— некоторая положительная константа.*

Доказательство. Сначала заметим, что утверждения «число $n$ является простым» и «число $n$ является составным» в шаге [Приготовления] являются корректными. Корректность утверждения о непростоте числа $n$ в шаге [Вычисление для проверки свойства вероятного простого числа] следует из леммы 4.4.2. Поэтому, если НОД в шаге [Проверка взаимной простоты] не равен 1, то мы получаем делитель числа $n$, следовательно, утверждение о непростоте в этом шаге тоже корректно. Очевидно, что утверждение о непростоте в шаге [Поиск делителя] тоже корректно. Таким образом, нам осталось показать, что все составные числа, которые прошли предыдущие шаги алгоритма, будут разложены на множители в шаге [Поиск делителя].

Предположим, что число $n$ является составным с наименьшим простым делителем $r$. Предположим также, что число $n$ прошло первые четыре шага алгоритма. Сначала покажем, что

$$p^{w(p)} | r^{p-1} - 1 \text{ для всех простых чисел } p|I. \tag{4.21}$$

Этот факт очевиден при $w(p) = 1$, поэтому предположим, что $w(p) \geqslant 2$. Пусть некоторое число $l(p,q) \neq 0$. Тогда по лемме 4.4.3 имеем

$$G(p,q)^{p^{w(p)}u_p} \equiv \zeta_p^{l(p,q)} \not\equiv 1 \pmod{n}.$$

Это сравнение также выполнено по модулю $r$. Пусть число $h$ равно порядку (по умножению) группы $G(p,q)$ по модулю $r$, тогда $p^{w(p)+1}|h$. Но из леммы 4.4.2 следует, что $h|p(r^{p-1}-1)$, поэтому $p^{w(p)}|r^{p-1}-1$, что и требовалось. Предположим теперь, что все числа $l(p,q) = 0$. Тогда из вычислений в шаге [Проверка взаимной простоты] мы имеем

$$G(p,q_0)^{p^{w(p)}u_p} \equiv 1 \pmod{r},\ G(p,q_0)^{p^{w(p)-1}u_p} \not\equiv \zeta_p^j \pmod{r}$$

для всех чисел $j$. Пусть снова число $h$ равно порядку (по умножению) группы $G(p,q_0)$ по модулю $r$, мы имеем $p^{w(p)}|h$. Также, из сравнения $G(p,q_0)^m \equiv \zeta_p^j \pmod{r}$ для некоторых целых чисел $m, j$ следует, что $\zeta_p^j = 1$. Из леммы 4.4.2 следует, что $G(p,q_0)^{r^{p-1}-1} \equiv 1 \pmod{r}$, поэтому $h|r^{p-1}-1$ и $p^{w(p)}|h$. Мы полностью доказали утверждение (4.21).

Для всех простых чисел $p|I$ из утверждения (4.21) следует существование целых чисел $a_p, b_p$, таких, что

$$\frac{r^{p-1}-1}{p^{w(p)}u_p} = \frac{a_p}{b_p}, \quad b_p \equiv 1 \pmod{p}. \tag{4.22}$$

Пусть число $a$ таково, что $a \equiv a_p \pmod{p}$ для всех простых чисел $p|I$. Теперь мы покажем, что

$$r \equiv l^a \pmod{F}. \tag{4.23}$$

Отсюда будет следовать наше утверждение о шаге [Поиск делителя]. Действительно, так как $F > \sqrt{n} \geqslant r$, мы имеем, что число $r$ является наименьшим положительным вычетом числа $l^a \pmod{F}$. Следовательно, делитель $r$ числа $n$ будет найден в шаге [Поиск делителя].

Заметим, что из определения характера $\chi_{p,q}$ и числа $l$ следует, что

$$G(p,q)^{p^{w(p)}u_p} \equiv \zeta_p^{l(p,q)} = \chi_{p,q}(g_q^{l(p,q)}) = \chi_{p,q}(g_q^{l(q)}) = \chi_{p,q}(l) \pmod{r}$$

для любой пары простых чисел $p, q$, такой, что $q|F, p|q-1$. Следовательно, из равенств (4.22) и леммы 4.4.2 мы имеем

$$\begin{aligned} \chi_{p,q}(r) = \chi_{p,q}(r)^{b_p} &\equiv G(p,q)^{(r^{p-1}-1)b_p} = G(p,q)^{p^{w(p)}u_p a_p} \\ &\equiv \chi_{p,q}(l)^{a_p} = \chi_{p,q}(l^a) \pmod{r}, \end{aligned}$$

и по лемме 4.4.3 имеем

$$\chi_{p,q}(r) = \chi_{p,q}(l^a).$$

Для любого простого числа $q|F$ существует целое число $\rho_q$, такое, что $r \equiv g_q^{\rho_q} \pmod{q}$, поэтому $\chi_{p,q}(r) = \zeta_p^{\rho_q}$. Но $\chi_{p,q}(l^a) = \chi_{p,q}\left(g_q^{l(q)a}\right) = \zeta_p^{l(q)a}$, следовательно, для каждой пары простых чисел $p, q$, таких, что $q|F$, $p|q-1$, мы имеем $\rho_q \equiv l(q)a \pmod{p}$. Для данного простого числа $q$ произведение всех простых чисел $p|q-1$ равно в точности $q-1$ (так как $q-1|I$, и число $I$ — бесквадратное).

Следовательно, $\rho_q \equiv l(q)a \pmod{q-1}$, и $r \equiv g_q^{l(q)a} \equiv l^a \pmod{q}$. Так как это выполняется для всех простых чисел $q|F$, и число $F$ является бесквадратным, то отсюда следует сравнение (4.23). Корректность алгоритма 4.4.5 доказана полностью.

Очевидно, что количество арифметических операций ограничено некоторой степенью числа $I$, поэтому утверждение о времени работы алгоритма следует из теоремы 4.3.5. □

Проделав некоторую дополнительную работу, можно распространить тест на простоту, основанный на суммах Гаусса, на случай, когда $I$ не предполагается бесквадратным. Эта дополнительная степень свободы приводит к более быстрому тесту. Кроме того, существуют улучшения, которые используют случайность, и тем самым отказываются от детерминированности этого теста. За информацией о довольно быстром варианте теста на простоту, основанного на суммах Гаусса, можно обратиться к новой статье [Schoof 2004].

### 4.4.2. Тест, основанный на суммах Якоби

Существует множество способов улучшить тест, основанный на суммах Гаусса, и один из главных способов — вообще не использовать суммы Гаусса! Точнее, использовать в оригинальном тесте Адлемана, Померанса и Румели суммы Якоби. Суммы Гаусса $G(p,q)$ лежат в кольце $\mathbf{Z}[\zeta_p, \zeta_q]$. Арифметические операции в этом кольце по модулю $n$ требуют работы с векторами с $(p-1)(q-1)$ координатами, у которых каждая координата является вычетом по модулю $n$. На практике мы можем взять простые числа $p$ достаточно маленькими, например, меньше, чем $\ln n$. Но простые числа $q$ могут быть несколько бóльшими, вплоть до $(\ln n)^{c \ln\ln\ln n}$. Суммы Якоби $J(p,q)$, которые мы собираемся ввести, лежат в гораздо меньшем кольце $\mathbf{Z}[\zeta_p]$, и арифметические операции с ними можно проделывать гораздо быстрее.

Вспомним определение характера $\chi_{p,q}$ в разд. 4.4.1, где $p, q$ — простые числа, такие что $p|q-1$. Мы предположим, что простое число $p$ является нечетным. Положим число $b = b(p)$ равным наименьшему натуральному числу, такому, что $(b+1)^p \not\equiv b^p + 1 \pmod{p^2}$. (Как показано в статье [Crandall et al. 1997], мы можем взять $b = 2$ для любого простого числа $p$ вплоть до $10^{12}$, кроме $p = 1093$ и $p = 3511$ (для этих чисел можно взять $b = 3$). Возможно, что для любого простого числа $p$ можно положить $b(p) = 2$ или 3. Но мы знаем точно, что $b(p) < \ln^2 p$; см. упр. 3.19.)

Теперь мы определим сумму Якоби $J(p,q)$. Сумма Якоби — это

$$J(p,q) = \sum_{m=1}^{q-2} \chi_{p,q}\left(m^b(m-1)\right).$$

Связь сумм Якоби с предполагаемой простотой числа $n$ устанавливается из следующего более общего результата. Предположим, что $n$ является нечетным простым числом, не делящимся на число $p$. Пусть число $f$ равно порядку числа $n$ в мультипликативной группе $\mathbf{Z}_p^*$. Тогда идеал $(n)$ в кольце $\mathbf{Z}[\zeta_p]$ расклады-

вается на $(p-1)/f$ простых идеалов $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_{(p-1)/f}$ с нормой $n^f$. Если элемент $\alpha$ лежит в кольце $\mathbf{Z}[\zeta_p]$, но не лежит в идеале $\mathcal{N}_j$, то существует некоторое целое число $a_j$, такое, что $\alpha^{(n^f-1)/p} \equiv \zeta_p^{a_j} \pmod{\mathcal{N}_j}$. Тест, основанный на суммах Якоби, проверяет это сравнение для числа $\alpha = J(p,q)$ для тех же пар простых чисел $p, q$ $(p > 2)$, которые возникают в ходе теста, основанного на суммах Гаусса. Чтобы применить этот тест, необходимо также найти идеалы $\mathcal{N}_j$. Это можно сделать, разложив многочлен $x^{p-1} + x^{p-2} + \ldots + 1$ по модулю $n$ в произведение $h_1(x)h_2(x)\ldots h_{(p-1)/f}(x)$, где каждый многочлен $h_j(x)$ является неприводимым и имеет степень $f$. Тогда в качестве $\mathcal{N}_j$ мы можем взять идеал, порожденный числом $n$ и элементом $h_j(\zeta_p)$. Можно попробовать произвести эти вычисления и не зная, является ли число $n$ простым, если же это не удается, то мы можем утверждать, что число $n$ является составным.

Для более полного описания теста см. работу [Adleman et al. 1983]. Практическую реализацию теста и другие усовершенствования см. в работе [Bosma and van der Hulst 1990].

## 4.5. Тест на простоту Агравала, Кайала и Саксены (AKS-тест)

В августе 2002 г. М. Агравал, Н. Кайал и Н. Саксена объявили о сенсационном новом результате — детерминированном полиномиальном тесте на простоту. Сейчас он известен как AKS-тест. Тест с такими свойствами существует в предположении расширенной гипотезы Римана (см. алгоритм 3.5.13). Далее, алгоритм 3.5.6 («тест Миллера—Рабина») — это вероятностный алгоритм, который за ожидаемое полиномиальное время доказывает, что поданное на вход составное число является составным. Нам был известен вероятностный алгоритм, который за ожидаемое полиномиальное время доказывает, что поданное на вход простое число является простым; это тест Адлемана—Хуанга, который будет вкратце описан в разд. 7.6. Наконец, как мы только что видели (см. теорему 4.4.6), алгоритм 4.4.5 является строго обоснованным детерминированным тестом на простоту с «почти полиномиальной» оценкой времени работы вида $(\ln n)^{c \ln\ln\ln n}$. Мы говорим «почти полиномиальной», поскольку показатель $\ln\ln\ln n$ растет так медленно, что на практике его можно считать ограниченным. (На эту тему можно пошутить: хоть мы и доказали, что $\ln\ln\ln n$ стремится к бесконечности с ростом $n$, но свидетелями этого мы никогда не были!)

Новый тест произвел сенсацию не только тем, что окончательно восполнил пробел в теории проверки на простоту, когда исследователи уже так близко подошли к решению с разных сторон; он замечателен еще и своей простотой. К тому же двое из авторов, Кайал и Саксена, работали над этой задачей в рамках своего дипломного проекта, получив степени бакалавров всего лишь за три месяца до объявления результата. Вскоре, учитывая поступившие замечания, Агравал, Кайал и Саксена представили еще более простой вариант теста. Оба варианта можно найти в работах [Agrawal et al. 2002], [Agrawal et al. 2004].

В этом разделе будет приведен второй вариант алгоритма Агравала—Кайла—Саксены, а также некоторые последующие разработки. На момент выхода этой книги еще предстояло выяснить, будет ли AKS-тест полезен для доказательства простоты больших чисел. Наибольшие шансы на успех здесь имеет квартичный вариант теста, приведенный в конце раздела.

### 4.5.1. Проверка на простоту с помощью корней из единицы

Если $n$ — простое, то

$$g(x)^n \equiv g(x^n) \pmod{n}$$

для любого многочлена $g(x) \in \mathbf{Z}[x]$. В частности,

$$(x+a)^n \equiv x^n + a \pmod{n} \tag{4.24}$$

для любого $a \in \mathbf{Z}$. Далее, если соотношение (4.24) выполнено хотя бы для одного значения $a$ с условием НОД $(a, n) = 1$, то $n$ должно быть простым(см. 4.25). Другими словами, соотношение (4.24) является необходимым и достаточным критерием простоты. Проблема в том, что нам не известен быстрый способ проверки соотношения (4.24) даже в простейшем случае $a = 1$: число членов в левой части этого сравнения слишком велико.

Если $f(x) \in \mathbf{Z}[x]$ — произвольный многочлен с единичным старшим коэффициентом, то из (4.24) следует, что

$$(x+a)^n \equiv x^n + a \pmod{f(x), n} \tag{4.25}$$

для каждого целого $a$. Итак, если $n$ — простое, то (4.25) выполнено для каждого целого $a$ и для каждого многочлена $f(x)$ с целыми коэффициентами и единичным старшим коэффициентом. Далее, можно будет быстро проверить условие (4.25), если степень $\deg f(x)$ не слишком велика. К примеру, возьмем $a = 1$ и $f(x) = x - 1$. Тогда условие (4.25) будет эквивалентно условию

$$2^n \equiv 2 \pmod{n},$$

сравнению Ферма с основанием 2. Но, как мы видели, это сравнение является необходимым условием простоты $n$ и не является достаточным. Итак, вводя сравнение по модулю $f(x)$, мы выигрываем в скорости, но, возможно, теряем наш критерий простоты.

Однако условие (4.25) допускает большую свободу выбора; нам не обязательно брать многочлен $f(x)$ степени 1. Мы, например, могли бы взять $f(x) = x^r - 1$ для некоторого небольшого числа $r$ и, по существу, работать с корнями $r$-ой степени из единицы. В сущности, все, что нам нужно сделать, — это выбрать подходящее $r$ (которое не должно превосходить многочлена от логарифма $n$) и проверить условие (4.25) для каждого $a$ вплоть до некоторой границы (которая снова не превосходит многочлена от логарифма $n$).

Этот новый тест на простоту является настолько прямым и простым, что мы не можем устоять перед соблазном сразу привести его псевдокод, отложив обсуждение деталей на потом.

**Алгоритм 4.5.1** (тест на простоту Агравала—Кайала—Саксены (AKS)). Нам задано целое число $n \geqslant 2$. Этот детерминированный алгоритм решает, является число $n$ простым или составным.

1. [Проверка на степень]
   Если $n$ является квадратом или более высокой степенью, возвратить «$n$ — составное»;
2. [Начальная установка]
   Найти наименьшее целое число $r$, такое, что порядок числа $n$ в группе $\mathbf{Z}_r^*$ превосходит $\lg^2 n$;
   Если $n$ имеет собственный делитель в промежутке $[2, \sqrt{\varphi(r)}\lg n]$, возвратить «$n$ — составное»;
   // $\varphi$ — функция Эйлера.
3. [Биномиальные сравнения]
   for($1 \leqslant a \leqslant \sqrt{\varphi(r)}\lg n$) {
   if($(x+a)^n \not\equiv x^n + a \pmod{x^r - 1, n}$) return «$n$ — составное»;
   }
   return «$n$ — простое»;

Проверку на равенство квадрату на шаге [Проверка на степень] можно осуществить посредством алгоритма 9.2.11, а проверку на равенство более высокой степени — посредством сходной итерации Ньютона, ср. упр. 4.11. Заметим, что нам нужно лишь проверить, будет ли $n$ числом вида $a^b$ для $b \leqslant \lg n$. (Заметим также, что в силу упр. 4.28 шаг [Проверка на степень] можно вообще опустить!) Целое число $r$ на шаге [Начальная установка] может быть найдено путем последовательного перебора целых чисел, не превосходящих $\lg^2 n$. Если в процессе перебора найдено значение $r$, для которого $1 < \text{НОД}(r, n) < n$, то мы, очевидно, доказали, что $n$ — составное, и алгоритм можно изменить с учетом этого факта. При таком изменении нам нужно проводить последовательный поиск собственного делителя числа $n$ лишь в промежутке $[2, \lg^2 n]$, а не в промежутке $[2, \sqrt{\varphi(r)}\lg n]$, поскольку процедура поиска $r$ сама распознает те $n$, которые имеют собственный делитель в промежутке $(\lg^2 n, r]$, и $r > \sqrt{\varphi(r)}\lg n$. Поскольку шаг [Начальная установка] включает в себя поиск малых делителей числа $n$, может случиться так, что мы перебрали все возможности до $\sqrt{n}$, так что простота $n$ доказана. В таком случае нам, разумеется, не надо переходить к шагу [Биномиальные сравнения], но это может произойти лишь для довольно малых значений $n$. Дополнительные замечания по реализации AKS см. в конце разд. 4.5.4.

Мы вернемся к вопросу о величине числа $r$, когда будем обсуждать сложность этого алгоритма, но сначала рассмотрим вопрос о корректности алгоритма. Алгоритм 4.5.1 базируется на следующем удивительном критерии.

**Теорема 4.5.2** (Агравал, Кайал, Саксена). *Пусть $n$ — целое число, $n \geqslant 2$, $r$ — натуральное число, взаимно простое с $n$, такое что порядок числа $n$ в группе $\mathbf{Z}_r^*$ больше, чем $\lg^2 n$, и сравнение*

$$(x+a)^n \equiv x^n + a \pmod{x^r - 1, n} \tag{4.26}$$

*выполнено для каждого целого числа $a$, такого, что $0 \leqslant a \leqslant \sqrt{\varphi(r)}\lg n$. Если $n$ имеет простой делитель $p > \sqrt{\varphi(r)}\lg n$, то $n = p^m$ при некотором натуральном $m$. В частности, если $n$ не имеет простых делителей в промежутке $[1, \sqrt{\varphi(r)}\lg n]$, и $n$ не является точной степенью, то $n$ — простое.*

Доказательство. Мы можем считать, что $n$ имеет простой делитель $p > \sqrt{\varphi(r)}\lg n$. Положим

$$G = \{g(x) \in \mathbf{Z}_p[x] \ : \ g(x)^n \equiv g(x^n) \pmod{x^r - 1}\}.$$

Из условия (4.26) следует, что для каждого целого числа $a$, такого, что $0 \leqslant a \leqslant \sqrt{\varphi(r)}\lg n$, многочлен $x + a$ принадлежит $G$. Поскольку $G$ замкнуто относительно умножения, каждое выражение, имеющее вид одночлена

$$\prod_{0 \leqslant a \leqslant \sqrt{\varphi(r)}\lg n} (x + a)^{\varepsilon_a},$$

где каждое $\varepsilon_a$ — неотрицательное целое число, принадлежит $G$. Поскольку $p > \sqrt{\varphi(r)}\lg n$, все такие многочлены различны и не равны нулю в $\mathbf{Z}_p[x]$, так что $G$ содержит много элементов. Вскоре мы найдем удачное применение этому наблюдению.

Теперь покажем, что $G$ является объединением классов вычетов по модулю многочлена $x^r - 1$. Т. е. если $g_1(x) \in G$, $g_2(x) \in \mathbf{Z}_p[x]$ и $g_2(x) \equiv g_1(x) \pmod{x^r - 1}$, то $g_2(x) \in G$. В самом деле, заменяя каждое $x$ на $x^n$, получим $g_1(x^n) \equiv g_2(x^n) \pmod{x^{nr} - 1}$, и поскольку многочлен $x^r - 1$ делит $x^{nr} - 1$, данное сравнение будет справедливо и по модулю $x^r - 1$. Таким образом,

$$g_2(x)^n \equiv g_1(x)^n \equiv g_1(x^n) \equiv g_2(x^n) \pmod{x^r - 1},$$

так что $g_2(x) \in G$, что и утверждалось. Подведем итог:

- Множество $G$ замкнуто относительно умножения, каждый одночлен $x+a$ принадлежит $G$ при $0 \leqslant a \leqslant \sqrt{\varphi(r)}\lg n$, и $G$ является объединением классов вычетов по модулю многочлена $x^r - 1$.

Положим

$$J = \{j \in \mathbf{Z} \ : \ j > 0, \ g(x)^j \equiv g(x^j) \pmod{x^r - 1} \text{ for each } g(x) \in G\}.$$

По определению множества $G$ имеем, что $n \in J$, и, тривиально, $1 \in J$. Мы также имеем, что $p \in J$. В самом деле, для каждого многочлена $g(x) \in \mathbf{Z}_p[x]$ мы имеем, что $g(x)^p = g(x^p)$, так что это соотношение, очевидно, будет выполнено по модулю $x^r - 1$ для каждого $g \in G$. Легко видеть, что множество $J$ замкнуто относительно умножения. В самом деле, пусть $j_1, j_2 \in J$ и $g(x) \in G$. Мы имеем, что $g(x)^{j_1} \in G$, поскольку $G$ замкнуто относительно умножения, а поскольку $g(x)^{j_1} \equiv g(x^{j_1}) \pmod{x^r - 1}$, по утверждению из предыдущего абзаца $g(x^{j_1}) \in G$. Значит, поскольку $j_2 \in J$,

$$g(x)^{j_1 j_2} \equiv g(x^{j_1})^{j_2} \equiv g((x^{j_2})^{j_1}) = g(x^{j_1 j_2}) \pmod{x^r - 1},$$

так что $j_1 j_2 \in J$. Таким образом $J$ также содержит много элементов. Подведем итог:

- Множество $J$ содержит $1, n, p$ и является замкнутым относительно умножения.

Пусть $K$ есть поле разложения многочлена $x^r - 1$ над конечным полем $\mathbf{F}_p$. Следовательно, $K$ есть конечное поле характеристики $p$ и наименьшее поле, содержащее все корни $r$-ой степени из единицы. В частности, пусть $\zeta \in K$ есть примитивный корень $r$-ой степени из единицы, и пусть $h(x) \in \mathbf{F}_p[x]$ — минимальный многочлен для $\zeta$, так что $h(x)$ есть неприводимый делитель многочлена $x^r - 1$. Следовательно, $K = \mathbf{F}_p(\zeta) \cong \mathbf{F}_p[x]/(h(x))$. Степень $k$ многочлена $h(x)$ есть мультипликативный порядок элемента $p$ в группе $\mathbf{Z}_r^*$, но нам этот факт не понадобится. Ключевым фактом, который нам понадобится, будет то, что $K$ есть образ кольца $\mathbf{Z}_p[x]/(x^r - 1)$ под действием гомоморфизма, при котором класс, представляющий $x$, переходит в $\zeta$. В самом деле, все, что нужно для доказательства — это то, что $h(x) | x^r - 1$. Пусть $\overline{G}$ обозначает образ множества $G$ под действием этого гомоморфизма. Таким образом,

$$\overline{G} = \{\gamma \in K \ : \ \gamma = g(\zeta) \text{ для некоторого } g(x) \in G\}.$$

Заметим, что, если $g(x) \in G$ и $j \in J$, то $g(\zeta)^j = g(\zeta^j)$.

Пусть $d$ обозначает порядок подгруппы в $\mathbf{Z}_r^*$, порожденной элементами $n$ и $p$. Положим

$$G_d = \{g(x) \in G \ : \ g(x) = 0 \text{ или } \deg g(x) < d\}.$$

Поскольку $d \leqslant \varphi(r) < r$, все элементы множества $G_d$ различны по модулю $x^r - 1$. Мы покажем, что сужение нашего гомоморфизма в $K$ на множество $G_d$ является биекцией. Пусть $g_1(x), g_2(x) \in G_d$ и $g_1(\zeta) = g_2(\zeta)$. Мы утверждаем, что это влечет $g_1(x) = g_2(x)$. Если $j = n^a p^b$, где $a, b$ — неотрицательные целые числа, то $j \in J$, так что

$$g_1(\zeta^j) = g_1(\zeta)^j = g_2(\zeta)^j = g_2(\zeta^j).$$

Это равенство справедливо для $d$ различных значений вычета $j$ по модулю $r$. Но степени $\zeta^j$ различны, если показатели $j$ различны по модулю $r$, поскольку $\zeta$ — примитивный корень $r$-ой степени из единицы. Таким образом, многочлен $g_1(x) - g_2(x)$ имеет не менее $d$ различных корней в $K$. Но многочлен не может иметь в поле больше корней, чем его степень, и, поскольку $g_1(x), g_2(x)$ принадлежат $G_d$, должно выполняться равенство $g_1(x) = g_2(x)$, что и утверждалось. Подведем итог:

- Различные многочлены из $G_d$ соответствуют различным элементам множества $\overline{G}$.

Мы применим этот принцип к многочленам

$$g(x) = 0 \text{ or } g(x) = \prod_{0 \leqslant a \leqslant \sqrt{d} \lg n} (x + a)^{\varepsilon_a},$$

где теперь каждое $\varepsilon_a$ равно 0 или 1. Поскольку $d \leqslant \varphi(r)$, мы знаем, что каждый многочлен $g(x)$ принадлежит $G$. Далее, поскольку $d > \lg^2 n$, получаем

$\sqrt{d}\lg n < d$, т. е. при условии, что мы не выбираем все показатели $\varepsilon_a$ равными 1, получим, что каждый многочлен $g(x)$ лежит в $G_d$. Значит, имеется не менее

$$1 + (2^{\lfloor\sqrt{d}\lg n\rfloor+1} - 1) > 2^{\sqrt{d}\lg n} = n^{\sqrt{d}}$$

элементов во множестве $G_d$, и, тем самым, более, чем $n^{\sqrt{d}}$ элементов во множестве $\overline{G}$. Подведем итог:

- Мы имеем $\#\overline{G} \geqslant \#G_d > n^{\sqrt{d}}$.

Вспомним, что $K \cong \mathbf{F}_p[x]/(h(x))$, где $h(x)$ — неприводимый многочлен из $\mathbf{F}_p[x]$. Обозначим степень $h(x)$ через $k$. Таким образом, $K \cong \mathbf{F}_{p^k}$, так что если $j, j_0$ — натуральные числа, такие, что $j \equiv j_0 \pmod{p^k - 1}$ и $\beta \in K$, то $\beta^j = \beta^{j_0}$. Положим

$$J' = \{j \in \mathbf{Z} \,:\, j > 0,\ j \equiv j_0 \pmod{p^k - 1} \text{ для некоторого } j_0 \in J\}.$$

Если $j \equiv j_0 \pmod{p^k - 1}$, где $j_0 \in J$, и $g(x) \in G$, то $g(\zeta)^j = g(\zeta)^{j_0} = g(\zeta^{j_0}) = g(\zeta^j)$. Кроме того, поскольку $J$ замкнуто относительно умножения, таковым является и $J'$. Наконец, поскольку $np^{k-1} \equiv n/p \pmod{p^k - 1}$, мы имеем $n/p \in J'$. Подведем итог:

- Множество $J'$ замкнуто относительно умножения, содержит $1, p, n/p$, и для каждого $j \in J', g(x) \in G$ имеем $g(\zeta)^j = g(\zeta^j)$.

Рассмотрим целые числа $p^a(n/p)^b$, где $a, b$ — целые числа из промежутка $[0, \sqrt{d}]$. Поскольку $p, n/p$ лежат в подгруппе порядка $d$ в $\mathbf{Z}_r^*$, порожденной $p$ и $n$, и поскольку имеется более чем $d$ вариантов выбора упорядоченных пар $(a, b)$, должны существовать две различные пары $(a_1, b_1), (a_2, b_2)$, такие, что $j_1 := p^{a_1}(n/p)^{b_1}$ и $j_2 := p^{a_2}(n/p)^{b_2}$ сравнимы по модулю $r$. Следовательно, $\zeta^{j_1} = \zeta^{j_2}$, и, поскольку $j_1, j_2 \in J'$, мы имеем

$$g(\zeta)^{j_1} = g(\zeta^{j_1}) = g(\zeta^{j_2}) = g(\zeta)^{j_2} \text{ для всех } g(x) \in G.$$

Иными словами, $\gamma^{j_1} = \gamma^{j_2}$ для всех элементов $\gamma \in \overline{G}$. Но мы знаем, что $\overline{G}$ имеет более, чем $n^{\sqrt{d}}$ элементов, и поскольку $j_1, j_2 \leqslant p^{\sqrt{d}}(n/p)^{\sqrt{d}} = n^{\sqrt{d}}$, многочлен $x^{j_1} - x^{j_2}$ имеет слишком много корней в поле $K$, что не позволяет ему быть отличным от нулевого многочлена. Следовательно, мы должны иметь $j_1 = j_2$; другими словами, $p^{a_1}(n/p)^{b_1} = p^{a_2}(n/p)^{b_2}$. Значит,

$$n^{b_1 - b_2} = p^{b_1 - b_2 - a_1 + a_2},$$

и поскольку пары $(a_1, b_1), (a_2, b_2)$ различны, мы имеем $b_1 \neq b_2$. По теореме об однозначности разложения на простые сомножители в $\mathbf{Z}$, мы, таким образом, получаем, что $n$ есть степень числа $p$. □

Приведенное выше доказательство использует некоторые идеи, содержащиеся в лекциях [Agrawal 2003].

Корректность алгоритма 4.5.1 немедленно вытекает из теоремы 4.5.2; см. упр. 4.26.

### 4.5.2. Сложность алгоритма 4.5.1

Время на проверку каждого из сравнений

$$(x+a)^n \equiv x^n + a \pmod{x^r - 1, n}$$

на шаге [Биномиальные сравнения] алгоритма 4.5.1 полиномиально зависит от $r$ и от $\ln n$. Таким образом, нам важно показать, что $r$ также полиномиально зависит от $\ln n$. Этот факт вытекает из следующей теоремы.

**Теорема 4.5.3.** *Пусть дано целое число $n \geqslant 3$, и пусть $r$ — наименьшее целое число, такое, что порядок элемента $n$ в подгруппе $\mathbf{Z}_r^*$ превосходит $\lg^2 n$. Тогда $r \leqslant \lg^5 n$.*

Доказательство. Пусть $r_0$ — наименьшее простое число, которое не делит число

$$N := n(n-1)(n^2-1)\dots\left(n^{\lfloor \lg^2 n \rfloor} - 1\right).$$

Тогда $r_0$ — наименьшее простое число, такое, что порядок элемента $n$ в группе $\mathbf{Z}_{r_0}^*$ превосходит $\lg^2 n$, так что $r \leqslant r_0$. Из неравенства (3.16) в работе [Rosser and Schoenfeld 1962] следует, что произведение простых чисел из промежутка $[1, x]$ превосходит $2^x$ при $x \geqslant 41$. Более простая и более сильная оценка типа Чебышёва из упр. 1.28 дает $\prod_{p \leqslant x} p > 2^x$ при $x \geqslant 31$. Далее, произведение простых, делящих $N$, не превосходит $N$, а

$$N < n^{1+1+2+\ +\lfloor \lg^2 n \rfloor} = n^{\frac{1}{2}\lfloor \lg^2 n \rfloor^2 + \frac{1}{2}\lfloor \lg^2 n \rfloor + 1} < n^{\lg^4 n} = 2^{\lg^5 n}.$$

Значит, существует простое число $r_0 \leqslant \lg^5 n$, такое, что $r_0$ не делит $N$, при $\lg^5 n \geqslant 31$. Последнее неравенство выполняется при $n \geqslant 4$. Однако при $n = 3$ наименьшее значение $r$ с указанными в теореме свойствами равно 5, так что теорема верна и в этом случае. □

Итак, мы завершили доказательство того, что детерминированный алгоритм 4.5.1 за полиномиальное время решает, является $n$ простым или составным. Но как только решена одна проблема, естественным образом возникают новые. Среди них: какова точная оценка скорости работы алгоритма 4.5.1? Можно ли ее улучшить? Представляет ли он практическую ценность?

Сначала проанализируем алгоритм 4.5.1, когда он использует только элементарные, «наивные» процедуры. Битовая сложность проверки ровно одного сравнения на шаге [Биномиальные сравнения] есть $O(r^2 \ln^3 n)$. Так что время, за которое проверяются все сравнения, ограничено величиной $O(r^{2\,5} \ln^4 n)$. Используя значение $r = O(\ln^5 n)$ из теоремы 4.5.3, мы получаем, что суммарная битовая сложность всех сравнений есть $O(\ln^{16\,5} n)$. Нетрудно видеть, что каждая из оставшихся стадий алгоритма имеет сложность, ограниченную величиной меньшего порядка, так что наша первая $O$-оценка, а именно $O(\ln^{16\,5} n)$, и будет давать сложность алгоритма.

Бывают ситуации, когда лучше всего выбрать элементарный и наивный путь. Но для больших чисел и многочленов высокой степени требуются методы гл. 9. Для получения $(x+a)^n$ по модулю $x^r - 1$ и по модулю $n$ мы можем использовать схему возведения в степень (состоящую из $O(\ln n)$ шагов) со

встроенным умножением по модулю для многочленов, имеющих степень, меньшую $r$, и коэффициенты, всегда меньшие $n$. Так что главное вычисление — вычисление $(x+a)^n$ — сводится к $O((\ln n)(r \ln r)M(\ln n))$ битовым операциям, где $M(b)$ — битовая сложность умножения двух целых чисел, состоящих из $b$ битов (см., например, обсуждение, следующее за алгоритмом 9.6.1). Таким образом, с помощью быстрых алгоритмов время на проверку одного сравнения на шаге [Биномиальные сравнения] сокращается до величины $\tilde{O}(r \ln^2 n)$. (Запись $\tilde{O}(f(n))$ означает верхнюю границу вида $c_1 f(n)(\ln f(n))^{c_2}$ и иногда называется «мягким символом O». Так, если $g(n) = \tilde{O}(f(n))$, где $f(n)$ стремится к бесконечности, то $g(n) = O(f(n)^{1+\varepsilon})$ для каждого фиксированного $\varepsilon > 0$.) В итоге мы получаем, что битовая сложность сравнений и всего алгоритма в целом есть $\tilde{O}(r^{1.5} \ln^3 n) = \tilde{O}(\ln^{10.5} n)$.

Ясно, что, имея более точную верхнюю границу для $r$, чем та, которую дает теорема 4.5.3, мы улучшим оценку битовой сложности алгоритма 4.5.1. Например, используя упр. 4.29, мы получим, что битовая сложность данного алгоритма есть $\tilde{O}(\ln^6 n)$, если $n \equiv \pm 3 \pmod 8$. Поскольку на проверку одного сравнения на шаге [Биномиальные сравнения] вряд ли удастся затратить существенно меньше, чем $r \ln^2 n$ битовых операций, можно считать $r^{1.5} \ln^3 n$ предполагаемой *нижней границей* порядка величины битовой сложности алгоритма в целом (хотя это не обязательно будет нижней границей для какого-то другого, неизвестного, теста на простоту). И поскольку алгоритм заставляет нас выбирать $r > \lg^2 n$, можно предположить, что нам не удастся получить для общего времени работы оценку лучше, чем $\tilde{O}(\ln^6 n)$. Заметим также, что из упр. 4.30 следует, что это общее время работы в самом деле ограничено сверху величиной $\tilde{O}(\ln^6 n)$ для почти всех простых $n$. (Для большинства составных $n$ время работы меньше.)

Но нас в наших бесконечных поисках наилучшего возможного алгоритма интересует вопрос: можно ли достичь оценки $\tilde{O}(\ln^6 n)$ для *всех* чисел $n$. Наверное, можно. Т. е. мы, по-видимому, всегда сможем выбрать $r = \tilde{O}(\ln^2 n)$. Такой результат немедленно вытекает из двух отличных друг от друга гипотез в сильной форме. Одна из них — гипотеза Артина, утверждающая, что если число $n$ не $-1$ и не квадрат (что нам, разумеется, подходит), то существует бесконечно много простых чисел $r$, таких, что $n$ является первообразным корнем для $r$. Любое такое простое $r$, удовлетворяющее условию $r > 1 + \lg^2 n$, может быть использовано в алгоритме 4.5.1, и разумно предположить, что существует простое число с такими свойствами, меньшее $2 \lg^2 n$ (при $n > 2$). Интересно, что в книге [Hooley 1976][2] приведено доказательство гипотезы Артина в предположении обобщенной гипотезы Римана (GRH) (см. комментарии в упр. 2.39), и, может быть, это доказательство можно усилить и показать, что существует хорошее значение $r < 2 \lg^2 n$; см. упр. 4.38. Но если мы хотим предположить справедливость GRH, то мы могли бы просто предположить справедливость ERH и воспользоваться теоремой 3.5.13, таким образом получив детерминированный тест на простоту с битовой сложностью $\tilde{O}(\ln^4 n)$.

[2] Имеется перевод: [Хооли 1983]. — *Прим. ред.*

В дополнение к гипотезе Артина мы также имеем гипотезу о простых числах Софи Жермен. Напомним, что это те простые $q$, для которых $r = 2q + 1$ также является простым. Если их не только бесконечно много (что пока неизвестно), но и встречаются они достаточно часто, то должно существовать такое простое $q > \lg^2 n$, что $q = \tilde{O}(\ln^2 n)$ и $r = 2q+1$ не делит $n \pm 1$; см. [Agrawal et al. 2004]. Такое значение $r$ допустимо в алгоритме 4.5.1. В самом деле, нас устраивает, если порядок числа $n$ по модулю $r$ равен $q$ или $2q$. Но в противном случае его порядок равен 1 или 2, а мы отметили, что $r$ не делит $n \pm 1$. Указанные гипотезы укрепляют нашу уверенность в том, что сложность алгоритма 4.5.1 *должна быть* $\tilde{O}(\ln^6 n)$.

Используя фундаментальную теорему из работы [Fouvry 1985], можно показать, что $r$ может быть выбрано таким, что $r = O(\ln^3 n)$; см. [Agrawal et al. 2004]. Таким образом, общая битовая сложность данного алгоритма есть $\tilde{O}(\ln^{7.5} n)$. Это хороший результат, но использование теоремы Фуври имеет один недостаток. Ее доказательство не только сложное, но и *неэффективное.* Это означает, что доказательство не предоставляет возможности в явном виде выразить численное значение верхней границы для числа битовых операций. Эта неэффективность обусловлена использованием теоремы Зигеля; мы уже сталкивались с проявлениями теоремы Зигеля в теореме 1.4.6, и мы снова встретимся с ней, когда будем обсуждать числа классов квадратичных форм.

Итак, с помощью результата Фуври мы приблизились к нашей естественной границе $\tilde{O}(\ln^6 n)$, но все же не достигли ее, и полученная нами оценка времени работы неэффективна. В следующем пункте мы обсудим, как можно устранить эти недостатки.

### 4.5.3. Проверка на простоту с помощью гауссовых периодов

В теореме 4.5.2 мы имели дело с многочленом $x^r - 1$. В следующем результате из работы [Lenstra and Pomerance 2005] мы делаем шаг в направлении более общего многочлена $f(x)$.

**Теорема 4.5.4.** *Пусть $n$ — целое число, $n \geqslant 2$, $f(x)$ — нормированный многочлен из $\mathbf{Z}_n[x]$ степени $d$, где $d > \lg^2 n$,*

$$f(x^n) \equiv 0 \pmod{f(x)},\ x^{n^d} \equiv x \pmod{f(x)}, \tag{4.27}$$

*и*

$$x^{n^{d/q}} - x \text{ и } f(x) \text{ взаимно просты для всех простых чисел } q, \text{ делящих } d. \tag{4.28}$$

*Пусть, далее,*

$$(x + a)^n \equiv x^n + a \pmod{f(x)} \tag{4.29}$$

*для каждого целого числа $a$, такого, что $0 \leqslant a \leqslant \sqrt{d}\lg n$. Тогда, если $n$ делится на простое число $p > \sqrt{d}\lg n$, то $n = p^m$ при некотором натуральном $m$.*

Понятие взаимной простоты двух многочленов в $\mathbf{Z}_n[x]$ было введено в определении 4.3.1. Заметим, что коэффициенты всех многочленов берутся по модулю $n$, поскольку предполагается, что многочлены в теореме 4.5.4 лежат в $\mathbf{Z}_n[x]$.

Доказательство. Мы почти полностью следуем доказательству теоремы 4.5.2. Пусть $p$ — простой делитель числа $n$, превосходящий $\sqrt{d}\lg n$. Действуя как и раньше, но подставляя $f(x)$ вместо $x^r - 1$, определим

$$G = \{g(x) \in \mathbf{Z}_p[x] \ : \ g(x)^n \equiv g(x^n) \pmod{f(x)}\}.$$

Так же, как и раньше, но теперь ввиду условия (4.27), имеем, что $f(x)|f(x^n)$ в $\mathbf{Z}_p[x]$. Следовательно, $G$ замкнуто относительно умножения и является объединением классов вычетов по модулю $f(x)$. Следовательно, наше доказательство того, что

$$J = \{j \in \mathbf{Z} \ : \ j > 0, \ g(x)^j \equiv g(x^j) \pmod{f(x)} \text{ для всех } g(x) \in G\}$$

замкнуто относительно умножения, такое же, как и раньше. Пусть $h(x)$ — неприводимый делитель многочлена $f(x)$ как многочлена по модулю $p$, и через $\zeta$ обозначен корень $h(x)$ в поле разложения $K$ многочлена $h(x)$ над $\mathbf{F}_p$. Тогда конечное поле $K = \mathbf{F}_p(\zeta)$ есть образ кольца $\mathbf{Z}_p[x]/(f(x))$ под действием гомоморфизма, при котором класс, представляющий $x$, переходит в $\zeta$. По условию (4.28) многочлен $x$ взаимно прост с $f(x)$ в $\mathbf{Z}_p[x]$, так что $\zeta \neq 0$ в $K$. Пусть $r$ есть мультипликативный порядок элемента $\zeta$. По условию (4.28) должно быть $\zeta^{n^{d/q}} \neq \zeta$ для каждого простого $q|d$, так что $\zeta^{n^{d/q}-1} \neq 1$ для таких $q$. Также, ввиду условия (4.27) и того факта, что $\zeta$ не равно нулю в $K$, мы имеем $\zeta^{n^d-1} = 1$. Следовательно, порядок числа $n$ в группе $\mathbf{Z}_r^*$ в точности равен $d$.

При доказательстве теоремы 4.5.2 $d$ было равно порядку подгруппы, порожденной $n$ и $p$ в $\mathbf{Z}_r^*$, в то время как сейчас это просто порядок подгруппы, порожденной $n$. Однако в нашем теперешнем контексте эти две подгруппы совпадают; т. е. $p \equiv n^i \pmod{r}$ для некоторого неотрицательного целого числа $i$. В этом мы можем убедиться следующим образом. Во-первых, заметим, что $f(x) \in G$, поскольку $f(x^n) \equiv 0 \equiv f(x)^n \pmod{f(x)}$. Следовательно, $f(\zeta)^j = f(\zeta^j)$ для всех $j \in J$. Но $f(\zeta) = 0$, так что каждое $\zeta^j$ является корнем многочлена $f$ в $K$. Итак, $\zeta$ имеет порядок $r$, а $f$ имеет степень $d$, так что число классов вычетов, пробегаемых величиной $j \bmod r$ для $j \in J$ не превосходит $d$; в самом деле, многочлен $f$ не может иметь в конечном поле $K$ больше корней, чем его степень. Но степени $n$ уже занимают $d$ классов вычетов по модулю $r$, так что любой другой элемент множества $J$, в частности $p$, сравним по модулю $r$ с некоторой степенью числа $n$. (Читатель может заметить сходство между этим рассуждением и доказательством теоремы 4.3.3.)

При доказательстве теоремы 4.5.2 мы имели $x + a \in G$ для каждого целого числа $a$, такого, что $0 \leqslant a \leqslant \sqrt{\varphi(r)}\lg n$, но мы пользовались только тем, что это условие выполнено при $0 \leqslant a \leqslant \sqrt{d}\lg n$. Нам известно, что последнее из этих условий выполнено. С этого момента все совпадает, и доказательство можно завершить точно так же, как доказательство теоремы 4.5.2. □

В приведенном доказательстве использовались некоторые идеи замечательной обзорной статьи [Granville 2004a].

В ситуации с теоремой 4.5.2 ограничение состояло в том, что, хоть мы и предположили существование подходящих значений $r$, достаточно близких к $\lg^2 n$, все, что нам удалось доказать — это $r \leqslant \lg^5 n$ (теорема 4.5.3), хотя, в свою очередь, при помощи неэффективных методов эта верхняя граница для $r$ может быть снижена до $O(\ln^3 n)$. Но в ситуации с теоремой 4.5.4 мы не обязаны рассматривать лишь многочлены вида $x^r - 1$. У нас теперь есть полное право рассмотреть все возможные нормированные многочлены $f(x)$, коль скоро их степень превосходит $\lg^2 n$, и они удовлетворяют условиям (4.27) и (4.28). Заметим, что если $n$ — простое, то по теореме 2.2.8 многочлен $f(x)$ удовлетворяет (4.27) и (4.28) тогда и только тогда, когда $f(x)$ неприводим в $\mathbf{Z}_n[x]$. И легко показать, что существует много нормированных неприводимых многочленов любой заданной степени (см. (2.5) и упр. 2.12). Так почему просто не взять $d = \lfloor \lg^2 n \rfloor + 1$ и не выбрать многочлен степени $d$, который был бы неприводим в случае простого $n$?

К несчастью, все не так просто. Построение неприводимых многочленов над полем $\mathbf{F}_p$, где $p$ — простое, можно осуществить за ожидаемое полиномиальное время посредством случайного алгоритма, состоящего в простом выборе произвольных многочленов нужной степени и их последующей проверке. Этот подход в точности совпадает с алгоритмом 4.3.4. Но что делать, если нам нужен детерминированный алгоритм? Уже в случае, когда степень равна 2, мы сталкиваемся с известной трудной задачей, поскольку нахождение неприводимого квадратичного многочлена в $\mathbf{F}_p[x]$ эквивалентно нахождению квадратичного невычета по модулю $p$. В предположении расширенной гипотезы Римана (ERH) мы знаем, как сделать это за детерминированное полиномиальное время (с помощью теоремы 1.4.5). Но нам не известно ни одного безусловного полиномиального метода. В работе [Adleman and Lenstra 1986] показано, как детерминированно находить неприводимые многочлены любой заданной степени за время, полиномиальное относительно $\ln p$ и этой степени, вновь в предположении ERH. Авторы также приводят безусловный вариант своей теоремы, в котором они допускают возникновение небольшой «погрешности». А именно, если желаемая степень равна $d$, они безусловно и за полиномиальное относительно $\ln p$ и $d$ время находят неприводимый многочлен по модулю $p$ степени $D$, где $d \leqslant D = O(d \ln p)$. В статье [Lenstra and Pomerance 2005] получено улучшение последнего результата, которое позволяет находить неприводимый многочлен по модулю $p$, степень которого принадлежит отрезку $[d, 4d]$, если $p$ достаточно велико (нижняя граница в принципе вычислима) и если $d > (\ln p)^{1.84}$. (Если мы не настаиваем на эффективности, нижнюю оценку для $d$ можно несколько ослабить.) Кроме того, число битовых операций, необходимых для нахождения такого многочлена, ограничено величиной $\tilde{O}(d^{8/5} \ln n)$ (обозначение $\tilde{O}$ введено в предыдущем пункте).

Пусть мы взяли $d = \lfloor \lg^2 n \rfloor + 1$ и запустили последний алгоритм с большим числом $n$. Если $n$ — простое, то этот алгоритм выдаст неприводимый многочлен, степень которого принадлежит отрезку $[d, 4d]$. Если $n$ — составное, то либо алгоритм выдаст многочлен, степень которого принадлежит $[d, 4d]$, для которого выполнены оба условия (4.27) и (4.28), либо произойдет сбой алгорит-

ма. В последнем случае будет доказано, что число $n$ — составное. Наконец, если алгоритму удастся найти многочлен, для которого выполнены условия (4.27) и (4.28), то можно переходить к проверке условия (4.29) для указанных значений $a$, на что потребуется время $\tilde{O}(d^{3/2}\ln^3 n) = \tilde{O}(\ln^6 n)$; тем самым мы установим с указанной оценкой времени работы, является число $n$ простым или составным.

Итак, построение многочленов из работы [Lenstra and Pomerance 2005] плюс теорема 4.5.4 дают детерминированный тест на простоту числа $n$ с числом битовых операций, ограниченным величиной $\tilde{O}(\ln^6 n)$. Данный метод построения многочленов слишком сложен для того, чтобы полностью описать его в этой книге, но мы хотели бы представить некоторые существенные его элементы. Как и в случае многих других идей, рассмотренных нами, эта история восходит к Гауссу.

Будучи еще совсем юным, Гаусс описал множество натуральных чисел $n$, для которых правильный $n$-угольник можно построить с помощью евклидовых циркуля и линейки, и предположил, что это множество исчерпывает все возможные случаи (он оказался прав, как было доказано Вантцелем в 1836 г.). Множество Гаусса — это в точности те целые числа $n \geqslant 3$, для которых $\varphi(n)$ есть степень числа 2 (также см. обсуждение в разд. 1.3.2). Нас будет интересовать не столько сама эта замечательная теорема, сколько ее доказательство. Ключом к рассуждению является то, что носит сейчас название гауссовых периодов. Пусть $r$ — простое число, и пусть $\zeta_r = e^{2\pi\imath/r}$, так что $\zeta_r$ есть примитивный корень $r$-й степени из 1. Пусть $d$ — положительный делитель числа $r-1$, и пусть

$$S = \{1 \leqslant j \leqslant r \ : \ j^{(r-1)/d} \equiv 1 \pmod r\}$$

есть подгруппа $d$-х степеней по модулю $r$. Определим гауссов период

$$\eta_{r,d} = \sum_{j\in S} \zeta_r^j.$$

Таким образом, $\eta_{r,d}$ есть сумма некоторого числа корней $r$-й степени из 1. Она обладает тем свойством, что $\mathbf{Q}(\eta_{r,d})$ есть (единственное) подполе поля $\mathbf{Q}(\zeta_r)$, имеющее степень $d$ над $\mathbf{Q}$. На самом деле, $\eta_{r,d}$ есть след элемента $\zeta_r$ по отношению к этому подполю. В особенности нас интересует минимальный многочлен $f_{r,d}$ для $\eta_{r,d}$ над $\mathbf{Q}$. Это нормированный многочлен с целыми коэффициентами, имеющий степень $d$ и неприводимый над $\mathbf{Q}$. Мы можем явно выписать многочлен $f_{r,d}$ следующим образом. Пусть $w$ — вычет по модулю $r$, такой, что порядок элемента $w^{(r-1)/d}$ равен $d$. Например, данным свойством обладает любой первообразный корень $w$ по модулю $r$, но можно привести и много других примеров. Тогда классы $S, wS, \ldots, w^{d-1}S$ не пересекаются и покрывают $\mathbf{Z}_r^*$. Сопряженными элементами для $\eta_{r,d}$ над $\mathbf{Q}$ являются всевозможные суммы $\sum_{j\in w^\imath S} \zeta_r^j$, и мы имеем

$$f_{r,d}(x) = \prod_{\imath=0}^{d-1}\Bigl(x - \sum_{j\in w^\imath S} \zeta_r^j\Bigr).$$

Будучи нормированным многочленом степени $d$ из $\mathbf{Z}[x]$, после приведения по модулю простого числа $p$ многочлен $f_{r,d}$ остается многочленом степени $d$. Но будет ли он неприводимым в $\mathbf{Z}_p[x]$? Не обязательно. Однако следующий результат содержит условие, гарантирующее неприводимость $f_{r,d}$ после редукции по модулю $p$.

**Лемма 4.5.5** (Куммер). *Если $r$ — простое число, $d$ — положительный делитель числа $r-1$ и $p$ — такое простое число, что порядок элемента $p^{(r-1)/d}$ по модулю $r$ равен $d$, то многочлен $f_{r,d}(x)$ остается неприводимым и как многочлен в $\mathbf{F}_p[x]$.*

Доказательство этого результата, использующее тот факт, что $\eta_{r,d}$ и его сопряженные образуют целочисленный базис кольца целых чисел в $\mathbf{Q}(\eta_{r,d})$, можно найти в работе [Adleman and Lenstra 1986]. Мы приведем другое доказательство, использующее суммы Гаусса.

Доказательство. Рассмотрим поле разложения $K$ многочлена $(x^r-1)(x^d-1)$ над $\mathbf{F}_p$, которое можно представить как гомоморфный образ кольца $\mathbf{Z}[\zeta_r, \zeta_d]$, где $\zeta_r = e^{2\pi\imath/r}$ и $\zeta_d = e^{2\pi\imath/d}$. Пусть $\zeta$ есть образ $\zeta_r$ в $K$ и пусть $\omega$ есть образ $\zeta_d$. Далее, пусть $\eta = \sum_{j\in S}\zeta^j$ есть образ $\eta_{r,d}$. Исходя из предположения, что порядок элемента $p^{(r-1)/d}$ по модулю $r$ есть $d$, мы собираемся показать, что $\eta$ имеет степень $d$ над $\mathbf{F}_p$ (мы имеем $f_{r,d}(\eta) = 0$, и $f_{r,d}$ имеет степень $d$, так что если $\eta$ имеет степень $d$, то $f_{r,d}$ должен быть неприводим над $\mathbf{F}_p$). Мы применяем отображение Фробениуса $p$-й степени к $\eta$; если сделать это $i$ раз, получим $\eta^{p^i}$. Мы собираемся показать, что наименьшее положительное $k$, такое что $\eta^{p^k} = \eta$ есть $k = d$. Для каждого $k$ имеем

$$\eta^{p^k} = \sum_{j\in S}\zeta^{p^k j} = \sum_{j\in p^k S}\zeta^j,$$

так что $\eta^{p^d} = \eta$, поскольку $p^d \in S$. Таким образом, наименьшее положительное $k$, такое, что $\eta^{p^k} = \eta$, есть делитель числа $d$. Наша цель — показать, что $k = d$, так что можно считать $d > 1$.

Пусть $\chi$ — характер Дирихле по модулю $r$ порядка $d$; конкретно, пусть $\chi(a^d) = 1$ для любого ненулевого вычета $a$ по модулю $r$ и пусть $\chi(p) = \zeta_d$. (Поскольку порядок элемента $p^{(r-1)/d}$ по модулю $r$ по предположению равен $d$, мы имеем полностью определенный $\chi$.) Рассмотрим сумму Гаусса

$$\tau(\chi) = \sum_{j=1}^{r-1}\chi(j)\zeta_r^j.$$

Заметим, что доказательство леммы 4.4.1 дает равенство $|\tau(\chi)|^2 = r$ (см. упр. 4.21), так что образ $\tau(\chi)$ в $K$ не нулевой. Перегруппировав эту сумму Гаусса, получаем, что

$$\tau(\chi) = \sum_{i=0}^{d-1}\sum_{j\in p^i S}\chi(j)\zeta_r^j = \sum_{i=0}^{d-1}\chi(p)^i\sum_{j\in p^i S}\zeta_r^j.$$

Таким образом, $\tau(\chi)$ есть «деформированная» сумма по множеству комплексных корней многочлена $f_{r,d}(x)$. Мы переносим данное равенство на $K$, замечая, что $\sum_{j\in p^i S}\zeta^j = \eta^{p^i}$. Но $\eta^{p^{i_1}} = \eta^{p^{i_2}}$ как только $i_1 \equiv i_2 \pmod k$, так что образ $\tau(\chi)$ в $K$ есть

$$\sum_{i=0}^{d-1}\omega^i\sum_{j\in p^iS}\zeta^j = \sum_{i=0}^{d-1}\omega^i\eta^{p^i} = \sum_{m=0}^{k-1}\eta^{p^m}\sum_{l=0}^{d/k-1}\omega^{m+kl} = \sum_{m=0}^{k-1}\eta^{p^m}\omega^m\sum_{l=0}^{d/k-1}\omega^{kl}.$$

Но если $k < d$, последняя внутренняя сумма есть 0, так что образ $\tau(\chi)$ в $K$ есть 0 — противоречие. Значит, $k = d$, что и требовалось доказать. □

В упр. 4.31 обсуждается утверждение, обратное лемме 4.5.5.

А теперь пусть у нас есть множество пар $r_i, d_i$, в которых каждое $r_i$ — простое и $d_i | r_i - 1$ при $i = 1, \ldots, k$. Пусть $\eta$ есть произведение всевозможных гауссовых периодов $\eta_{r_i,d_i}$, и пусть $f$ — его минимальный многочлен над $\mathbf{Q}$. Если числа $d_i$ попарно взаимно просты, то степень многочлена $f$ есть произведение $d_1 \ldots d_k$. И нетрудно убедиться, что если $p$ — простое, не совпадающее ни с одним из $r_i$, то $f$ неприводим по модулю $p$, если порядок вычета $p^{(r_i-1)/d_i}$ в $\mathbf{Z}^*_{r_i}$ равен $d_i$ при $i = 1, \ldots, k$; см. упр. 4.32. А эту процедуру можно рассматривать как своего рода «машину» для производства неприводимых многочленов по модулю $p$. То, что с помощью этой «машины» можно попасть в малую окрестность желаемой степени, получается из следующего результата работы [Lenstra and Pomerance 2005].

**Теорема 4.5.6.** *Существует такое число $B$ (оно, в принципе, вычислимо), что если $n$ — целое число, $n > B$, и $d$ — целое число, $d > (\ln n)^{1\,84}$, то существует бесквадратное число $D$ в интервале $[d, 4d]$, такое, что каждый простой делитель $q$ числа $D$ удовлетворяет условиям* (1) $q < d^{3/11}$ *и* (2) *существует простое число $r < d^{6/11}$, причем $r \equiv 1 \pmod q$, такое, что $r$ не делит $n$, и $n$ не является $q$-й степенью по модулю $r$.*

Заметим, что, поскольку $q$ — простое, высказывание «$n$ не является $q$-й степенью по модулю $r$» эквивалентно высказыванию «$n^{(r-1)/q}$ имеет порядок $q$ по модулю $r$».

Вооруженные теоремой 4.5.6, мы можем быть уверены, что найдем число $D$ с указанными свойствами, и этот поиск легко осуществим. Поскольку граница $B$ пока не вычислена, мы можем сомневаться в том, что нужное нам $D$ будет из промежутка $[d, 4d]$ для заданного числа $n$, но последовательный поиск с началом в $d$ в итоге приведет к подходящему числу $D$, которое есть $O(d)$, причем константа в символе $O$ также в принципе вычислима. Как только найдено $D$, можно задействовать нашу «машину» гауссовых периодов для построения многочлена $f$ степени $D$, который был бы неприводимым, если бы число $n$ было простым.

Итак, применяя подход теоремы 4.5.4 и используя гауссовы периоды для построения подходящих многочленов, можно построить детерминированный тест на простоту с (эффективным) временем работы, которое ограничено величиной $\tilde{O}(\ln^6 n)$ битовых операций. Мы представили некоторые из ключевых идей.

Само доказательство, в особенности, доказательство теоремы 4.5.6, довольно сложное и выходит за рамки этой книги. Подробности см. в работе [Lenstra and Pomerance 2005]. В заключение укажем, что тест на простоту Агравала—Кайала—Саксены в варианте Ленстры—Померанса, который обсуждался в настоящем пункте, не дает практических преимуществ по сравнению с алгоритмом 4.5.1, поскольку на практике последний всегда будет находить малое $r$, так что он не такой уж обременительный. (Чтобы доказать это, мы и погрузились в изучение метода настоящего пункта.) А теперь, после того как мы приоткрыли дверь, ведущую к практическим аспектам нового теста на простоту, можно оставить в стороне вопрос о детерминизме и даже о строгом обосновании алгоритма и выяснить, действительно ли эти новые идеи могут помочь нам при доказательстве простоты больших чисел. Далее мы займемся этим вопросом.

### 4.5.4. Квартичный тест на простоту

Поскольку наиболее трудоемкой стадией алгоритма 4.5.1 является проверка сравнения $(x+a)^n \equiv x^n + a \pmod{x^r-1, n}$ для большого количества значений $a$, эта стадия может оказаться подходящей для поиска оптимизаций. В теореме 4.5.4 мы имели улучшение, связанное с заменой $x^r-1$ на многочлен $f(x)$, степень которого будет, возможно, меньше. Другая идея состоит в том, чтобы биномиальные сравнения проверялись «без затрат». В следующей теореме мы заменяем $x^r-1$ на $x^r-b$ при подходящем целом $b$, и нам необходимо проверить лишь одно биномиальное сравнение.

**Теорема 4.5.7.** *Пусть $n, r, b$ — целые числа, такие, что $n > 1$, $r|n-1$, $r > \lg^2 n$, $b^{n-1} \equiv 1 \pmod{n}$ и $\text{НОД}(b^{(n-1)/q}-1, n) = 1$ для каждого простого $q|r$. Если*

$$(x-1)^n \equiv x^n - 1 \pmod{x^r-b, n}, \tag{4.30}$$

*то $n$ является простым или степенью простого числа.*

Доказательство. Пусть простое число $p|n$ и пусть $A = b^{(n-1)/r} \bmod p$. Тогда $A$ имеет порядок $r$ в $\mathbf{Z}_p^*$, так что, в частности, $r|p-1$ (см. теорему Поклингтона 4.1.3). Заметим, что

$$x^n = x \cdot x^{n-1} = x(x^r)^{(n-1)/r} \equiv Ax \pmod{x^r-b, p}. \tag{4.31}$$

Значит, по нашему предположению,

$$(x-1)^n \equiv x^n - 1 \equiv Ax - 1 \pmod{x^r-b, p}.$$

Также заметим, что если $f(x) \equiv g(x) \pmod{x^r-b, p}$, то $f(A^i x) \equiv g(A^i x) \pmod{x^r-b, p}$ для любого целого $i$, поскольку $(A^i x)^r - b \equiv x^r - b \pmod{p}$. Таким образом, полагая $f(x) = (x-1)^n$ и $g(x) = Ax - 1$, мы получим

$$(x-1)^{n^2} \equiv (Ax-1)^n \equiv A^2 x - 1 \pmod{x^r-b, p},$$

и далее по индукции

$$(x-1)^{n^j} \equiv A^j x - 1 \pmod{x^r-b, p} \tag{4.32}$$

для каждого неотрицательного целого $j$.

Заметим, что если $c$ — целое и $c^r \equiv 1 \pmod p$, то $c \equiv A^k \pmod p$ для некоторого целого $k$; в самом деле, все, что нужно для доказательства этого наблюдения, — это то, что $p$ — простое и $A$ имеет порядок $r$ по модулю $p$. Итак, мы имеем

$$x^p = x \cdot x^{p-1} = x(x^r)^{(p-1)/r} \equiv b^{(p-1)/r} x \equiv A^k x \pmod{x^r - b, p}$$

для некоторого целого $k$. Таким образом, поскольку $(A^k)^p \equiv A^k \pmod p$, мы по индукции получаем, что

$$x^{p^i} \equiv A^{ik} x \pmod{x^r - b, p} \tag{4.33}$$

для каждого неотрицательного целого $i$. Мы имеем $f(x)^{p^i} = f(x^{p^i})$ для каждого $f(x) \in \mathbf{Z}_p[x]$, так что в силу (4.32) и (4.33) мы имеем

$$(x-1)^{p^i n^j} \equiv (A^j x - 1)^{p^i} \equiv A^j x^{p^i} - 1 \equiv A^{j+ik} x - 1 \pmod{x^r - b, p}$$

для всех неотрицательных целых чисел $i, j$. Так что для таких $i, j$ мы получаем

$$(x-1)^{p^i (n/p)^j} \equiv A^{j(1-k)+ik} x - 1 \pmod{x^r - b, p}, \tag{4.34}$$

поскольку обе части имеют одинаковую $p^j$-ю степень $\pmod{x^r - b, p}$, и возведение в $p^j$-ю степень взаимно однозначно в $\mathbf{Z}_p[x]/(x^r - b)$. Последнее утверждение справедливо ввиду того, что возведение в $p$-ю степень взаимно однозначно в любом кольце $\mathbf{Z}_p[x]/(f(x))$, для которого $f(x)$ не имеет кратных неприводимых сомножителей по модулю $p$, и поскольку $\text{НОД}(x^r - b, rx^{r-1}) = 1$ в $\mathbf{Z}_p[x]$, многочлен $x^r - b$ в самом деле не имеет кратных сомножителей.

Заметим, что $x-1$ является единицей в кольце $\mathbf{Z}_p[x]/(x^r - b)$. В самом деле, в $\mathbf{Z}_p[x]$ мы имеем $\text{НОД}(x-1, x^r - b) = \text{НОД}(x-1, 1-b) = 1$, при условии, что $p$ не делит $b-1$. Но поскольку $A = b^{(n-1)/r}$ по модулю $p$ имеет порядок $r > \lg^2 n \geqslant 1$, $p$ в самом деле не делит $b-1$. Обозначим через $E$ мультипликативный порядок элемента $x-1$ в кольце $\mathbf{Z}_p[x]/(x^r - b)$. Заметим, что

$$E \geqslant 2^r - 1,$$

поскольку многочлены

$$\prod_{j \in S} (A^j x - 1),$$

где $S$ пробегает собственные подмножества чисел $\{0, 1, \ldots, r-1\}$, не только различны в кольце $\mathbf{Z}_p[x]/(x^r - b)$, но и каждый из них является степенью многочлена $x-1$ в силу (4.32).

Рассмотрим целые числа $i, j$, такие, что $0 \leqslant i, j \leqslant \sqrt{r}$. Должны существовать две различные пары $(i_1, j_1), (i_2, j_2)$, такие, что

$$j_1(1-k) + i_1 k \equiv j_2(1-k) + i_2 k \pmod r,$$

так что если $u_1 = p^{i_1}(n/p)^{j_1}, u_2 = p^{i_2}(n/p)^{j_2}$, то

$$(x-1)^{u_1} \equiv A^{j_1(1-k)+i_1 k} x - 1 \equiv A^{j_2(1-k)+i_2 k} x - 1 \equiv (x-1)^{u_2} \pmod{x^r - b, p}.$$

Следовательно,

$$u_1 \equiv u_2 \pmod E.$$

Но $u_1, u_2 \in [1, n^{\sqrt{r}}]$ и $E > 2^r - 1 > n^{\sqrt{r}} - 1$, последнее неравенство следует из условия, что $r > \lg^2 n$. Итак, $u_1 = u_2$, и как мы видели при доказательстве теоремы 4.5.2, это сразу приводит к тому, что $n$ есть степень числа $p$. □

В сущности, данная теорема содержится в работах [Bernstein 2003] и (независимо) [Mihăilescu and Avanzi 2003]. Первоначально она была доказана Беррицбейтиа в случае, когда $r$ является степенью числа 2, и Ченом в случае, когда $r$ является простым или степенью простого числа.

Заметим, что при использовании быстрой арифметики для многочленов и целых чисел, сравнение (4.30) можно проверить за $\tilde{O}(r \ln^2 n)$ битовых операций; обозначение $\tilde{O}$ введено нами в разд. 4.5.2. Так что если бы нам удалось выбрать $r$ таким, что $r = O(\ln^2 n)$, это послужило бы основой для создания теста на простоту, сложность которого $\tilde{O}(\ln^4 n)$. Но здесь возникают две проблемы. Во-первых, не каждое простое число $n$ имеет делитель $r$ числа $n-1$ такой, что $\lg^2 n < r = O(\ln^2 n)$; на самом деле, можно показать, что для большинства простых чисел $n$ число $n-1$ *не имеет* такого делителя $r$. Во-вторых, даже если бы мы имели такое $r$, мы столкнулись бы с проблемой выбора $b$. Разумеется, если $n$ — простое, то существует много подходящих чисел $b$. Действительно, в качестве $b$ можно взять первообразный корень по модулю $n$; существуют и другие подходящие числа. Так что мы без труда нашли бы подходящее $b$ методом подбора, однако мы не знаем, как решить подобную задачу за детерминированно полиномиальное время без каких-либо дополнительных предположений, таких как расширенная гипотеза Римана (ERH).

Откажемся на время от детерминизма. Если $n-1$ имеет делитель $r$, такой, что $\lg^2 n < r = O(\ln^2 n)$, и $n$ — простое, мы можем использовать быстрый метод подбора $b$, показать, что $n$ не является точной степенью, а затем на основании теоремы 4.5.7 провести доказательство простоты $n$, требующее $\tilde{O}(\ln^4 n)$ битовых операций. Оказывается, Бернштейн использовал именно этот тест и с его помощью доказал простоту числа, состоящего из 1000 битов. Пока этот тест не выглядит предпочтительнее теста, основанного на суммах Якоби, или доказательства простоты при помощи эллиптических кривых, но он превращается в один из возможных вариантов.

Обратимся к более серьезному вопросу: что делать в случае, когда $n-1$ не имеет делителя $r > \lg^2 n$, который не велик? В работе [Berrizbeitia 2002] показано, как можно быстро доказывать простоту числа $n$, если $n+1$ делится на степень числа 2, имеющую порядок $\lg^2 n$. Читатель может заметить параллели с ранее изложенным, поскольку, в некотором смысле, данная глава прошла полный круг. Мы столкнулись с ограничениями, присущими $(n-1)$-тесту, что привело нас к $(n+1)$-тесту, и, наконец, к тесту, основанному на конечных полях, где мы ищем подходящий делитель $y$ числа вида $n^d - 1$ при некотором небольшом целом $d$. Заметим, что как следует из теоремы 4.3.5 при $x = \lg^2 n$, если $n > 16$ (так что $\lg^2 n > 16$), существует натуральное число $d < (2 \ln \ln n)^{c \ln \ln \ln (\ln^2 n)}$, такое, что $n^d - 1$ имеет делитель $r > \lg^2 n$, и такое, что каждый простой делитель числа $r$ на единицу больше одного из делителей числа $d$. Значит, отбросив при необходимости некоторые из этих простых

делителей числа $r$, мы можем считать, что $\lg^2 n < r \leqslant (d+1)\lg^2 n$. В нижеследующем результате нам нужно чуть большее значение $r$, а именно, $r > d^2 \lg^2 n$, но ситуация, в сущности, такая же; существует некоторое $d$, ограниченное величиной $(\ln\ln n)^{O(\ln\ln\ln n)}$, такое, что $n^d - 1$ имеет делитель $r$ с условием $d^2\lg^2 n < r \leqslant (d+1)d^2\lg^2 n$. Следующий результат, взятый из работ [Bernstein 2003] и [Mihăilescu and Avanzi 2003], позволяет построить быстро проверяемое условие простоты, в котором используются такие вспомогательные числа $r, d$.

**Теорема 4.5.8.** *Пусть $n, r, d$ — целые числа, $n > 1$, $r | n^d - 1$, $r > d^2\lg^2 n$. Пусть также $f(t)$ — нормированный многочлен из $\mathbf{Z}_n[t]$ степени $d$, $R$ есть кольцо $\mathbf{Z}_n[t]/(f(t))$, и пусть элемент $b = b(t) \in R$ таков, что $b^{n^d-1} = 1$, и $b^{(n^d-1)/q} - 1$ совпадает с нулем кольца $R$ для каждого простого $q | r$. Если*

$$(x-1)^{n^d} \equiv x^{n^d} - 1 \pmod{x^r - b}$$

*в кольце $R[x]$, то $n$ — простое или степень простого числа.*

Доказательство теоремы 4.5.8 во многом очень схоже с доказательством теоремы 4.5.7, так что мы приведем лишь схему. Пусть $p$ — простой делитель числа $n$ и $h(t)$ — неприводимый делитель многочлена $f(t)$ по модулю $p$. Пусть $K$ есть конечное поле $\mathbf{Z}_p[t]/(h(t))$, так что $K$ есть гомоморфный образ кольца $R$. Положим $N = n^d$ и $P = p^{\deg h}$, так что $P \mid p^d \mid N$. Отождествим $b$ с его образом в $K$ и положим $A = b^{(N-1)/r}$, так что по условию $A$ имеет порядок $r$. Тогда существует некоторое целое число $k$, такое, что для всех неотрицательных целых чисел $j, i$

$$(x-1)^{N^j} \equiv A^j x - 1 \pmod{x^r - b}, \quad (x-1)^{P^i} \equiv A^{ik} x - x \pmod{x^r - b},$$

где многочлены рассматриваются как элементы кольца $K[x]$. Это доказывается точно так же, как в теореме 4.5.7, и далее мы получаем

$$(x-1)^{P^i (N/P)^j} \equiv A^{ik + j(1-k)} x - 1 \pmod{x^r - b}.$$

Если $E$ — порядок элемента $x-1$ в кольце $K[x]/(x^r - b)$, то $E \geqslant 2^r - 1$ в силу тех же причин, что и раньше. Но вновь, как и раньше, существуют различные пары целых чисел $i_1, j_1$ и $i_2, j_2$, такие, что $U_l := P^{i_l}(N/P)^{j_l} \in [1, N^{\sqrt{r}}]$ при $l = 1, 2$ и $U_1 \equiv U_2 \pmod{E}$. Это влечет равенство $U_1 = U_2$, а значит, $n$ является степенью числа $p$ (поскольку $N$ есть степень $n$, а $P$ есть степень $p$).

Читателю предлагается обратить внимание на поразительное сходство теорем 4.3.3 и 4.5.8, в частности, $I, F, g$ в первой теореме отвечают $d, r, b$ во второй соответственно.

Мы можем использовать теорему 4.5.8 в качестве основы для построения быстрого вероятностного алгоритма, который позволял бы осуществить доказательство простоты числа:

**Алгоритм 4.5.9** (квартичный вариант AKS-теста). Нам задано целое число $n > 1$. Этот вероятностный алгоритм позволяет решить, является число $n$ простым или составным, и в случае завершения выдает правильный ответ.

1. [Начальная установка]

Если $n$ есть квадрат или более высокая степень, возвратить «$n$ — составное»;

Найти пару $r, d$ натуральных чисел с минимальным значением $rd^2$, такую, что $r | n^d - 1$ и $d^2 \lg^2 n < r \leqslant (d+1) d^2 \lg^2 n$;

Выбирать случайные нормированные многочлены $f(t) \in \mathbf{Z}_n[t]$ степени $d$ до тех пор, пока либо $n$ не будет объявлено составным, либо не найдется такого многочлена $f(t)$, что $t^{n^d} \equiv t \pmod{f(t)}$, и многочлен $t^{n^{d/q}} - t$ взаимно прост с $f(t)$ для каждого простого $q | d$;

Выбирать случайные многочлены $b(t) \in \mathbf{Z}_n[t]$ степени меньше $d$ до тех пор, пока либо $n$ не будет объявлено составным, либо не найдется такого многочлена $b(t)$, что $b(t)^{n^d - 1} \equiv 1 \pmod{f(t)}$, и многочлен $b(t)^{(n^d-1)/q} - 1$ взаимно прост с $f(t)$ для каждого простого $q | r$.

2. [Биномиальное сравнение]

Если $(x-1)^{n^d} \not\equiv x^{n^d} - 1 \pmod{x^r - b(t), f(t), n}$, возвратить «$n$ — составное»;

Возвратить «$n$ — простое»;

Следует сделать ряд замечаний. Поиск значений $d, r$ может проводиться детерминированно, и, как обсуждалось ранее, теорема 4.3.5 гарантирует скорый успех. При первом подборе ищется многочлен $f(t)$, обладающий рядом свойств. Использование алгоритма 4.3.2 при попытке доказать взаимную простоту может привести к доказательству того, что $n$ — составное, если оно в самом деле является таковым. Если $n$ — простое, то алгоритм 4.3.2 не объявляет $n$ составным, и у нас будет многочлен $f$ с нужными свойствами, как только выбранный многочлен окажется неприводимым; см. алгоритм 2.2.10. В предположении, что $n$ — простое, и многочлен $f(t)$ неприводим по модулю $n$, кольцо $\mathbf{Z}_n[t]/(f(t))$ является конечным полем, и поиск $b(t)$ увенчается успехом, как только будет найден образующий элемент мультипликативной группы этого конечного поля, и возможно, даже раньше. Снова заметим, что если $n$ — составное, алгоритм 4.3.2 может обнаружить этот факт.

Если $n$ — простое, ожидаемое время работы каждого пункта на стадии [Начальная установка] мало по сравнению со временем выполнения единственного вычисления на стадии [Биномиальное сравнение], причем это время оценивается как $\tilde{O}(rd^2 \ln^2 n)$. При $d$, ограниченном величиной $(\ln \ln n)^{O(\ln \ln \ln \ln n)}$, общая ожидаемая сложность есть $(\ln n)^4 (\ln \ln n)^{O(\ln \ln \ln \ln n)}$. Это выражение не есть в точности $\tilde{O}(\ln^4 n)$, но оно имеет вид $(\ln n)^{4+o(1)}$. По этой причине Бернштейн называет время работы этого алгоритма «в сущности» квартичным.

Если читателя интересует практическое применение круга идей Агравала—Кайала—Саксены, относящихся к проверке на простоту, то на современном этапе их развития следует начинать с алгоритма 4.5.9. И поскольку наиболее предпочтительным случаем этого алгоритма является случай $d = 1$, вероятно, лучше всего сосредоточиться сначала на этом случае и выяснить, можно ли доказать простоту чисел, способных конкурировать с числами других алгоритмов.

Читателю, планирующему реализовать алгоритм AKS, могут оказаться полезными следующие замечания. Пытаетесь ли Вы реализовать первоначальный вариант AKS — алгоритм 4.5.1, или один из его более поздних вариантов, в любом случае Вас могут заинтересовать многие алгоритмы из нашей книги. Например, умножение с двоичным сегментированием, алгоритм 9.6.1, — отличный кандидат для вычисления произведений многочленов с модулем, сводящий вычисление такого произведения к одному умножению больших целых чисел. Существует также возможность организации полностью параллельных вычислений для ключевых степеней многочлена в некоторых вариантах алгоритма AKS. Работа [Crandall and Papadopoulos 2003] из списка литературы позволяет взглянуть на проблему глазами разработчика, и большинство ее концепций применимо ко всем вариантам AKS. В этом исследовании для прямолинейного алгоритма 4.5.1 установлено эмпирическое правило: с помощью надлежащих быстрых алгоритмов можно доказать простоту числа $p$ примерно за

$$T(p) \approx 1000 \ln^6 p$$

операций микропроцессора для значений $p$, поддающихся изучению. Это эмпирический результат из реальной жизни, который согласуется с оценками сложности, приведенными в тексте. Так, например, простое число Мерсенна $p = 2^{31} - 1$ требует порядка $10^{11}$ операций (и тем самым порядка минуты на современном ПК), если применить к нему этот простейший AKS-подход. Заметим, что сложность в операциях $T$ возрастает примерно на два порядка величины при удвоении числа битов в $p$. После этих измерений для простейшего варианта AKS рассуждения, связанные с реализацией, появились в работе [Bernstein 2003], в результате чего мы пришли к вышеупомянутому «в сущности» квартичному времени работы, что позволяет решать вопрос о простоте чисел, состоящих из нескольких сотен десятичных знаков, за время порядка одного дня.

## 4.6. Упражнения

**4.1.** Показать, что для простого числа $n$, большего, чем 200560490131, количество первообразных корней по модулю $n$ больше, чем $(n-1)/(2\ln\ln n)$. Можно использовать следующий план:

(1) Количество первообразных корней по модулю $n$ равно $\varphi(n-1)$.

(2) Если произведение $P$ всех простых чисел $p \leqslant T$ таково, что $P \geqslant m$, тогда

$$\frac{\varphi(m)}{m} \geqslant \prod_{p \leqslant T} \left(1 - \frac{1}{p}\right).$$

Используйте идеи, похожие на эту, чтобы доказать неравенство $200560490131 < n < 5.6 \cdot 10^{12}$.

(3) Завершите доказательство при помощи следующей оценки (см. [Rosser and Schoenfeld 1962]):

$$\frac{m}{\varphi(m)} < e^{\gamma} \ln\ln m + \frac{2.5}{\ln\ln m} \quad \text{для } m > 223092870.$$

**4.2.** Предположим, что условие (4.1) заменено на «для любого простого числа $q|n-1$ существует целое число $a_q$, такое что $a_q^{n-1} \equiv 1 \pmod{n}$ и $a_q^{(n-1)/q} \not\equiv 1 \pmod{n}$». Покажите, что и в этом случае число $n$ будет простым.

**4.3.** Предположим, что нам дано простое число $n$ и полное разложение на множители числа $n-1$. Мы пытаемся использовать упр. 4.2, чтобы доказать простоту числа $n$, выбирая числа $a_q$ случайным образом. Однако, если $q_1, q_2, \ldots, q_k$ — различные простые числа, делящие число $n-1$, и если числа $a_{q_1}, \ldots, a_{q_{i-1}}$ уже найдены, то перед тем как случайным образом выбирать следующее число $a_{q_i}$, мы проверяем, не подходит ли одно из уже найденных чисел $a_q$ для простого делителя $q_i$. Покажите, что существует число $c$, не зависящее от исходного числа $n$, такое что количество различных чисел $a_q$ не превосходит этого числа $c$.

**4.4.** Предположим, что элементы $b_1, b_2, \ldots$ выбираются случайным образом, равномерно и независимо друг от друга, из мультипликативной группы $\mathbf{Z}_n^*$. Пусть $g(n)$ равно математическому ожиданию наименьшего числа $g$, такого что подгруппа, порожденная элементами $b_1, \ldots, b_g$ совпадает со всей группой $\mathbf{Z}_n^*$. В духе упр. 4.3 покажите, что $g(n) < 3$ для всех простых чисел $n$. Что можно сказать про число $g(n)$ в общем случае, когда число $n$ не обязательно является простым?

**4.5.** Покажите, что тест Пепена работает с числом 5 вместо числа 3 для всех чисел Ферма, больших 5. (Именно такая формулировка теста была дана Пепеном в оригинале.)

**4.6.** В 1999 г. группа исследователей (Крэндалл, Майер, Пападопулос) сделали и проверили цепочку возведений в квадрат Пепена для двадцать четвертого числа Ферма $F_{24}$. Это число является составным. Эти вычисления можно назвать самыми масштабными из предпринятых до 2000 г. для получения однобитового ответа (т. е. простое или составное), см. [Crandall et al. 1999]. (Позднее Персиваль определил, что квадриллионный бит двоичного разложения числа $\pi$ равен 0; указанное вычисление было еще более масштабным, чем решение вопроса о $F_{24}$.) Число $F_{24}$ можно назвать «чисто составным» числом Ферма (доказано, что число является составным, но не известен ни один делитель этого числа). См. упр. 1.82, в котором также упоминаются чисто составные числа.

На момент выхода книги наименьшим числом Ферма неизвестного характера является число $F_{33}$. Оцените, сколько арифметических операций необходимо проделать, чтобы осуществить тест Пепена для числа $F_{33}$. Каково это количество операций по сравнению со всем количеством операций, проделанных на компьютере для всех задач за все время до 2000 г.? В каком году число $F_{33}$ будет проверено при помощи теста Пепена? В таблице 1.3 приведена информация о нескольких первых числах Ферма.

Проанализируйте следующие задачи:

(1) Возможность распараллеливания алгоритма Пепена (никто не знает как распараллелить весь тест Пепена эффективным способом, но можно рас-

параллелить одну операцию возведения в квадрат, вычисляя каждый элемент свертки на параллельных машинах и пользуясь китайской теоремой об остатках).

(2) Проблема *доказанности* характера числа $F_n$ после завершения теста Пепена. Это, несомненно, является проблемой, потому что компьютер может совершить ошибку, либо наведенную космическим или иным излучением (аппаратная ошибка), либо из-за логических ошибок в программе (программная ошибка). Кстати, аппаратные ошибки *в самом деле* случаются! Физики говорят, что компьютер живет в энтропийной ванне, и существует ненулевая вероятность ошибки. Для избежания программных ошибок важно иметь две различные программы (лучше даже разных авторов), запущенные на различных компьютерах, которые проверяли бы друг друга.

В связи с последним вопросом можно использовать метод «фронта волны». Суть этого метода состоит в том, что самый быстрый компьютер проделывает весь тест Пепена, в котором последовательные возведения в квадрат играют роль фронта волны, а остальные (более слабые) компьютеры независимо проверяют вычисления быстрого компьютера на различных стадиях теста. Например, фронтовой компьютер предоставил миллионный, двухмиллионный, трехмиллионный и четырехмиллионный квадрат числа 3; т. е. вычислил степени

$$3^{2^{1000000}},\ 3^{2^{2000000}},\ 3^{2^{3000000}},\ 3^{2^{4000000}}$$

по модулю $F_n$. Каждый из слабых компьютеров берет свой остаток и производит всего миллион возведений в квадрат (получая таким образом следующую вычисленную степень), затем сравнивает с результатом фронтового компьютера.

**4.7.** Докажите следующие теоремы Суямы (см. [Williams 1998]):

(1) Пусть $k$ — нечетное число, и число $N = k2^n + 1$ делит число Ферма $F_m$. Докажите, что если $N < (3 \cdot 2^{m+2} + 1)^2$, то число $N$ является простым.

(2) Предположим, что число Ферма $F_m$ разложено на множители $FR$, причем мы знаем полное разложение на простые множители числа $F$, и ничего не знаем об остатке $R$. Возможно, число $R$ является простым и, таким образом, мы имеем полное разложение на множители числа $F_m$. Если число $R$ является составным, то следующий простой тест часто помогает установить этот факт. Пусть $r_1 = 3^{F_m - 1} \bmod F_m$ и $r_2 = 3^{F-1} \bmod F_m$. Если $r_1 \not\equiv r_2 \pmod R$, то остаток $R$ является составным. (Этот результат полезен, так как он заменяет множество арифметических операций по модулю $R$ на арифметические операции по модулю $F_m$. Как показывает алгоритм 9.2.13, операции деления на числа Ферма $F_m$ особенно просты.)

**4.8.** Для некоторых множителей больших чисел Ферма используется следующая схема вычислений, похожая на результат Суямы в упр. 4.7, см. [Crandall et al. 1999]. Пусть число $F_n$ было проверено при помощи теста Пепена, и у нас есть последний остаток теста Пепена, а именно,

$$r = 3^{(F_n - 1)/2} \bmod F_n.$$

Пусть мы обнаружили делитель $f$ числа $F_n$, т. е. мы можем представить число $F_n$ в виде

$$F_n = fG.$$

Обозначим

$$x = 3^{f-1} \bmod F_n.$$

Докажите, что из условия

$$\text{НОД}\,(r^2 - x, G) = 1$$

следует, что сомножитель $G$ не является ни простым, ни степенью простого числа. Как и в упр. 4.7, причиной именно такой последовательности операций служат относительно быстрые (по модулю числа $F_n$) операции, нежели вычисления по модулю числа $G$ при взятии НОД. Исходя из существования таких схем, возникает необходимость запасать полученные остатки после теста Пепена, так как они потом могут еще понадобиться!

**4.9.** Существует интересный способ нахождения достаточно больших простых чисел, имеющих вид Прота $p = k2^n + 1$. Докажите следующую теорему Суямы [Williams 1998]: если число $p$, имеющее указанный выше вид, делит некоторое число Ферма $F_m$ и $k2^{n-m-2} < 9 \cdot 2^{m+2} + 6$, то число $p$ является простым.

**4.10.** Докажите следующую теорему Прота: если $n > 1, 2^k | n - 1, 2^k > \sqrt{n}$ и $a^{(n-1)/2} \equiv -1 \pmod{n}$ для некоторого целого числа $a$, то число $n$ является простым.

**4.11.** В алгоритме, основанном на теореме 4.1.6, требуются целочисленные корни (если таковые существуют) кубического многочлена с целыми коэффициентами. Для начала покажите, как эффективно вычислить их с помощью метода Ньютона или стратегии «разделяй и властвуй». Руководством может послужить простой алгоритм 9.2.11. Исследуйте возможность быстрого нахождения предполагаемых целочисленных корней для многочленов более высокой степени.

Дадим указание, относящееся к более простому случаю многочленов вида $x^k - a$. С целью обобщения алгоритма 9.2.11 на случай нахождения целочисленных корней $k$-й степени, скажем $\lfloor N^{1/k} \rfloor$, рассмотрите следующие изменения:

- на шаге [Начальная установка] сделайте замену: $B(N)/2 \rightarrow B(N)/k$;
- на шаге [Шаг итерации] проводите итерацию

$$y = \lfloor ((k-1)x + \lfloor N/x^{k-1} \rfloor)/k \rfloor$$

или используйте аналогичную формулу для подобной редукции.

**4.12.** Докажите теорему 4.2.4.

**4.13.** Пусть частичная факторизация (4.2) была найдена пробным делением числа $n - 1$ вплоть до границы $B$, тогда мы знаем, что все простые делители числа $R$ больше этой границы $B$. Покажите, что если число $a$ удовлетворяет условию (4.3), и НОД $(a^F - 1, n) = 1$, то все простые делители числа $n$ превосходят число $BF$. В частности, если $BF \geqslant n^{1/2}$, то число $n$ является простым.

**4.14.** Предположим, что дополнительно к условиям теоремы 4.2.10 мы знаем также, что все простые делители числа $R_1R_2$ превосходят число $B$ (где $n-1=F_1R_1$, $n+1=F_2R_2$). Также предположим, что существует число $a_1$, такое что $a_1^{n-1}\equiv 1 \pmod n$, $\text{НОД}(a_1^{F_1}-1,n)=1$, и существуют многочлен $f$ и соответствующее число $\Delta$ (см. (4.12)), такие что $\text{НОД}(n,2b)=1$, $\left(\frac{\Delta}{n}\right)=-1$, $U_{n+1}\equiv 0 \pmod n$, $\text{НОД}(U_{F_2},n)=1$. Пусть число $F$ обозначает наименьшее общее кратное чисел $F_1,F_2$. Покажите, что если остаток $n \bmod F$ не является собственным делителем числа $n$, и $BF>\sqrt{n}$, то число $n$ является простым.

**4.15.** Докажите теорему 4.2.9.

**4.16.** Методами, использующимися в упр. 4.1, докажите следующий факт: если число $n>892271479$ является простым, число $N$ равно математическому ожиданию количества случайных выборов пар чисел $a,b\in\{0,1,\ldots,n-1\}$ ($a$ и $b$ одновременно не равны 0), пока мы не получим $r_f(n)=n+1$ (многочлен $f$ взят из определения (4.12)). Тогда $N<4\ln\ln n$.

**4.17.** Докажите, что число $n=700001$ является простым, сначала используя разложение на множители числа $n-1$, затем используя разложение на множители числа $n+1$.

**4.18.** Покажите, как алгоритм Копперсмита, упомянутый в конце разд. 4.2.3, может быть использован для улучшения $(n-1)$-теста, $(n+1)$-теста, объединенного $(n^2-1)$-теста, теста на простоту при помощи конечного поля и теста, основанного на суммах Гаусса.

**4.19.** Покажите, что любой идеал кольца $\mathbf{Z}_n[x]$ является главным (т. е. состоит из элементов, кратных одному фиксированному многочлену) тогда и только тогда, когда число $n$ является простым.

**4.20.** Пусть $q$ — нечетное простое число. Используя обозначения разд. 4.4.1 и определение 2.3.6, покажите, что для целого числа $m$, не делящегося на число $q$, имеют место равенства $\chi_{2,q}(m)=\left(\frac{m}{q}\right)$ и $G(2,q)=G(1,q)$.

**4.21.** Пусть $q$ — нечетное простое число, и пусть $\chi$ — неглавный характер по модулю $q$. Обобщите доказательство леммы 4.4.1 с тем чтобы показать, что $|\tau(\chi)|^2=q$. Иными словами, лемма 4.4.1 сформулирована для характера по простому модулю, имеющего простой порядок, в то время как это упражнение состоит в ее обобщении на любой характер по простому модулю при условии, что его порядок превосходит 1. Докажите еще более общее утверждение: $|\tau(\chi)|^2=q$ для любого примитивного характера $\chi$ по модулю $q$, независимо от того, является число $q$ простым или нет.

**4.22.** Предположим, что число $n$ прошло шаги [Приготовления] и [Вычисление для проверки свойства вероятного простого числа] алгоритма 4.4.5, и для любого простого числа $p|I$ выполняется либо равенство $w(p)=1$, либо некоторое число $l(p,q)\neq 0$. Следовательно, в шаге [Проверка взаимной простоты] никакие вычисления не нужны. Покажите, что в качестве числа $l$ в шаге [Поиск делителя] можно взять число $n$, т. е. вычисления по китайской теореме об остатках на этом шаге могут быть пропущены.

**4.23.** В обозначениях определения 4.4.4 покажите, что если $\alpha$ является единицей в кольце $\mathbf{Z}_n[\zeta_p, \zeta_q]$, то НОД $(n, c(\alpha)) = 1$. Покажите, что обратное неверно.

**4.24.** Если $q$ — простое, и $\chi$ — характер по модулю $q$ порядка $q-1$, то докажите, что $\chi$ является взаимно однозначным на классах вычетов по модулю $q$. Докажите и обратное утверждение.

**4.25.** Для целого числа $n$, не меньшего 2, покажите, что многочлены $(x+1)^n$ и $x^n+1$ равны в кольце $\mathbf{Z}_n[x]$ тогда и только тогда, когда $n$ — простое. В общем случае, покажите, что если НОД $(a, n) = 1$, то $(x+a)^n = x^n + a$ в кольце $\mathbf{Z}_n[x]$ тогда и только тогда, когда $n$ — простое.

**4.26.** Используя теорему4.5.2, докажите, что алгоритм 4.5.1 корректно решает, является число $n$ простым или составным.

**4.27.** Покажите, что множество $\overline{G}$ в доказательстве теоремы 4.5.2 есть объединение множества $\{0\}$ и циклической мультипликативной группы.

**4.28.** Тем же методом, что и в упр. 3.19, покажите, что если $a^n \equiv a \pmod{n}$ для каждого положительного целого числа $a$, меньшего $\ln^2 n$, то число $n$ — бесквадратное. Далее, покажите, что из AKS-сравнения $(x+a)^n \equiv x^n + a \pmod{x^r - 1, n}$ следует, что $(a+1)^n \equiv a+1 \pmod{n}$. Сделайте вывод, что из предположений теоремы 4.5.2 следует, что если $n$ делится на простое число, большее, чем $\sqrt{\varphi(r)} \lg n$, то $n$ равно этому простому числу. Используйте этот факт для обоснования укороченной версии алгоритма 4.5.1, в которой стадия [Проверка на степень] может быть полностью опущена.

**4.29.** Покажите, что если $n \equiv \pm 3 \pmod{8}$, то значение $r$ на стадии [Начальная установка] в алгоритме 4.5.1 ограничено сверху величиной $8 \lg^2 n$. Указание: покажите, что если $r_2$ есть наименьшая степень числа 2, такая, что порядок элемента $n$ в группе $\mathbf{Z}_{r_2}^*$ превосходит $\lg^2 n$, то $r_2 < 8 \lg^2 n$.

**4.30.** Используя соответствующее обобщение идеи, предложенной в упр. 4.29, и теорему 1.4.7, покажите, что значение $r$ на стадии [Начальная установка] в алгоритме 4.5.1 ограничено сверху величиной $\lg^2 n \lg\lg n$ для всех, за исключением, возможно, $o(\pi(x))$ простых чисел $n \leqslant x$. Сделайте вывод, что алгоритм 4.5.1 имеет время работы $\tilde{O}(\ln^6 n)$ для почти всех простых чисел $n$ в том смысле, что число исключительных простых чисел $n \leqslant x$ есть $o(\pi(x))$.

**4.31.** Докажите обращение леммы 4.5.5; а именно, предположив, что $r, p$ — различные простые, $d | r-1$ и многочлен $f_{r,d}(x)$ неприводим по модулю $p$, докажите, что порядок элемента $p^{(r-1)/d}$ по модулю $r$ равен $d$.

**4.32.** Пусть $r_1, r_2, \ldots, r_k$ — простые числа и $d_1, d_2, \ldots, d_k$ положительны и попарно взаимно просты, причем $d_i | r_i - 1$ для каждого $i$. Пусть $f(x)$ — минимальный многочлен для $\eta_{r_1,d_1}\eta_{r_2,d_2}\ldots\eta_{r_k,d_k}$ над $\mathbf{Q}$. Покажите, что для простых чисел $p$, отличных от каждого из $r_i$, многочлен $f(x)$ неприводим по модулю $p$ тогда и только тогда, когда порядок каждого из $p^{(r_i-1)/d_i}$ по модулю $r_i$ равен $d_i$.

**4.33.** В тексте главы мы лишь наметили схему доказательства теоремы 4.5.8. Приведите полное доказательство.

## 4.7. Проблемы для исследования

**4.34.** Разработайте алгоритм для точного распознавания простоты (или непростоты) произвольного числа $n \in [2, \ldots, x]$ для как можно большей границы $x$, действуя по следующей схеме.

Используйте *вероятностный* тест на простоту и составьте (он, скорее всего, будет минимальным) список *исключений*. Или используйте небольшую комбинацию простых тестов, не имеющую исключений до границы $x$. Например, в книге [Jaeschke 1993] показано, что ни одно составное число, меньшее 341550071728321, не проходит сильный тест на вероятное простое число (алгоритм 3.5.2) одновременно для всех простых оснований, меньших 20.

**4.35.** Рассматривая диофантово уравнение

$$n^k - 4^m = 1,$$

докажите, что ни одно число Ферма не может быть степенью $n^k$, $k > 1$. Это уже известный факт, но неясным остается следующее: обязательно ли числа Ферма бесквадратны? Покажите также, что ни одно из чисел Мерсенна $M_p$ не является нетривиальной степенью какого-либо числа. Обязательно ли числа Мерсенна бесквадратны?

**4.36.** Вспомните функцию $M(p)$, определяемую в разд. 4.1.3 как количество умножений, необходимое для доказательства простоты числа $p$ при помощи прохождения дерева Люка для числа $p$. Докажите или опровергните следующее утверждение: для всех простых чисел $p$ имеем $M(p) = O(\lg p)$.

**4.37.** (Бродхёрст). Последовательность Фибоначчи $(u_n)$, определенная в упр. 2.5, порождает при определенных $n$ некоторые удивительные простые числа. Разработайте эффективную схему проверки простоты для чисел Фибоначчи, возможно, используя при этом общедоступные вспомогательные средства.

К слову, согласно Д. Бродхёрсту, вопрос о простоте $u_n$ строго разрешен для *всех* $n$ вплоть до $n = 35999$ включительно (кстати, $u_{35999}$ является простым). Кроме того, известно, что $u_{81839}$ является простым, хотя все еще необходимы вычисления для решения вопроса о двух предполагаемых (вероятных) простых числах, таких как $u_n$ при $n \in \{50833, 104911\}$, что позволит распространить результаты до $n = 104911$.

**4.38.** При условии, что задано натуральное $n$, отличное от квадрата, покажите, что существует простое число $r$ с условиями $1 + \lg^2 n < r = O(\ln^2 n)$, такое, что $n$ есть первообразный корень для $r$. Если Вы готовы предположить справедливость обобщенной гипотезы Римана (GRH), Вам может оказаться полезным обсуждение гипотезы Артина в книге [Hooley 1976][3].

**4.39.** В упр. 4.28 мы увидели, что проверку на степень можно исключить из алгоритма 4.5.1. Можно ли также исключить проверку на степень из алгоритма 4.5.9? Следует ли из предположений теоремы 4.5.6, что $n$ — бесквадратное?

[3] Имеется перевод: [Хооли 1983]. — *Прим. ред.*

Глава 5

# ЭКСПОНЕНЦИАЛЬНЫЕ АЛГОРИТМЫ РАЗЛОЖЕНИЯ НА МНОЖИТЕЛИ

На протяжении почти всей многовековой практики разложения чисел на множители все алгоритмы были экспоненциальными. Это означает, что время их работы в худшем случае является фиксированной положительной степенью разлагаемого числа. В начале 70-х годов XX в. стали появляться субэкспоненциальные алгоритмы. Эти методы обсуждаются в гл. 6, время их работы по разложению числа $n$ на множители ограничено выражением вида $n^{o(1)}$. Не удивляйтесь тому, что настоящая глава также включена в книгу, на это есть несколько причин:

(1) Для разложения на множители небольших чисел обычно используются экспоненциальные алгоритмы. В частности, в некоторых субэкспоненциальных методах необходимо раскладывать на множители маленькие вспомогательные числа, что делается в соответствующей процедуре при помощи экспоненциального алгоритма.

(2) В некоторых случаях экспоненциальный алгоритм является прямым предком субэкспоненциального. Например, субэкспоненциальный метод эллиптических кривых появился из экспоненциального $(p-1)$-метода. Можно также считать, что экспоненциальные алгоритмы являются материалом для дальнейших исследований, подобно тому как различные дикие сорта сельскохозяйственных культур могут послужить генетическим материалом для выведения новых сортов.

(3) На самом деле до сих пор самые быстрые строго проанализированные *детерминированные* алгоритмы разложения чисел на множители являются экспоненциальными.

(4) Некоторые алгоритмы разложения на множители, как экспоненциальные, так и субэкспоненциальные, являются основой для аналогичных алгоритмов вычисления дискретных логарифмов. Для некоторых алгебраических групп нам известны *только* экспоненциальные алгоритмы для вычисления дискретных логарифмов.

(5) Многие экспоненциальные алгоритмы являются произведениями искусства.

Мы надеемся, читатель согласится с тем, что эта глава заслуживает включения в книгу!

# 5.1. Метод квадратов

Старый подход к разложению числа на множители заключается в представлении его в виде разности двух несоседних квадратов. Давайте обсудим эту тему подробнее.

## 5.1.1. Метод Ферма

Если мы сможем представить число $n$ в виде $a^2 - b^2$, где $a, b$ — неотрицательные целые числа, то мы сразу можем разложить число $n$ на два множителя $(a+b)(a-b)$. Если $a - b > 1$, то мы получим нетривиальное разложение на множители. Более того, каждое разложение на множители любого нечетного числа $n$ имеет именно такой вид. В самом деле, если $n$ — нечетное число и $n = uv$, где $u, v$ — натуральные числа, то $n = a^2 - b^2$, где $a = \frac{1}{2}(u+v)$ и $b = \frac{1}{2}|u-v|$.

Для нечетных чисел $n$, которые являются произведениями двух близких целых чисел, найти подходящие числа $a, b$ довольно просто. Например, рассмотрим $n = 8051$. Первый полный квадрат, больший $n$, это $8100 = 90^2$, разница с числом $n$ составляет $49 = 7^2$. Следовательно, $8051 = (90+7)(90-7) = 97 \cdot 83$.

Оформим этот метод в виде алгоритма. Мы берем пробные значения числа $a$ из последовательности $\lceil\sqrt{n}\rceil, \lceil\sqrt{n}\rceil + 1, \ldots$ и проверяем, является ли число $a^2 - n$ полным квадратом. Если $a^2 - n = b^2$, то мы получили разложение на множители $n = a^2 - b^2 = (a+b)(a-b)$. Для составных нечетных чисел $n$ эта процедура остановится на нетривиальном разложении на множители при $a \leqslant \lfloor (n+9)/6 \rfloor$. В худшем случае, когда $n = 3p$, где $p$ — простое число, единственным числом $a$, которое даст нам нетривиальное разложение, будет $(n+9)/6$ (соответствующим числом $b$ будет $(n-9)/6$).

**Алгоритм 5.1.1** (метод Ферма). **Дано нечетное число $n > 1$. Алгоритм либо выдает нетривиальный делитель числа $n$, либо доказывает, что число $n$ является простым.**

1. [Основной цикл]
   for$\left(\lceil\sqrt{n}\rceil \leqslant a \leqslant (n+9)/6\right)$ {
   // Далее применяем алгоритм 9.2.11.
   if$\left(b = \sqrt{a^2 - n}\right.$ является целым числом$\left.\right)$ return $a - b$;
   }
   return «число $n$ является простым»;

Очевидно, что в худшем случае алгоритм 5.1.1 гораздо более медленный, чем пробное деление. Но худшие случаи для алгоритма 5.1.1 на самом деле являются самыми простыми случаями для пробного деления, и наоборот. Поэтому можно попытаться объединить эти два метода.

Существуют различные трюки, которые можно использовать для ускорения метода Ферма. При помощи сравнений можно понять, что для $a$ из некоторых классов вычетов число $a^2 - n$ не может быть полным квадратом. Например, если $n \equiv 1 \pmod 4$, то число $a$ не может быть четным, или, если $n \equiv 2 \pmod 3$, то число $a$ должно быть кратно 3.

Можно также использовать дополнительные множители. Как мы видели, если число $n$ является произведением двух близких целых чисел, то алгоритм 5.1.1 найдет разложение на множители довольно быстро. Даже если число $n$ не обладает таким свойством, возможно, им обладает число $kn$ (оно является произведением двух близких целых чисел). Тогда мы можем взять $\text{НОД}(a \pm b, n)$ для получения разложения на множители числа $n$. Например, рассмотрим $n = 2581$. Алгоритм 5.1.1 начнет работу с числа $a = 51$ и закончит на девятом выборе $a = 59$. В итоге мы получим, что $59^2 - 2581 = 900 = 30^2$ и $2581 = 89 \cdot 29$. (Учитывая, что $n \equiv 1 \pmod 4, n \equiv 1 \pmod 3$, мы знаем, что число $a$ является нечетным и не кратно 3. Если мы будем использовать эту информацию, то число 59 будет всего лишь третьим выбором.) Попробуем использовать алгоритм 5.1.1 для числа $3n = 7743$. Мы остановимся на первом выборе числа $a$, а именно, $a = 88$ (и $b = 1$). Следовательно, $3n = 89 \cdot 87$, и, замечая, что $89 = \text{НОД}(89, n)$ и $29 = \text{НОД}(87, n)$, мы получаем искомое разложение на множители.

## 5.1.2. Метод Лемана

Но как мы узнаем, что в примере, описанном выше, надо пробовать множитель 3? Следующий метод Р. Лемана формализует поиск такого множителя.

**Алгоритм 5.1.2** (метод Лемана). Дано целое число $n > 21$. Этот алгоритм выдает нетривиальный делитель числа $n$ или доказывает, что число $n$ является простым.

1. [Пробное деление]
   Проверяем, имеет ли число $n$ нетривиальный делитель $d \leqslant n^{1/3}$. Если имеет, то возвратить ответ $d$;
2. [Цикл]
   for$\left(1 \leqslant k \leqslant \lceil n^{1/3} \rceil\right)$ {
   for$\left(\lceil 2\sqrt{kn} \rceil \leqslant a \leqslant \lfloor 2\sqrt{kn} + n^{1/6}/(4\sqrt{k}) \rfloor\right)$ {
   if$\left(b = \sqrt{a^2 - 4kn}\right.$ является целым числом$\left.\right)$ return $\text{НОД}(a + b, n)$;
   // При помощи алгоритма 9.2.11.
   }
   }
   return «число $n$ является простым»;

Предполагая, что этот алгоритм корректен, можно легко оценить время его работы. Шаг [Пробное деление] требует $O(n^{1/3})$ операций, и если шаг [Цикл] выполняется, то он требует не более

$$\sum_{k=1}^{\lceil n^{1/3} \rceil} \left( \frac{n^{1/6}}{4\sqrt{k}} + 1 \right) = O(n^{1/3})$$

вызовов алгоритма 9.2.11, при этом каждый вызов требует $O(\ln \ln n)$ операций. Следовательно, в сумме алгоритм 5.1.2 требует в худшем случае $O(n^{1/3} \ln \ln n)$

арифметических операций с целыми числами размера $n$. Докажем теперь корректность метода Лемана.

**Теорема 5.1.3.** *Метод Лемана* (*алгоритм* 5.1.2) *корректен.*

ДОКАЗАТЕЛЬСТВО. Мы можем предположить, что число $n$ не было разложено на множители на шаге [Пробное деление]. Если число $n$ не является простым, то оно является произведением двух простых чисел, больших $n^{1/3}$. Т. е. $n = pq$, где $p, q$ — простые числа и $n^{1/3} < p \leqslant q$. Мы утверждаем, что существует число $k \leqslant \lceil n^{1/3} \rceil$, имеющее разложение на множители $k = uv$, где $u, v$ — натуральные числа и

$$|uq - vp| < n^{1/3}.$$

В самом деле, по известному результату (см. [Hardy and Wright 1979, теорема 36]), для любой границы $B > 1$ существуют натуральные числа $u, v$, такие что $v \leqslant B$ и $|\frac{u}{v} - \frac{p}{q}| < \frac{1}{vB}$. Применим этот результат для границы $B = n^{1/6}\sqrt{q/p}$. Тогда

$$|uq - vp| < \frac{q}{n^{1/6}\sqrt{q/p}} = n^{1/3}.$$

Осталось показать, что $k = uv \leqslant \lceil n^{1/3} \rceil$. Так как $\frac{u}{v} < \frac{p}{q} + \frac{1}{vB}$ и $v \leqslant B$, мы имеем

$$k = uv = \frac{u}{v}v^2 < \frac{p}{q}v^2 + \frac{v}{B} \leqslant \frac{p}{q} \cdot \frac{q}{p}n^{1/3} + 1 = n^{1/3} + 1,$$

таким образом, наше утверждение доказано.

Пусть $k, u, v$ — числа, описанные выше, положим $a = uq + vp$, $b = |uq - vp|$. Тогда $4kn = a^2 - b^2$. Покажем, что $2\sqrt{kn} \leqslant a < 2\sqrt{kn} + \frac{n^{1/6}}{4\sqrt{k}}$. Так как $uq \cdot vp = kn$, мы имеем $a = uq + vp \geqslant 2\sqrt{kn}$. Положим $a = 2\sqrt{kn} + E$. Тогда

$$4kn + 4E\sqrt{kn} \leqslant \left(2\sqrt{kn} + E\right)^2 = a^2 = 4kn + b^2 < 4kn + n^{2/3},$$

следовательно, $4E\sqrt{kn} < n^{2/3}$ и $E < \frac{n^{1/6}}{4\sqrt{k}}$, как и утверждалось в алгоритме.

Наконец, мы покажем, что если $a, b$ — числа, полученные на шаге [Цикл], то число НОД $(a + b, n)$ является нетривиальным делителем числа $n$. Так как число $n$ делит $(a + b)(a - b)$, достаточно показать, что $a + b < n$. Имеем

$$a + b < 2\sqrt{kn} + \frac{n^{1/6}}{4\sqrt{k}} + n^{1/3} < 2\sqrt{(n^{1/3} + 1)n} + \frac{n^{1/6}}{4\sqrt{n^{1/3} + 1}} + n^{1/3} < n,$$

последнее неравенство выполняется при $n \geqslant 21$. □

Существуют различные способы ускорения метода Лемана, например, вначале проверяются значения числа $k$, имеющие много делителей. За подробностями мы отсылаем читателя к работе [Lehman 1974].

## 5.1.3. Отсев делителей

В методе Ферма мы искали целые числа $a$ так, чтобы число $a^2 - n$ являлось полным квадратом. Один из возможных путей состоит в использовании многих

значений числа $a$, для которых число $a^2 - n$ не является полным квадратом. Например, предположим, что $a^2 - n = 17$. Говорит ли нам это что-нибудь о числе $n$? Да, говорит. Если число $p$ является простым делителем числа $n$, то $a^2 \equiv 17 \pmod{p}$. Поэтому если $p \neq 17$, то число $p$ обязано лежать в одном из классов вычетов $\pm 1, \pm 2, \pm 4, \pm 8 \pmod{17}$. Таким образом мы разом исключили из рассмотрения в качестве делителей числа $n$ половину всех простых чисел. При других значениях числа $a$ мы можем аналогично отсечь другие классы вычетов простых делителей числа $n$. Есть надежда, что мы сможем получить столько информации о классах вычетов, в которых должны лежать простые делители числа $n$, что эти делители будут полностью определены и их можно будет легко найти.

Неприятная особенность этих вычислений состоит в экспоненциальном росте их сложности. Предположим, мы провели эти вычисления для $k$ значений чисел $a$ и получили $k$ значений модулей $m_1, m_2, \ldots, m_k$. И для каждого модуля мы знаем, в каких классах вычетов должны лежать простые делители $p$ числа $n$. Для простоты рассуждений предположим, что все числа $m_i$ являются различными простыми числами, и у нас есть $\frac{1}{2}(m_i - 1)$ возможных классов вычетов $\pmod{m_i}$ для простых делителей числа $n$. Тогда по модулю произведения $M = m_1 m_2 \ldots m_k$ мы будем иметь $2^{-k}(m_1 - 1)(m_2 - 1)\ldots(m_k - 1) = 2^{-k}\varphi(M)$ возможных классов вычетов $(\mathrm{mod}\ M)$. С одной стороны это довольно маленькое число, но с другой стороны оно огромно! Т. е. вероятность того, что произвольное простое число $p$ лежит в одном из этих классов вычетов, равна $2^{-k}$. Поэтому, если число $k$ большое, то мы сильно уменьшаем количество возможных делителей $p$ и уточняем их. Но мы не знаем ни одного быстрого способа нахождения маленьких решений, которые бы одновременно удовлетворяли всем нужным сравнениям, а перебор всех $2^{-k}\varphi(M)$ решений для поиска подходящего — недопустимо большое вычисление. Ранние вычислительные попытки решения этой проблемы использовали оригинальные аппараты с велосипедными цепями, картами и фотоэлектрическими элементами. Существуют также современные специальные компьютеры, созданные для решения этой проблемы. За подробностями по этой проблеме обращайтесь к работе [Williams and Shallit 1993].

## 5.2. Методы Монте-Карло

Существует несколько интересных эвристических методов, использующих определенные детерминированные последовательности, которые рассматриваются как случайные. Хотя эти последовательности могут содержать в себе элемент случайности, по сути они не являются случайными. Но, тем не менее, мы называем обсуждаемые методы методами Монте-Карло. Все методы этого раздела в своей основе принадлежат Дж. Полларду.

### 5.2.1. Ро-метод Полларда для разложения на множители

В 1975 г. Дж. Поллард представил новаторский алгоритм разложения чисел на множители, см. [Pollard 1975]. Рассмотрим случайную функцию $f$, отображающую $\mathcal{S}$ в $\mathcal{S}$, где $\mathcal{S} = \{0, 1, \ldots, l-1\}$. Выберем случайный элемент $s \in \mathcal{S}$ и рассмотрим последовательность

$$s, f(s), f(f(s)), \ldots.$$

Так как функция $f$ принимает значения из конечного набора, то очевидно, что с какого-то момента последовательность повторится и станет циклической. Мы можем изобразить поведение этой последовательности при помощи буквы $\rho$, хвост которой означает доциклическую часть последовательности, и овал — циклическую часть. Какой длины может быть хвост, и какой длины может быть цикл?

Становится очевидно, что здесь имеет место парадокс дней рождения элементарной теории вероятностей, и ожидаемая длина и хвоста, и овала имеет порядок $\sqrt{l}$. Остается вопрос: какое это имеет отношение к разложению чисел на множители?

Возьмем простое число $p$ и положим $\mathcal{S} = \{0, 1, \ldots, p-1\}$. Возьмем конкретную функцию $f$, отображающую $\mathcal{S}$ в $\mathcal{S}$, а именно, $f(x) = x^2 + 1 \bmod p$. Если эта функция «достаточно случайна», то мы можем ожидать, что последовательность итераций $(f^{(i)}(s))$, $i = 0, 1, \ldots$, начинающаяся со случайного числа $s \in \mathcal{S}$, начнет повторяться раньше, чем через $O(\sqrt{p})$ шагов. Т. е. мы ожидаем существование шагов $0 \leqslant j < k = O(\sqrt{p})$, таких, что $f^{(j)}(s) = f^{(k)}(s)$.

Теперь предположим, что мы хотим разложить на множители число $n$, и пусть число $p$ является наименьшим простым делителем числа $n$. Так как мы не знаем число $p$, то мы не можем вычислить последовательность, описанную выше. Однако мы можем вычислить значения функции $F$, определенной как $F(x) = x^2 + 1 \bmod n$. Нам не требуется найти две равные итерации функции $F$, а требуется найти две равные итерации функции $f(x) = x^2 + 1 \bmod p$. Как можно это сделать, зная только $F$? Очевидно, что $f(x) = F(x) \bmod p$. Следовательно, $F^{(j)}(s) \equiv F^{(k)}(s) \pmod p$. Т. е. НОД $\bigl(F^{(j)}(s) - F^{(k)}(s), n\bigr)$ делится на число $p$. Если нам повезет и этот НОД не равен самому числу $n$, то мы получим нетривиальный делитель числа $n$.

В ро-методе Полларда есть еще одна составляющая. Естественно, мы не можем перебирать все пары чисел $j, k$, удовлетворяющие условию $0 \leqslant j < k$, и вычислять НОД$(F^{(j)}(s) - F^{(k)}(s), n)$ для каждой пары. Это может занять больше времени, чем поиск простого делителя $p$ при помощи пробного деления, так как для перебора чисел до границы $B$ надо рассмотреть около $\frac{1}{2}B^2$ пар $j, k$. Причем мы, скорее всего, не найдем нужную пару, пока граница $B$ не будет порядка $\sqrt{p}$. Поэтому нам нужен другой способ поиска подходящей пары чисел $j, k$, отличный от полного перебора. Для этого используют замечательный прием, а именно, метод поиска циклов Флойда. Пусть $l = k - j$, тогда для всех $m \geqslant j$ мы имеем $F^{(m)}(s) \equiv F^{(m+l)}(s) \equiv F^{(m+2l)}(s) \equiv \ldots \pmod p$. Рассмот-

рим эту последовательность при $m = l\lceil j/l \rceil$, первом числе, кратном числу $l$ и превосходящем число $j$. Тогда $F^{(m)}(s) \equiv F^{(2m)}(s) \pmod{p}$, и $m \leqslant k = O(\sqrt{p})$.

Итак, основная идея ро-метода Полларда состоит в вычислении последовательности НОД$(F^{(i)}(s) - F^{(2i)}(s), n)$ для $i = 1, 2, \ldots$, и это вычисление должно закончиться нетривиальным разложением числа $n$ на множители за $O(\sqrt{p})$ шагов, где число $p$ является наименьшим простым делителем числа $n$.

**Алгоритм 5.2.1** (ро-метод Полларда для разложения на множители). Дано составное число $n$. Этот алгоритм пытается найти нетривиальный делитель числа $n$.

1. [Выбрать инициализаторы]
   Выбираем случайное $a \in [1, n-3]$;
   Выбираем случайное $s \in [0, n-1]$;
   $U = V = s$;
   Определяем функцию $F(x) = (x^2 + a) \bmod n$;
2. [Поиск делителя]
   $U = F(U)$;
   $V = F(V)$;
   $V = F(V)$; // $F(V)$ специально вычисляется дважды.
   $g =$ НОД$(U - V, n)$;
   if($g == 1$) goto [Поиск делителя];
3. [Неудачный инициализатор]
   if($g == n$) goto [Выбрать инициализаторы];
4. [Успех]
   return $g$; // Найден нетривиальный делитель.

Замечательное свойство ро-метода Полларда состоит в том, что для него требуется очень мало памяти: необходимо хранить только число $n$ и текущие значения чисел $U, V$.

Основной цикл, шаг [Поиск делителя], требует 3 модульных умножения (на самом деле, возведения в квадрат) и вычисление НОД. На самом деле, за счет еще одного модульного умножения мы можем сократить количество вычислений НОД, так что он будет вычисляться лишь изредка. А именно, числа $U - V$ могут аккумулироваться (умножаться друг на друга) по модулю $n$ для $k$ итераций, и затем может быть взят НОД этого модульного произведения и числа $n$. Поэтому, если $k$ равно, скажем, 100, то затраты на вычисление НОД будут ничтожны. Поэтому можно считать, что один общий проход цикла состоит из 3 модульных возведений в квадрат и одного модульного умножения.

Есть вероятность, что НОД на шаге [Неудачный инициализатор] будет равен числу $n$, и эта вероятность увеличивается, если использовать идею, описанную выше, для сокращения вычислений НОД. Однако этот дефект можно сгладить, если хранить значения чисел $U, V$ на момент последнего вычисления НОД. Если следующий НОД будет равен $n$, мы можем вернуться к сохраненным значениям чисел $U, V$ и проходить алгоритм по одному шагу, каждый раз вычисляя НОД.

На самом деле существует много вариантов выбора функции $F(x)$. Основной критерий состоит в том, чтобы итерации функции $F$ по модулю $p$ не имели бы длинных $\rho$, или, как их назвал Гай (см. [Guy 1976]), «эпакт». Эпакта простого числа $p$ по отношению к функции $F$, отображающей $\mathbf{Z}_p$ в $\mathbf{Z}_p$, — это наибольшее число $k$, такое, что существует число $s$, при котором все значения $F^{(0)}(s), F^{(1)}(s), \ldots, F^{(k)}(s)$ различны.

Поэтому плохим выбором функции $F(x)$ является функция вида $ax+b$, так как ее эпакта для простого числа $p$ равна мультипликативному порядку числа $a$ по модулю $p$ (если $a \not\equiv 1 \pmod p$), обычно это большой делитель числа $p-1$. (Если $a \equiv 1 \pmod p$ и $b \not\equiv 0 \pmod p$, то эпакта равна $p$.)

Даже среди квадратичных функций $x^2+b$ могут встретиться плохие, например, при $b=0$. Другой, менее очевидный, но, тем не менее, плохой пример функции — $x^2-2$. Если число $x$ можно представить в виде $y+y^{-1}$ по модулю $p$, то $k$-я итерация будет иметь вид $y^{2^k}+y^{-2^k}$ по модулю $p$.

Не известно, является ли эпакта функции $x^2+1$ для числа $p$ достаточно медленно растущей функцией от числа $p$, но Гай предполагает, что она имеет порядок $O\left(\sqrt{p \ln p}\right)$.

Если мы будем знать какую-нибудь дополнительную информацию о простых делителях $p$ числа $n$, то можно использовать многочлены более высоких степеней. Например, так как все простые делители числа Ферма $F_k$ сравнимы с $1 \pmod{2^{k+2}}$, где $k \geqslant 2$ (см. теорему 1.3.5), то мы можем взять многочлен $x^{2^{k+2}}+1$ в качестве функции $F$ для разложения на множители числа $F_k$ при помощи ро-метода Полларда. Мы можем ожидать, что эпакта для простого делителя $p$ числа $F_k$ у этой функции будет меньше, чем у функции $x^2+1$ примерно в $\sqrt{2^{k+1}}$ раз. Действительно, итерации функции $x^2+1$ можно представлять себе как случайное блуждание по множеству квадратов плюс 1, множеству размера $(p-1)/2$. А используя функцию $x^{2^{k+2}}+1$ мы будем блуждать по $2^{k+2}$-м степеням плюс 1, по множеству размера $(p-1)/2^{k+2}$. Однако за использование функции $x^{2^{k+2}}+1$ нам придется заплатить, так как обычный цикл теперь будет требовать $3(k+2)$ модульных возведений в квадрат и одно модульное умножение. Для больших чисел $k$ преимущество очевидно. В связи с этими рассуждениями см. упр. 5.24. Такое ускорение было успешно использовано в работе [Brent and Pollard 1981] для разложения на множители числа $F_8$, исторически это самое замечательное разложение на множители, полученное при помощи ро-метода Полларда. В работе Брента и Полларда также обсуждается несколько более быстрый метод поиска циклов, альтернативный методу поиска циклов Флойда. Он состоит в сохранении значений определенных итераций и сравнении последующих значений с ними.

### 5.2.2. Ро-метод Полларда для вычисления дискретных логарифмов

Поллард также предложил ро-метод для вычисления дискретных логарифмов, но он основан не на итерационном вычислении функции $x^2+1$ или какого-либо простого многочлена такого рода, см. [Pollard 1978]. Пусть нам дана конечная

циклическая группа $G$ и порождающий элемент $g$ группы $G$. Задача дискретного логарифмирования для группы $G$ стоит в том, чтобы выразить данный элемент группы $G$ в виде $g^l$, где $l$ — целое число. Ро-метод можно использовать в любой группе, в которой можно эффективно производить групповые операции и эффективно нумеровать элементы. Мы, однако, будем обсуждать этот метод только для группы $\mathbf{Z}_p^*$ ненулевых вычетов по модулю $p$, где $p$ — простое число, большее 3.

Мы будем рассматривать элементы группы $\mathbf{Z}_p^*$ как числа $\{1, 2, \ldots, p-1\}$. Пусть $g$ — порождающий элемент, и $t$ — какой-то произвольный элемент. Наша задача состоит в нахождении целого числа $l$, такого что $g^l = t$, т. е. $t = g^l \bmod p$. Так как порядок элемента $g$ равен $p-1$, то мы, на самом деле, ищем не конкретное целое число $l$, а класс вычетов по модулю $(p-1)$. Хотя, конечно, мы могли бы потребовать наименьшее неотрицательное значение числа $l$.

Рассмотрим последовательность пар $(a_i, b_i)$ целых чисел по модулю $(p-1)$ и последовательность $(x_i)$ целых чисел по модулю $p$, такие что $x_i = t^{a_i} g^{b_i} \bmod p$. Начальные значения равны $a_0 = b_0 = 0$, $x_0 = 1$. Правила получения $(i+1)$-х значений из $i$-х следующие:

$$(a_{i+1}, b_{i+1}) = \begin{cases} ((a_i + 1) \bmod (p-1), b_i), & \text{если } 0 < x_i < \frac{1}{3}p, \\ (2a_i \bmod (p-1), 2b_i \bmod (p-1)), & \text{если } \frac{1}{3}p < x_i < \frac{2}{3}p, \\ (a_i, (b_i + 1) \bmod (p-1)), & \text{если } \frac{2}{3}p < x_i < p, \end{cases}$$

и поэтому

$$x_{i+1} = \begin{cases} tx_i \bmod p, & \text{если } 0 < x_i < \frac{1}{3}p, \\ x_i^2 \bmod p, & \text{если } \frac{1}{3}p < x_i < \frac{2}{3}p, \\ gx_i \bmod p, & \text{если } \frac{2}{3}p < x_i < p. \end{cases}$$

Поскольку та третья часть отрезка $[0, p]$, которой принадлежит элемент, вряд ли как-то связана с группой $\mathbf{Z}_p^*$, мы можем считать последовательность $(x_i)$ «случайной». Поэтому могут существовать такие числа $j, k$, $j < k = O(\sqrt{p})$, что $x_j = x_k$. Если мы сможем найти такую пару чисел $j, k$, то мы будем иметь $t^{a_j} g^{b_j} = t^{a_k} g^{b_k}$. Следовательно, если $l$ — дискретный логарифм числа $t$, мы будем иметь

$$(a_j - a_k) l \equiv b_k - b_j \pmod{(p-1)}.$$

Если число $a_j - a_k$ взаимно просто с числом $p-1$, это сравнение можно решить и найти дискретный логарифм $l$. Если НОД чисел $a_j - a_k$ и $p-1$ равен $d > 1$, то мы можем решить это сравнение для $l$ по модулю $(p-1)/d$, скажем, $l \equiv l_0 \pmod{(p-1)/d}$. Тогда искомое число $l = l_0 + m(p-1)/d$ для некоторого $m = 0, 1, \ldots, d-1$, поэтому, если НОД $d$ мал, мы можем проверить все эти возможности.

Так же как и для ро-метода разложения на множители, мы используем алгоритм поиска циклов Флойда. Следовательно, на $i$-м шаге алгоритма у нас есть значения $x_i, a_i, b_i$ и $x_{2i}, a_{2i}, b_{2i}$. Если $x_i = x_{2i}$, то нужный нам цикл найден. Если нет, то мы переходим к $(i+1)$-му шагу, вычисляя $x_{i+1}, a_{i+1}, b_{i+1}$ из $x_i, a_i, b_i$ и вычисляя $x_{2i+2}, a_{2i+2}, b_{2i+2}$ из $x_{2i}, a_{2i}, b_{2i}$. Основная работа алгоритма состоит

в вычислении последовательностей $(x_i)$ и $(x_{2i})$. Эти вычисления требуют 3 модульных умножения для перехода от $i$-го шага к $(i+1)$-му шагу. Как и у ро-метода Полларда для разложения на множители, у этого алгоритма размер необходимой памяти минимален.

В работе [Teske 1998] описана немного более сложная версия ро-метода для вычисления дискретных логарифмов. Она имеет 20 веток для итерационной функции вместо 3-х, описанных выше. Численные эксперименты показывают, что ее алгоритм случайного блуждания работает в среднем на 20% быстрее.

Ро-метод для вычисления дискретных логарифмов может быть легко распределен между несколькими процессорами. Описание того, как это сделать, приведено ниже в связи с лямбда-методом.

### 5.2.3. Лямбда-метод Полларда для вычисления дискретных логарифмов

В той же работе, где описывается ро-метод для вычисления дискретных логарифмов ([Pollard 1978]), Поллард также предлагает «лямбда»-метод. Он назван так потому, что форма буквы «$\lambda$» напоминает изображение двух путей, соединяющихся в один. Идея состоит в том, чтобы идти сразу двумя путями: одним от числа $t$, чей дискретный логарифм надо найти, другим от числа $T$, чей дискретный логарифм мы уже знаем. Если эти два пути сойдутся, то мы сможем найти дискретный логарифм числа $t$. Поллард рассматривает шаги на каждом пути как прыжки кенгуру, поэтому этот алгоритм иногда называют «методом кенгуру». Если мы знаем, что искомый дискретный логарифм лежит в известном коротком интервале, то мы можем адаптировать метод кенгуру, а именно, использовать кенгуру с более короткими прыжками.

Одним потрясающем свойством лямбда-метода является тот факт, что он легко распределяется на несколько компьютеров. Каждый участник распределенных вычислений выбирает случайное число $r$ и начинает делать псевдослучайные шаги от числа $t^r$, где $t$ — элемент группы, для которого ищется дискретный логарифм. Каждый участник использует одну и ту же легко вычислимую псевдослучайную функцию $f : G \to S$, где $S$ — относительно небольшое множество чисел со средним значением, сравнимым с размером группы $G$. Степени $g^s$ для $s \in S$ вычисляются заранее. Тогда «блуждание», начиная с $t^r$, выглядит так:

$$w_0 = t^r,\ w_1 = w_0 g^{f(w_0)},\ w_2 = w_1 g^{f(w_1)},\ \ldots.$$

Пусть другой участник, выбрав начальное число $r'$, получил последовательность $w'_0, w'_1, w'_2, \ldots$. Если она пересекается с последовательностью $w_0, w_1, w_2, \ldots$, т. е. $w'_i = w_j$ для некоторых $i, j$, то

$$t^{r'} g^{f(w'_0)+f(w'_1)+\ldots+f(w'_{i-1})} = t^r g^{f(w_0)+f(w_1)+\ldots+f(w_{j-1})}.$$

Поэтому, если $t = g^l$, то

$$(r' - r)l \equiv \sum_{\mu=0}^{j-1} f(w_\mu) - \sum_{\nu=0}^{i-1} f(w'_\nu) \pmod{n},$$

где $n$ — порядок группы.

Обычно этот метод используется, когда порядок группы $n$ является простым. Так как тогда, если все выбираемые в начале вычислений числа $r$ различны по модулю $n$, то сравнение, написанное выше, может быть легко решено для нахождения дискретного логарифма $l$. (Есть небольшая проблема в том, что совпадение может произойти внутри одной последовательности, это означает, что $r = r'$. Однако, если количество участников вычислений велико, то вероятность совпадения между последовательностями больше, чем вероятность совпадения внутри одной последовательности.)

Можно также использовать псевдослучайную функцию, которая обсуждалась в разд. 5.2.2, для лямбда-метода. В таком случае нам будут полезны все совпадения: совпадение внутри одной последовательности также может быть использовано для вычисления дискретного логарифма. В случае такого совпадения лямбда-метод просто превращается в ро-метод. Однако, если нам известно, что искомый дискретный логарифм лежит в коротком интервале, то можно использовать первоначальный метод. Тогда время работы будет около квадратного корня из длины интервала. В этом случае среднее значение целых чисел из множества $S$ должно быть меньше для того, чтобы «кенгуру» прыгали только по интервалу нужной длины.

Центральный компьютер должен отслеживать все последовательности от всех участников для выявления совпадений. По парадоксу дней рождения мы будем ожидать совпадения, когда количество элементов во всех последовательностях будет порядка $O(\sqrt{n})$. Очевидно, что в описанном виде этот метод требует больших затрат памяти центрального компьютера. Следующая идея, описанная в работе [van Oorschot and Wiener 1999] (и приписываемая Куискуатеру и Делескелю, которые, в свою очередь, признают заслугу Р. Райвеста), сильно уменьшает требования к памяти и, таким образом, делает этот метод применимым к решению сложных проблем. Идея состоит в рассмотрении так называемых выделенных точек. Мы предполагаем, что элементы группы представляются целыми числами (или, возможно, наборами целых чисел). Выделенное двоичное поле длины $k$ в таком числе будет состоять из одних нулей примерно $1/2^k$-ю часть всего времени. Случайное блуждание будет проходить через такие выделенные точки в среднем через каждые $2^k$ шагов. Если две случайные последовательности где-нибудь пересекутся, то они будут пересекаться и дальше и вместе попадут в следующую выделенную точку. Итак, идея состоит в отправке на центральный компьютер только этих выделенных точек, что уменьшит необходимый размер памяти в $2^k$ раз.

В марте 1998 г. был достигнут значительный успех в вычислении дискретного логарифма в группе эллиптической кривой, имеющей порядок, равный 97-битовому простому числу $n$; см. [Escott et al. 1998]. Группа из 588 человек

из 16 стран использовала около 1200 компьютеров и потратила более 53 дней на решение этой задачи. Было произведено около $2 \cdot 10^{14}$ сложений в группе эллиптической кривой, количество полученных выделенных точек равнялось 186364. (Значение числа $k$ в определении выделенной точки было взято равным 30, поэтому значения только примерно 1 шага из миллиарда отсылались на центральный компьютер.) В 2002 г. был вычислен дискретный логарифм на эллиптической кривой (elliptic-curve discrete logarithm — EDL) для 109-битового (= 33-значного) простого числа; см. замечания вслед за алгоритмом 8.1.8.

Для вычисления дискретных логарифмов в мультипликативной группе конечного поля у нас есть субэкспоненциальные методы (см. разд. 6.4), позволяющие оперировать гораздо бóльшими числами. На данный момент рекордом[1] дискретного логарифмирования над полем $\mathbf{F}_p$ является вычисление 2001 г., выполненное Жу и Лерсье для $p$, равного 120-значному простому числу $\lfloor 10^{119}\pi \rfloor + 207819$. Фактически они нашли два дискретных логарифма в этом поле по основанию порождающего элемента 2, а именно, дискретный логарифм для $t = \lfloor 10^{119}e \rfloor$ и для $t+1$. Их метод был основан на решете числового поля.

Относящиеся к более позднему времени достижения в области параллельных ро-методов включают рассмотрение криптографической задачи дискретного логарифмирования (DL) [van Oorschot and Wiener 1999] и попытки распараллеливания самого ро-метода Полларда для разложения чисел на множители (не для задачи DL) [Crandall 1999d]. По поводу последней проблемы см. упр. 5.24 и 5.25. Некоторые последние достижения, относящиеся к DL-варианту ро-метода, см. в работах [Pollard 2000] и [Teske 2001]. Есть также очень доступная обзорная статья, касающаяся общей задачи DL [Odlyzko 2000].

## 5.3. Детские шаги, гигантские шаги

Пусть $G = \langle g \rangle$ — циклическая группа порядка не выше $n$, и пусть $t \in G$ — элемент этой группы. Нам надо найти целое число $l$, такое что $g^l = t$. Мы можем ограничиться поиском числа $l$ в интервале $[0, n-1]$. Запишем число $l$ в системе счисления с основанием $b$, где $b = \lceil \sqrt{n} \rceil$. Тогда $l = l_0 + l_1 b$, где $0 \leqslant l_0, l_1 \leqslant b-1$. Заметим, что $g^{l_1 b} = t g^{-l_0} = t h^{l_0}$, где $h = g^{-1}$. Следовательно, мы можем найти числа $l_0, l_1$, вычислив последовательности $\{g^0, g^b, \ldots, g^{(b-1)b}\}$ и $\{th^0, th^1, \ldots, th^{b-1}\}$ и отсортировав их. Когда они отсортированы, мы проходим по одной из последовательностей и находим, между какими двумя элементами другой последовательности лежит каждый элемент, выявляя при этом совпадения. (Эта идея описана в псевдокоде в алгоритме 7.5.1.) Если $g^{ib} = th^j$, то мы можем взять $l = j + ib$.

Дадим более формальное описание алгоритма:

**Алгоритм 5.3.1** («детские шаги, гигантские шаги» для дискретных логарифмов). Дана циклическая группа $G$ с порождающим элементом $g$, верхняя граница $n$ для порядка группы $G$ и элемент $t \in G$. Алгоритм выдает целое число $l$, такое что $g^l = t$. (Имеется в виду, что мы можем представить элементы группы в неком числовом виде, чтобы можно было их отсортировать.)

[1] См. [Dorofeev et al. 2007]. — *Прим. ред.*

1. [Установка границ]
   $b = \lceil \sqrt{n} \rceil$;
   $h = \left(g^{-1}\right)^b$; // Например, при помощи алгоритма 2.1.5
2. [Составление последовательностей]
   $A = \left\{ g^i : i = 0, 1, \ldots, b-1 \right\}$;
   $B = \left\{ th^j : j = 0, 1, \ldots, b-1 \right\}$;
3. [Сортировка и поиск пересечения]
   Сортировка последовательностей $A, B$;
   Поиск пересечения, скажем $g^i = th^j$; // При помощи алгоритма 7.5.1.
   return $l = i + jb$;

Заметим, что условия применимости алгоритма гарантируют нам, что последовательности $A, B$ имеют общий элемент. Заметим также, что обе последовательности можно не сортировать, достаточно отсортировать одну. Пусть, например, последовательность $A$ отсортирована, а последовательность $B$ генерируется по одному элементу. Тогда мы можем искать совпадение с данным элементом в последовательности $A$, если у нас есть алгоритм быстрого поиска в отсортированной последовательности. После того как мы найдем совпадение, можно не вычислять дальше последовательность $B$. Таким образом мы сэкономим в среднем 50% времени.

Шаг [Составление последовательностей] требует $O(\sqrt{n})$ групповых операций, шаг [Сортировка и поиск пересечения] — $O(\sqrt{n} \ln n)$ сравнений. Алгоритму требуется память для хранения $O(\sqrt{n})$ элементов группы. Если мы не знаем размер группы $G$, то мы можем заставить число $n$ пробегать последовательность $2^k$ для $k = 1, 2, \ldots$. Если мы не нашли совпадения для какого-то $n = 2^k$, то мы повторим алгоритм для числа $n = 2^{k+1}$. Конечно, наборы, полученные на предыдущем шаге, необходимо сохранить и расширить для следующего шага. Таким образом, если группа $G$ имеет порядок $m$, мы вычислим дискретный логарифм числа $t$ за $O(\sqrt{m} \ln m)$ операций с необходимыми затратами памяти для $O(\sqrt{m})$ элементов группы.

Улучшенную версию этой идеи можно найти в работах [Buchmann et al. 1997], [Terr 1999]. См. также [Blackburn and Teske 1999] для информации о других методах типа «детские шаги, гигантские шаги».

Сравним алгоритм 5.3.1 с ро-методом для вычисления дискретных логарифмов из разд. 5.2.2. Время работы последнего имеет порядок $O(\sqrt{m})$ и необходимые затраты памяти очень малы. Однако ро-метод является эвристическим, а метод «детские шаги, гигантские шаги» является строгим. На практике же нет причин не использовать эвристический метод для вычисления дискретных логарифмов лишь по той причине, что теоретики еще не смогли доказать, что этот метод работает и проводит вычисления за указанное время. Поэтому на практике ро-метод предпочтительнее метода «детские шаги, гигантские шаги».

Однако простую и изящную идею, лежащую в основе метода «детские шаги, гигантские шаги», можно также использовать и во многих других случаях, как мы увидим в разд. 7.5. Как показано в работе [Shanks 1971], эту идею

можно использовать для разложения на множители. На самом деле, в этой статье и появляется впервые метод «детские шаги, гигантские шаги». В статье этот метод рассматривается в связи с группой классов бинарных квадратичных форм заданного дискриминанта. Мы еще обратимся к этому методу в конце главы, в разд. 5.6.4.

## 5.4. $(p-1)$-метод Полларда

Из малой теоремы Ферма мы знаем, что если $p$ — нечетное простое число, то $2^{p-1} \equiv 1 \pmod{p}$. Более того, если $p-1|M$, то $2^M \equiv 1 \pmod{p}$. Поэтому, если число $p$ является простым делителем числа $n$, то $p$ делит НОД$(2^M - 1, n)$. $(p-1)$-метод Дж. Полларда использует эту идею для разложения числа $n$ на множители. Его идея состоит в том, чтобы выбирать числа $M$ с большим количеством делителей вида $p-1$, и тем самым проверять множество простых чисел $p$ (на возможность быть делителями числа $n$) за один шаг.

Обозначим через $M(k)$ наименьшее общее кратное всех целых чисел до числа $k$. Так, $M(1)=1$, $M(2)=2$, $M(3)=6$, $M(4)=12$ и т. д. Последовательность $M(1), M(2), \ldots$ можно вычислить рекурсивно следующим образом. Предположим, что число $M(k)$ мы уже вычислили. Если число $k+1$ не является простым или степенью простого числа, то $M(k+1)=M(k)$. Если $k+1=p^a$, где число $p$ — простое, то $M(k+1)=pM(k)$. Проведя предварительные вычисления при помощи решета (см. разд. 3.2), мы можем вычислить все простые числа до некоторой границы и дополнить этот список степенями простых чисел. Следовательно, последовательность $M(1), M(2), \ldots$ может быть вычислена довольно просто. В следующем алгоритме мы получим $M(B)$, используя простые числа до границы $B$ и их степени до границы $B$.

**Алгоритм 5.4.1** (базовый $(p-1)$-метод Полларда). Нам дано составное нечетное число $n$ и граница поиска $B$. Алгоритм пытается найти нетривиальный делитель числа $n$.

1. [Задание базы степеней простых чисел]
   Найдем, например при помощи алгоритма 3.2.1, последовательность простых чисел $p_1 < p_2 < \ldots < p_m \leqslant B$. Для каждого такого простого числа $p_i$ найдем также максимальную степень $a_i$, такую что $p_i^{a_i} \leqslant B$;
2. [Выполнить вычисление степеней]
   $c = 2$; // На самом деле можно использовать случайное значение $c$.
   for$(1 \leqslant i \leqslant m)$ {
   for$(1 \leqslant j \leqslant a_i)$ $c = c^{p_i} \bmod n$;
   }
3. [Проверка НОД]
   $g = $ НОД$(c-1, n)$;
   return $g$; // Мы надеемся, что $1 < g < n$.

Существуют два варианта неверного завершения работы такого $(p-1)$-метода: (1) если НОД$(c-1, n) = 1$ или (2) если этот НОД равен самому числу $n$.

Приведем пример, иллюстрирующий эти проблемы. Пусть $n = 2047$ и $B = 10$. Получаем набор степеней простых чисел $2^3, 3^2, 5, 7$ и итоговое значение числа $g$ получаем равным 1. Однако мы можем увеличить границу поиска. Увеличим границу $B$ до 12, получим еще одну степень простого числа, а именно 11. Теперь итоговое значение числа $g$ получаем равным самому числу $n$. Таким образом, алгоритм опять не выдал собственный делитель числа $n$. Даже более частое вычисление НОД на шаге [Проверка НОД] не поможет нам для нахождения делителей числа $n$.

Все дело в том, что $2047 = 2^{11} - 1 = 23 \cdot 89$, следовательно, НОД $\left(2^M - 1, n\right) = n$, если $11|M$, и равен 1 в противном случае. Очевидно, что в этом случае увеличение границ поиска ничего не даст. Однако мы можем заменить начальное число $c = 2$ на число $c = 3$ или на какое-нибудь другое. При выборе числа $c = 3$ нам надо будет вычислять НОД $\left(3^{M(B)} - 1, n\right)$. Однако этот способ тоже плохо работает для числа $n = 2047$; наименьшее значение начального числа, при котором делитель числа $n$ находится, — это $c = 12$. Для этого значения мы получаем НОД $\left(12^{M(8)} - 1, n\right) = 89$.

Существует другой подход в случае, когда алгоритм завершается со значением НОД, равным самому числу $n$. Выберем начальное целое число $c$ случайным образом и сделаем перестановку в списке степеней простых чисел так, чтобы степень числа 2 была в конце. Затем, так же как на шаге [Проверка НОД], будем повторно вычислять НОД каждый раз перед тем как использовать коэффициент 2. Нетрудно показать, что если число $n$ делится как минимум на 2 различных нечетных простых числа, то вероятность того, что при случайном выборе начального числа $c$ алгоритм не сработает из-за того, что НОД равен $n$, не превосходит $1/2$.

Надо отметить, что на практике очень редко алгоритм заканчивает работу с НОД, равным самому числу $n$. Гораздо чаще алгоритм заканчивает свою работу с НОД, равным 1. В этом случае мы можем увеличить границу поиска $B$ и/или использовать так называемую *вторую стадию*.

Можно предложить несколько различных версий второй стадии, мы опишем оригинальный вариант. Рассмотрим вторую границу поиска $B'$, большую, чем $B$. После проведения поиска по степеням $M(1), M(2), \ldots, M(B)$ мы будем исследовать степени $QM(B)$, где простые числа $Q$ пробегают интервал $(B, B']$. Таким образом, у нас будет шанс найти простые числа $p|n$, такие что $p-1 = Qu$, где $Q$ — некоторое простое число из интервала $(B, B']$ и $u|M(B)$. Провести поиск по экспонентам $QM(B)$ можно довольно просто. Пусть простые числа из интервала $(B, B']$ — это $Q_1 < Q_2 < \ldots$. Заметим, что значение $2^{Q_1 M(B)} \bmod n$ можно вычислить из значения $2^{M(B)} \bmod n$ за $O(\ln Q_1)$ шагов. Для получения $2^{Q_2 M(B)} \bmod n$ мы можем умножить $2^{Q_1 M(B)} \bmod n$ на $2^{(Q_2 - Q_1)M(B)} \bmod n$, потом умножить на $2^{(Q_3 - Q_2)M(B)} \bmod n$ для получения $2^{Q_3 M(B)} \bmod n$ и т. д. Все разности $Q_{i+1} - Q_i$ гораздо меньше, чем сами числа $Q_i$, поэтому для разных значений $d$ этих разностей мы можем заранее вычислить вычеты $2^{dM(B)} \bmod n$. Следовательно, если, например, $B' > 2B$, то затраты на вычисление всех вычетов $2^{Q_i M(B)} \bmod n$ составляют всего одно модульное умножение на каждое $Q_i$. Если мы готовы потратить на вторую стадию алгоритма столько же време-

ни, сколько на базовый $(p-1)$-метод, то мы можем взять границу $B'$ гораздо большую, чем $B$, даже порядка $B \ln B$.

Существует множество интересных аспектов, относящихся ко второй стадии алгоритма, такие как средства для дальнейшего ускорения, проявления парадокса дней рождения и т. д. Некоторые из этих аспектов см. в работах [Montgomery 1987, 1992a], [Crandall 1996a] и в упр. 5.9.

С основной идеей $(p-1)$-метода Полларда мы еще встретимся в методе эллиптических кривых (ECM) Ленстры для разложения на множители целых чисел (см. разд. 7.4).

## 5.5. Метод вычисления значений многочлена

Предположим, что мы умеем легко вычислять значения функции $F(k,n) = k! \bmod n$. Тогда задача разложения чисел на множители и задача проверки на простоту также могут быть легко решены. Например, теорема Вильсона—Лагранжа (теорема 1.3.6) утверждает, что целое число $n > 1$ является простым тогда и только тогда, когда $F(n-1,n) = n-1$. Или число $n > 1$ является простым тогда и только тогда, когда $F(\lceil\sqrt{n}\rceil, n)$ взаимно просто с $n$. Задачу разложения на множители можно решить, например, так: при помощи двоичного поиска найдем наименьшее натуральное число $k$, такое что НОД $(F(k,n),n) > 1$; это число $k$ естественно будет являться наименьшим простым делителем числа $n$.

Какой бы странной эта идея не казалась, в самом деле существуют быстрые алгоритмы факторизации на ее основе. В том числе самый быстрый детерминированный, строго проанализированный алгоритм разложения на множители. Это метод вычисления значений многочлена Полларда—Штрассена (см. [Pollard 1974] и [Strassen 1976]).

Идея состоит в следующем. Положим $B = \lceil n^{1/4} \rceil$, и пусть $f(x)$ — многочлен $x(x-1)\ldots(x-B+1)$. Тогда $f(jB) = (jB)!/((j-1)B)!$ для любого натурального числа $j$, поэтому наименьшее число $j$, такое что НОД $(f(jB),n) > 1$, ограничивает наименьший простой делитель числа $n$ интервалом $((j-1)B, jB]$. Далее, если сам НОД находится в указанном интервале, то он и является наименьшим простым делителем числа $n$. Если же этот НОД превосходит число $jB$, мы можем последовательно перебрать числа из указанного интервала и проверить, являются ли они делителями числа $n$. Первый найденный делитель будет наименьшим простым делителем числа $n$. Очевидно, что последние вычисления требуют не более $B$ арифметических операций с целыми числами размера $n$, т. е. $O(n^{1/4})$ операций. Подсчитаем теперь количество операций в предыдущих шагах. Если мы сможем вычислить все значения $f(jB) \bmod n$ при $j = 1, 2, \ldots, B$, то мы сможем вычислить все значения НОД и найти среди них первое, превосходящее 1.

При помощи алгоритма 9.6.7 мы можем вычислить значения $f(x)$ как многочлена в кольце $\mathbf{Z}_n[x]$ (т. е. все коэффициенты взяты по модулю $n$). Таким образом, мы можем вычислить все значения $f(jB)$ по модулю $n$ при

$j = 1, 2, \ldots, B$ за $O\left(B \ln^2 B\right) = O\left(n^{1/4} \ln^2 n\right)$ арифметических операций с целыми числами размера $n$. Выражение $O\left(n^{1/4} \ln^2 n\right)$ и является оценкой сложности метода вычисления значений многочлена Полларда—Штрассена для разложения числа $n$.

## 5.6. Бинарные квадратичные формы

Лагранж, Лежандр и Гаусс в конце XVIII в. разработали теорию бинарных квадратичных форм, которая играла и до сих пор играет важную роль в вычислительной теории чисел.

### 5.6.1. Основы теории квадратичных форм

Для каждой тройки целых чисел $a, b, c$ мы можем рассмотреть квадратичную форму $ax^2 + bxy + cy^2$. Эта форма является многочленом от двух переменных $x, y$, но часто мы будем опускать сами переменные и просто считать квадратичную форму упорядоченной тройкой целых чисел $(a, b, c)$.

Мы будем говорить, что квадратичная форма $(a, b, c)$ представляет целое число $n$, если существуют целые числа $x, y$, такие что $ax^2 + bxy + cy^2 = n$. Таким образом, мы можем присоединить к каждой квадратичной форме $(a, b, c)$ определенное множество целых чисел, а именно, те числа, которые эта форма представляет. Заметим также, что определенные замены переменных могут перевести квадратичную форму $(a, b, c)$ в другую форму $(a', b', c')$, но при этом сохранить множество целых чисел, которые она представляет. В частности, пусть

$$x = \alpha X + \beta Y, \quad y = \gamma X + \delta Y,$$

где $\alpha, \beta, \gamma, \delta$ — целые числа. Сделав такую замену переменных, мы получим

$$\begin{aligned} ax^2 + bxy + cy^2 &= a(\alpha X + \beta Y)^2 + b(\alpha X + \beta Y)(\gamma X + \delta Y) + c(\gamma X + \delta Y)^2 \\ &= a'X^2 + b'XY + c'Y^2. \end{aligned} \tag{5.1}$$

Следовательно, каждое целое число, представимое формой $(a', b', c')$, является также представимым формой $(a, b, c)$. Мы также можем утверждать обратное, если существуют такие целые числа $\alpha', \beta', \gamma', \delta'$, что

$$X = \alpha' x + \beta' y, \quad Y = \gamma' x + \delta' y.$$

Т. е. матрицы

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}, \quad \begin{pmatrix} \alpha' & \beta' \\ \gamma' & \delta' \end{pmatrix}$$

являются обратными друг к другу. Квадратная матрица с целыми числами имеет обратную матрицу с целыми числами тогда и только тогда, когда ее определитель равен $\pm 1$. Следовательно, мы можем сделать вывод, что квадратичные формы $(a, b, c)$ и $(a', b', c')$, полученные одна из другой заменой переменных (5.1), представляют одно и то же множество целых чисел, если $\alpha\delta - \beta\gamma = \pm 1$.

Рассмотрение определителей $+1$ и $-1$ не даст нам ничего существенно нового по сравнению с рассмотрением только определителей $+1$. (Например, мы можем перейти от формы $(a, b, c)$ к $(a, -b, c)$ и к $(c, b, a)$ при помощи замены переменных с определителем $-1$, но такие замены легко распознаются, и их можно считать частью более сложных замен с определителем $+1$. Поэтому мы не сильно ограничиваем общность, рассматривая только случай определителей $+1$.) Мы будем говорить, что две квадратичные формы *эквивалентны*, если для них существует замена переменных (5.1) с определителем $+1$. Такая замена переменных называется унимодулярной; таким образом, две квадратичные формы являются эквивалентными, если мы можем перейти от одной к другой при помощи унимодулярной замены переменных.

Понятие эквивалентности квадратичных форм является «отношением эквивалентности»: каждая форма $(a, b, c)$ эквивалентна самой себе; если $(a, b, c)$ эквивалентна $(a', b', c')$, то верно и обратное; и если две формы эквиваленты третьей, то они эквивалентны между собой. Мы оставляем доказательство этих простых фактов в качестве упражнения 5.10.

Остается вычислительная проблема определения эквивалентности двух данных квадратичных форм. Дискриминантом формы $(a, b, c)$ называется целое число $b^2 - 4ac$. Эквивалентные формы имеют одинаковый дискриминант (см. упр. 5.12). Поэтому иногда очень просто можно определить, что две формы *не* являются эквивалентными, а именно, если они имеют разные дискриминанты. Однако обратное неверно. Например, две формы $x^2 + xy + 4y^2$ и $2x^2 + xy + 2y^2$ имеют одинаковый дискриминант $-15$, но первая может принимать значение 1 (при $x = 1$ и $y = 0$), а вторая — нет. Следовательно, эти формы не эквивалентны.

Если бы в каждом классе эквивалентности можно было выбрать одну специальную квадратичную форму и если бы мы могли легко найти эту специальную форму по любому представителю данного класса, то мы могли бы легко понять, являются две данные формы эквивалентными или нет. А именно, мы могли бы найти специальную форму для каждой из данных двух форм и сравнить их между собой. Если они совпадают, то формы эквивалентны, и наоборот.

Это можно легко сделать для квадратичных форм с отрицательным дискриминантом. На самом деле, вся теория бинарных квадратичных форм разделяется на две части в зависимости от знака дискриминанта. Квадратичные формы с положительным дискриминантом могут представлять как положительные, так и отрицательные числа, что неверно для форм с отрицательным дискриминантом. (Формы с дискриминантом, равным нулю, являются тривиальными, и их изучение сводится к изучению последовательностей полных квадратов.)

Теория квадратичных форм с положительным дискриминантом несколько более сложна, чем соответствующая теория форм с отрицательным дискриминантом. Существуют интересные алгоритмы разложения на множители, связанные как с квадратичными формами с положительным дискриминантом, так и с квадратичными формами с отрицательным дискриминантом. В це-

лях краткости изложения мы в основном будем рассматривать более простой случай форм с отрицательным дискриминантом. За описаниями алгоритмов, относящихся к квадратичным формам с положительным дискриминантом, мы отсылаем читателя к книге [Cohen 2000].

Введем еще одно ограничение. Мы говорили, что бинарная квадратичная форма с отрицательным дискриминантом не может представлять как положительные, так и отрицательные числа одновременно. Поэтому мы будем рассматривать только те формы, которые никогда не представляют отрицательные числа. Если форма $(a, b, c)$ является таковой, то форма $(-a, -b, -c)$ никогда не представляет положительные числа, поэтому общность мы не ограничиваем. Это ограничение можно ввести и по-другому: мы будем рассматривать только те формы $(a, b, c)$, для которых $b^2 - 4ac < 0$ и $a > 0$. Заметим, что эти условия влекут за собой неравенство $c > 0$.

Будем говорить, что форма $(a, b, c)$ с отрицательным дискриминантом является приведенной, если

$$-a < b \leqslant a < c \quad \text{или} \quad 0 \leqslant b \leqslant a = c. \tag{5.2}$$

**Теорема 5.6.1** (Гаусс). *Никакие две приведенные формы с отрицательным дискриминантом не эквивалентны. И каждая форма $(a, b, c)$ с отрицательным дискриминантом и с $a > 0$ эквивалентна какой-то приведенной форме.*

Таким образом, теорема 5.6.1 дает нам способ определения одной специальной формы в каждом классе эквивалентности; в качестве такой формы можно взять приведенную форму. Доказательство теоремы можно найти, например, в книге [Rose 1988]. Обсудим, как найти приведенную форму, эквивалентную данной. Рассмотрим очень простой алгоритм Гаусса, решающий эту задачу.

**Алгоритм 5.6.2** (приведение формы с отрицательным дискриминантом). Дана квадратичная форма $(A, B, C)$, где $A, B, C$ — целые числа, удовлетворяющие условиям $B^2 - 4AC < 0, A > 0$. Этот алгоритм строит приведенную квадратичную форму, эквивалентную $(A, B, C)$.

1. [Цикл замен]
   while($A > C$ or $B > A$ or $B \leqslant -A$) {
   if($A > C$) $(A, B, C) = (C, -B, A)$; // Замена «типа (1)».
   if($A \leqslant C$ and ($B > A$ or $B \leqslant -A$)) {
   Найдем $B^*, C^*$, такие, что выполняются условия:
   $$-A < B^* \leqslant A,$$
   $$B^* \equiv B \pmod{2A},$$
   $$B^{*2} - 4AC^* = B^2 - 4AC$$
   $(A, B, C) = (A, B^*, C^*)$; // Замена «типа (2)».
   }
   }
2. [Последняя корректировка]
   if($A == C$ and $-A < B < 0$) $(A, B, C) = (A, -B, C)$;
   return $(A, B, C)$;

Замены типа (2) не меняют первый коэффициент $A$, а замены типа (1) уменьшают его. Следовательно, мы можем сделать только конечное число замен типа (1). Заметим также, что две замены типа (2) не могут быть сделаны подряд. Следовательно, этот алгоритм когда-нибудь завершит свою работу для любой формы. Доказательство того, что алгоритм выдает эквивалентную форму мы оставим в качестве упражнения 5.13. (Эти факты показывают, что каждая форма с отрицательным дискриминантом и положительным первым коэффициентом эквивалентна некоторой приведенной форме, что составляет половину теоремы 5.6.1.)

### 5.6.2. Разложение на множители при помощи представления квадратичными формами

Одним из старых подходов к разложению чисел на множители является способ, восходящий к Ферма и заключающийся в попытке представить число $n$ двумя различными способами квадратичной формой $(1, 0, 1)$. Т. е. мы пытаемся представить число $n$ двумя различными способами в виде суммы двух квадратов. Например, $65 = 8^2 + 1^2 = 7^2 + 4^2$. Тогда НОД чисел $(8 \cdot 4 - 1 \cdot 7)$ и 65 равен делителю, а именно, числу 5. В общем случае, если

$$n = x_1^2 + y_1^2 = x_2^2 + y_2^2, \quad x_1 \geqslant y_1 \geqslant 0, \quad x_2 \geqslant y_2 \geqslant 0, \quad x_1 > x_2,$$

то $1 < \text{НОД}(x_1y_2 - y_1x_2, n) < n$. Действительно, положим $A = x_1y_2 - y_1x_2$, $B = x_1y_2 + y_1x_2$. Нам будет достаточно показать, что

$$AB \equiv 0 \pmod{n}, \quad 1 < A \leqslant B < n.$$

Первое сравнение следует из того, что $y_i^2 \equiv -x_i^2 \pmod{n}$ при $i = 1, 2$, поскольку $AB = x_1^2y_2^2 - y_1^2x_2^2 \equiv -x_1^2x_2^2 + x_1^2x_2^2 \equiv 0 \pmod{n}$. Очевидно, что $A \leqslant B$. Для доказательства того, что $A > 1$, заметим, что $y_1x_2 < y_2x_2 < y_2x_1$. Для получения неравенства $B < n$, заметим, что $uv \leqslant \frac{1}{2}u^2 + \frac{1}{2}v^2$ для положительных чисел $u, v$, причем равенство достигается тогда и только тогда, когда $u = v$. Следовательно, так как $x_1 > y_2$, мы имеем

$$B = x_1y_2 + y_1x_2 < \tfrac{1}{2}x_1^2 + \tfrac{1}{2}y_2^2 + \tfrac{1}{2}y_1^2 + \tfrac{1}{2}x_2^2 = \tfrac{1}{2}n + \tfrac{1}{2}n = n,$$

что завершает доказательство.

Сразу возникают два вопроса. Все ли составные числа $n$ имеют два различных представления в виде суммы двух квадратов? Если число $n$ имеет два таких различных представления, сможем ли мы их легко найти? К сожалению, ответ на оба вопроса отрицательный. К первому вопросу относится теорема о том, что множество чисел, представимых хотя бы одним способом в виде суммы двух квадратов, имеет асимптотическую плотность нуль. На самом деле, любое число, делящееся на простое число $p \equiv 3 \pmod{4}$ в нечетной степени, не имеет представления в виде суммы двух квадратов. Такие числа составляют почти все множество натуральных чисел (см. упр. 5.16). Однако существует довольно много чисел, которые все-таки можно представить в виде суммы двух квадратов. Например, любое число вида $pq$, где $p, q$ — простые числа, сравнимые с 1 (mod 4), может быть представлено в виде суммы двух квадратов

двумя различными способами. Но мы все равно не знаем способа быстрого нахождения таких представлений.

Несмотря на эти трудности люди пытались работать с этой идеей, чтобы прийти к методу разложения чисел на множители. Мы опишем алгоритм, приведенный в работе [McKee 1996], который может разложить число $n$ на множители за $O(n^{1/3+\varepsilon})$ арифметических операций для любого фиксированного $\varepsilon > 0$.

Пусть форма $(a, b, c)$ представляет натуральное число $n$, скажем $ax^2+bxy+cy^2 = n$, и пусть $D = b^2 - 4ac$ — дискриминант формы $(a, b, c)$. Тогда $(2ax + by)^2 - Dy^2 = 4an$. Т. е. мы получили решение $u, v$ сравнения $u^2 - Dv^2 \equiv 0 \pmod{4n}$. Положим

$$\mathcal{S}(D, n) = \left\{(u, v) : u^2 - Dv^2 \equiv 0 \pmod{4n}\right\},$$

таким образом, это замечание дает нам отображение из множества представлений числа $n$ формами дискриминанта $D$ в $\mathcal{S}(D, n)$. Можно также показать, что эквивалентные представления числа $n$ посредством (5.1) дают нам пары $(u, v), (u', v')$ из множества $\mathcal{S}(D, n)$, удовлетворяющие условию $uv' \equiv u'v \pmod{2n}$ (см. упр. 5.18).

Зафиксируем теперь числа $D, n$, причем $D < 0$ и $n$ не делится ни на одно простое число, меньшее $\sqrt{|D|}$. Если число $h$ является решением сравнения $h^2 \equiv D \pmod{4n}$, то форма $(A, h, n)$, где $h^2 = D + 4An$, представляет число $n$ при $x = 0, y = 1$. Это представление отображается в пару $(h, 1)$ в множестве $\mathcal{S}(D, n)$. Предположим, что мы привели форму $(A, h, n)$, и форма $(a, b, c)$ — эквивалентная ей приведенная форма. Пусть соответствующее представление числа $n$ приведенной формой дается значениями $x, y$ и отображается в пару $(u, v)$ в множестве $\mathcal{S}(D, n)$. Тогда по замечанию из предыдущего абзаца мы имеем $u \equiv vh \pmod{2n}$. Более того, число $v$ взаимно просто с числом $n$. В самом деле, если $p$ — простое число, которое делит $v\,(= y)$ и $n$, то $p$ также делит $u = 2ax + by$, и, следовательно, $p$ делит $2ax$. Но НОД$\,(x, y) = 1$, так как пара $0, 1$ была приведена к паре $x, y$ при помощи унимодулярной замены переменных. Поэтому $p$ делит $2a$. Но форма $(a, b, c)$ является приведенной, поэтому $0 < a \leqslant \sqrt{|D|/3}$ (см. упр. 5.14). Из ограничений на число $n$ следует, что $p > \sqrt{|D|} \geqslant 2$, следовательно, в итоге получаем, что число $p$ не может делить число $2a$.

Предположим теперь, что мы нашли два решения $h_1, h_2$ сравнения $h^2 \equiv D \pmod{4n}$, причем $h_1 \not\equiv \pm h_2 \pmod{n}$. Как показано в предыдущем абзаце, эти два решения отображаются в две пары $(u_i, v_i)$ в множестве $\mathcal{S}(D, n)$, причем $u_i \equiv v_i h_i \pmod{2n}$ и $v_1 v_2$ взаимно просто с $n$. Мы утверждаем, что

$$1 < \text{НОД}\,(u_1 v_2 - u_2 v_1, n) < n.$$

Действительно, мы имеем $u_1^2 v_2^2 - u_2^2 v_1^2 \equiv D v_1^2 v_2^2 - D v_2^2 v_1^2 \equiv 0 \pmod{4n}$, поэтому нам будет достаточно показать, что $u_1 v_2 \not\equiv \pm u_2 v_1 \pmod{n}$. Если $u_1 v_2 \equiv u_2 v_1 \pmod{n}$, то

$$0 \equiv u_1 v_2 - u_2 v_1 \equiv v_1 h_1 v_2 - v_2 h_2 v_1 = v_1 v_2 (h_1 - h_2) \pmod{n},$$

т. е. $h_1 \equiv h_2 \pmod{n}$, пришли к противоречию. Аналогично, если $u_1 v_2 \equiv -u_2 v_1 \pmod{n}$, то $h_1 \equiv -h_2 \pmod{n}$, снова получаем противоречие.

Таким образом, мы получили, что если существуют два квадратных корня $h_1, h_2$ из числа $D$ по модулю $4n$, такие что $h_1 \not\equiv \pm h_2 \pmod{n}$, то существуют две пары $(u_1, v_1), (u_2, v_2)$, указанные выше, такие что НОД$(u_1 v_2 - u_2 v_1, n)$ является нетривиальным делителем числа $n$.

Макки, таким образом, предлагает среди всех пар $(u, v)$ в множестве $\mathcal{S}(D, n)$ найти две пары $(u_1, v_1), (u_2, v_2)$, указанные выше. Очевидно, что мы можем ограничиться поиском пар $(u, v)$, таких что $u \geqslant 0, v \geqslant 0$.

Пусть форма $(a, b, c)$ имеет отрицательный дискриминант $D$ и $ax^2 + bxy + cy^2 = n$. Тогда соответствующая пара $(u, v)$ из множества $\mathcal{S}(D, n)$ удовлетворяет условию $u^2 - Dv^2 = 4an$, и, следовательно, $|u| \leqslant 2\sqrt{an}$. Далее, если форма $(a, b, c)$ является приведенной, то $1 \leqslant a \leqslant \sqrt{|D|/3}$. Макки предлагает выбрать число $a$ так, чтобы выполнялось условие $1 \leqslant a \leqslant \sqrt{|D|/3}$. Затем для этого $a$ искать целые числа $u$, такие что $0 \leqslant u \leqslant 2\sqrt{an}$ и $u^2 \equiv 4an \pmod{|D|}$. Затем для каждого найденного числа $u$ проверяем, является ли число $(u^2 - 4an)/D$ полным квадратом. Если мы знаем разложение на простые сомножители числа $D$, то мы можем быстро определить, в каких классах вычетов по модулю $|D|$ должно лежать число $u$. Таких классов вычетов будет не больше, чем $|D|^{\varepsilon}$. Для каждого класса вычетов наш поиск числа $u$ сводится к перебору арифметической прогрессии из не более чем $\lceil 1 + 2\sqrt{an}/|D| \rceil$ элементов. Следовательно, для данного числа $a$ мы проверяем не более $|D|^{\varepsilon} + 2\sqrt{an}/|D|^{1-\varepsilon}$ вариантов числа $u$. Суммируя это выражение для всех вариантов выбора числа $a$ до $\sqrt{|D|/3}$, получаем $O(|D|^{1/2+\varepsilon} + \sqrt{n}/|D|^{1/4-\varepsilon})$. Т. е. если мы сможем найти подходящее число $D$, такое что $|D|$ будет порядка $n^{2/3}$, то мы получим алгоритм, требующий не более $O(n^{1/3+\varepsilon})$ шагов для разложения числа $n$ на множители.

Подходящее число $D$ может быть найдено очень просто. Положим $x_0 = \left\lfloor \sqrt{n - n^{2/3}} \right\rfloor$, тогда, если $d = n - x_0^2$, то $n^{2/3} \leqslant d < n^{2/3} + 2n^{1/2}$. Возьмем $D = -4d$. Заметим, что квадратичная форма $(1, 0, d)$ уже является приведенной и представляет число $n$ при $x = x_0, y = 1$. Это представление дает пару $(2x_0, 1)$ в множестве $\mathcal{S}(D, n)$. Следовательно, мы уже получили одну из двух нужных нам пар. Более того, если число $n$ делится хотя бы на 2 нечетных простых числа, не делящих число $d$, то существуют два решения $h_1, h_2$ сравнения $h^2 \equiv D \pmod{4n}$, такие что $h_1 \not\equiv \pm h_2 \pmod{n}$. Поэтому поиск второй нужной нам пары в множестве $\mathcal{S}(D, n)$ будет удачным, и мы, вместе с парой $(2x_0, 1)$, получим разложение числа $n$.

Следующий алгоритм подводит итог всему вышесказанному.

**Алгоритм 5.6.3** (тест Макки). Нам дано число $n > 1$, не имеющее простых делителей, меньших $3n^{1/3}$. Алгоритм определяет, является число $n$ простым или составным, и в случае составного числа выдает разложение на простые сомножители числа $n$. (Любое нетривиальное разложение на множители будет являться разложением на простые сомножители, так как все простые делители числа $n$ превосходят кубический корень из $n$.)

1. [Проверка на полный квадрат]

Если число $n$ является полным квадратом, например, $p^2$, то возвратить разложение $p \cdot p$;

// Проверить число на полный квадрат можно при помощи алгоритма 9.2.11.

2. [Вспомогательное разложение на множители]

$d = n - \left\lfloor \sqrt{n - n^{2/3}} \right\rfloor^2$; // Т. е. все простые делители числа $n$ // превосходят $2\sqrt{d}$.

if (НОД$(n, d) > 1$) возвратить разложение НОД$(n, d) \cdot (n/$НОД$(n, d))$;

Пробным делением находим полное разложение на простые сомножители числа $d$;

3. [Сравнения]

for($1 \leqslant a \leqslant \left\lfloor 2\sqrt{d/3} \right\rfloor$) {

Используя разложение на простые сомножители числа $d$ и метод из разд. 2.3.2, найдем решения $u_1, \ldots, u_t$ сравнения $u^2 \equiv 4an \pmod{4d}$;

for($1 \leqslant i \leqslant t$) { // Если $t = 0$, то этот цикл не выполняется.

Для всех целых чисел $u$, таких что $0 \leqslant u \leqslant 2\sqrt{an}$, $u \equiv u_i \pmod{4d}$, используем алгоритм 9.2.11 для проверки, является ли число $(4an - u^2)/4d$ полным квадратом;

Если полный квадрат найден, например, $v^2$, и $u \not\equiv \pm 2x_0 v \pmod{2n}$, то перейти к шагу [Вычисление НОД];

}

}

return «число $n$ является простым»;

4. [Вычисление НОД]

$g =$ НОД$(2x_0 v - u, n)$;

возвратить разложение $g \cdot (n/g)$;

// Разложение нетривиально, и делители — простые числа.

**Теорема 5.6.4.** *Рассмотрим следующий алгоритм. Нам дано целое число $n > 1$, сначала выделим в $n$ все простые сомножители вплоть до $3n^{1/3}$ (при помощи пробного деления). Если число $n$ еще не будет полностью разложено на множители, то используем для оставшейся неразложенной части алгоритм* 5.6.3. *Таким образом мы получим полное разложение числа $n$ на множители. Для любого фиксированного $\varepsilon > 0$ время работы такого алгоритма для нахождения полного разложения числа $n$ на множители составляет $O(n^{1/3+\varepsilon})$.*

Еще один метод Макки с другим временем работы указан в упр. 5.21.

## 5.6.3. Композиция и группа классов

Пусть $D$ — целое число, не являющееся полным квадратом, $(a_1, b, c_1)$, $(a_2, b, c_2)$ — квадратичные формы дискриминанта $D$, и пусть $c_1/a_2$ является целым числом. Так как средние коэффициенты совпадают, то $a_1 c_1 = a_2 c_2$, т. е.

$c_1/a_2 = c_2/a_1$. Мы утверждаем, что произведение числа, представимого первой формой, и числа, представимого второй формой, будет представимо формой $(a_1a_2, b, c_1/a_2)$. Для доказательства этого утверждения достаточно проверить тождество

$$\left(a_1x_1^2 + bx_1y_1 + c_1y_1^2\right)\left(a_2x_2^2 + bx_2y_2 + c_2y_2^2\right) = a_1a_2x_3^2 + bx_3y_3 + (c_1/a_2)y_3^2,$$

где

$$x_3 = x_1x_2 - (c_1/a_2)y_1y_2, \quad y_3 = a_1x_1y_2 + a_2x_2y_1 + by_1y_2.$$

Таким образом, мы можем в некотором смысле объединить две формы $(a_1, b, c_1)$, $(a_2, b, c_2)$ дискриминанта $D$ в одну форму $(a_1a_2, b, c_1/a_2)$. Заметим, что эта форма также будет иметь дискриминант $D$. Продолжим определение *композиции* форм.

Будем говорить, что квадратичная форма $(a, b, c)$ является *примитивной*, если НОД$(a, b, c) = 1$. Пусть нам дано целое число $D$, не являющееся полным квадратом, но сравнимое с 0 или 1 (mod 4). Пусть $\mathcal{C}(D)$ обозначает множество классов эквивалентности примитивных бинарных квадратичных форм дискриминанта $D$. Каждый класс эквивалентности — это множество форм, эквивалентных данной. Класс эквивалентности, содержащий форму $(a, b, c)$, мы будем обозначать через $\langle a, b, c\rangle$.

**Лемма 5.6.5.** *Пусть* $\langle a_1, b, c_1\rangle = \langle A_1, B, C_1\rangle \in \mathcal{C}(D)$, $\langle a_2, b, c_2\rangle = \langle A_2, B, C_2\rangle \in \mathcal{C}(D)$, *и числа* $c_1/a_2, C_1/A_2$ *являются целыми. Тогда* $\langle a_1a_2, b, c_1/a_2\rangle = \langle A_1A_2, B, C_1/A_2\rangle$.

Доказательство см., например, в книге [Rose 1988].

**Лемма 5.6.6.** *Пусть* $(a_1, b_1, c_1), (a_2, b_2, c_2)$ — *примитивные квадратичные формы дискриминанта* $D$. *Тогда существуют форма* $(A_1, B, C_1)$, *эквивалентная* $(a_1, b_1, c_1)$, *и форма* $(A_2, B, C_2)$, *эквивалентная* $(a_2, b_2, c_2)$, *такие что* НОД$(A_1, A_2) = 1$.

Доказательство. Сначала покажем, что существуют взаимно простые целые числа $x_1, y_1$, такие что $a_1x_1^2 + b_1x_1y_1 + c_1y_1^2$ будет взаимно просто с $a_2$. Запишем $a_2$ в виде $a_2 = m_1m_2m_3$, где каждый простой делитель числа $m_1$ делит также и $a_1$, но не делит $c_1$; каждый простой делитель числа $m_2$ делит также и $c_1$, но не делит $a_1$; и каждый простой делитель числа $m_3$ делит также и НОД$(a_1, c_1)$. Найдем целые числа $u_1, v_1$, такие что $u_1m_1 + v_1m_2m_3 = 1$, и положим $x_1 = u_1m_1$. Найдем целые числа $u_2, v_2$, такие что $u_2m_2 + v_2m_3x_1 = 1$, и положим $y_1 = u_2m_2$. Тогда числа $x_1, y_1$ будут удовлетворять нужным нам свойствам.

Сделаем унимодулярную замену переменных $x = x_1X - Y$, $y = y_1X + v_2m_3Y$. Форма $(a_1, b_1, c_1)$ перейдет в эквивалентную форму $(A_1, B_1, C_1')$, где $A_1 = ax_1^2 + b_1x_1y_1 + c_1y_1^2$ взаимно просто с $a_2$. Для согласования коэффициентов $B_1$ и $b_2$ с утверждением леммы найдем целые числа $r, s$, такие что $rA_1 + sa_2 = 1$, и положим $k = r(b_2 - B_1)/2$. (Заметим, что числа $b_2$ и $B_1$ имеют ту же четность, что и $D$.) Положим $B = B_1 + 2kA_1$, тогда $B \equiv b_2 \pmod{2a_2}$. Тогда (см. упр. 5.18) форма $(A_1, B_1, C_1')$ эквивалентна $(A_1, B, C_1)$ при некотором целом $C_1$, и форма

$(a_2, b_2, c_2)$ эквивалентна $(a_2, B, C_2)$ при некотором целом $C_2$. Положим $A_2 = a_2$, и доказательство будет завершено. □

Пусть нам даны две примитивные квадратичные формы $(a_1, b_1, c_1), (a_2, b_2, c_2)$ дискриминанта $D$, пусть $(A_1, B, C_1), (A_2, B, C_2)$ — соответствующие эквивалентные формы из леммы 5.6.6. Определим следующую операцию:

$$\langle a_1, b_1, c_1\rangle * \langle a_2, b_2, c_2\rangle = \langle a_3, b_3, c_3\rangle,$$

где $a_3 = A_1A_2$, $b_3 = B$, $c_3 = C_1/A_2$. (Заметим, что из условий $A_1C_1 = A_2C_2$ и $\text{НОД}(A_1, A_2) = 1$ следует, что $C_1/A_2$ является целым числом.) Тогда лемма 5.6.5 утверждает, что бинарная операция «$*$» будет корректно определена на множестве $\mathcal{C}(D)$. Это операция композиции, на которую мы ссылались выше. Коммутативность операции $*$ очевидна, ассоциативность проверяется непосредственно. Если число $D$ четно, то форма $\langle 1, 0, D/4\rangle$ является единицей для операции $*$, если же $D$ нечетно, то единицей будет форма $\langle 1, 1, (1-D)/4\rangle$. Обозначим эту единицу через $1_D$. И, наконец, если форма $\langle a, b, c\rangle$ лежит в множестве $\mathcal{C}(D)$, то $\langle a, b, c\rangle * \langle c, b, a\rangle = 1_D$ (см. упр. 5.20). Следовательно, множество $\mathcal{C}(D)$ с операцией $*$ является абелевой группой. Эта группа называется группой классов примитивных бинарных квадратичных форм дискриминанта $D$.

Из всего сказанного выше можно получить алгоритм для вычисления композиции двух форм. Ниже приведена относительно простая процедура вычисления композиции. Ее можно найти в работах [Shanks 1971] и [Schoof 1982].

**Алгоритм 5.6.7** (композиция форм). **Нам даны две примитивные квадратичные формы $(a_1, b_1, c_1), (a_2, b_2, c_2)$ одинакового отрицательного дискриминанта. Алгоритм вычисляет целые числа $a_3, b_3, c_3$, такие что $\langle a_1, b_1, c_1\rangle * \langle a_2, b_2, c_2\rangle = \langle a_3, b_3, c_3\rangle$.**

1. [Расширенный алгоритм Евклида]
   $g = \text{НОД}(a_1, a_2, (b_1 + b_2)/2)$;
   Найдем $u, v, w$, такие что $ua_1 + va_2 + w(b_1 + b_2)/2 = g$;
2. [Итоговое присваивание]
   Возвратить ответ:

$$a_3 = \frac{a_1a_2}{g^2}, \quad b_3 = b_2 + 2\frac{a_2}{g}\left(\frac{b_1 - b_2}{2}v - c_2w\right), \quad c_3 = \frac{b_3^2 - g}{4a_3}.$$

(Для нахождения чисел $g, u, v, w$ на шаге [Расширенный алгоритм Евклида] используем алгоритм 2.1.4 сначала для нахождения чисел $U, V$, таких что $h = \text{НОД}(a_1, a_2) = Ua_1 + Va_2$; потом для нахождения чисел $U', V'$, таких что $g = \text{НОД}(h, (b_1+b_2)/2)) = U'h+V'(b_1+b_2)/2$. Тогда $u = U'U$, $v = U'V$, $w = V'$.) Отметим, что даже если исходные формы $(a_1, b_1, c_1), (a_2, b_2, c_2)$ были приведенными, получившаяся форма $(a_3, b_3, c_3)$ не обязательно будет таковой. Для получения приведенной формы из класса $\langle a_3, b_3, c_3\rangle$ после применения алгоритма 5.6.7 можно применить алгоритм 5.6.2.

Если дискриминант $D < 0$, то из теоремы 5.6.1 вытекает, что группа $\mathcal{C}(D)$ конечна. В самом деле, каждый элемент группы $\mathcal{C}(D)$ соответствует единствен-

ной приведенной форме $(a, b, c)$, удовлетворяющей условиям (5.2). Следовательно, $h(D)$, порядок группы $\mathcal{C}(D)$, равен количеству взаимно простых троек $a, b, c$, удовлетворяющих условиям (5.2) и таких, что $b^2 - 4ac = D$. Из неравенства $|b| \leqslant a$ мы получим $-D = 4ac - b^2 \geqslant 4ac - a^2$, и из $a \leqslant c$ получим $-D \geqslant 3a^2$. Следовательно, $0 < a \leqslant \sqrt{|D|/3}$. Так как число $c$ определяется однозначно после выбора чисел $a, b$, в итоге получаем $h(D) \leqslant \sum 2a < 2|D|/3$.

Однако эту оценку можно улучшить. Возьмем целое число $b$, такое что $|b| \leqslant \sqrt{|D|/3}$ и $b \equiv D \pmod 2$. Тогда количество возможных чисел $a$, соответствующих этому числу $b$, не превосходит количества делителей числа $b^2 - D$. Количество делителей числа $n$ асимптотически равно $n^{o(1)}$ при $n \to \infty$, следовательно, $h(D) \leqslant |D|^{1/2+o(1)}$ при $D \to -\infty$.

И эту оценку можно улучшить. Знаменитая формула Дирихле для числа классов (см. [Davenport 1980][2]) утверждает, что при $D < 0$ и $D \equiv 0$ или $1 \pmod 4$

$$h(D) = \frac{w}{\pi} L(1, \chi_D)\sqrt{|D|}, \tag{5.3}$$

где $w = 3$, если $D = -3$; $w = 2$, если $D = -4$; $w = 1$ в остальных случаях. Характер $\chi_D$ является символом Кронекера $(D/\cdot)$. Он определяется следующим образом: $\chi_D$ — мультипликативная функция, $\chi_D(p)$ равно символу Лежандра $(D/p)$ для нечетных простых чисел $p$; $\chi_D(2)$ равно 0, если $D$ четно, равно 1, если $D \equiv 1 \pmod 8$, и равно $-1$, если $D \equiv 5 \pmod 8$. $L$-функция $L(s, \chi_D)$ обсуждается в разд. 1.4.3; $L(1, \chi_D)$ равно сумме бесконечного ряда $\sum \chi_D(n)/n$. В 1918 г. Шур показал, что $L(1, \chi_D) < \frac{1}{2}\ln|D| + \ln\ln|D| + 1$, поэтому $\frac{w}{\pi} L(1, \chi_D) < \ln|D|$ при $D \leqslant -4$. Следовательно, $h(D) < \sqrt{|D|}\ln|D|$ для этих значений дискриминанта $D$. Так как $h(-3) = 1$, это неравенство выполняется и для $D = -3$, т. е. оно выполняется для всех отрицательных дискриминантов.

К. Зигель показал, что $h(D) = |D|^{1/2+o(1)}$ при $D \to -\infty$, но его доказательство неэффективно. Т. е. используя это доказательство, невозможно, например, найти верхнюю границу чисел $|D|$, таких что $h(D) < 1000$, хотя теорема утверждает, что такая граница существует. После работ Голдфельда, Гросса и Цагира Остерле (см. [Oesterlé 1985]; см. также [Watkins 2004]) установил явное неравенство

$$h(D) > \frac{1}{7000}\ln|D| \prod_p \left(1 - \frac{\lfloor 2\sqrt{p} \rfloor}{p+1}\right),$$

где произведение берется по всем простым числам $p$, делящим $D$ и меньшим $\sqrt{|D|/4}$. Объединив этот результат с тем фактом, что $2^{k-1} | h(D)$, где $k$ — количество различных нечетных простых делителей числа $D$ (см. лемму 5.6.8), мы получим, например, что $h(D) > 1000$ при $-D > 10^{1.3 \cdot 10^{10}}$. Наверняка эта оценка очень далека от истины, но, тем не менее, она дает явную границу, что невозможно получить из теоремы Зигеля. Существует недоказанная гипотеза, более слабая, чем расширенная гипотеза Римана, а именно, что $L$-функции $L(s, \chi)$ не имеют *вещественных* нулей, больших $1/2$. В предположении этой

[2] Имеется перевод с 1-го издания: [Давенпорт 1971]. — *Прим. ред.*

гипотезы Татудзава (см. [Tatuzawa 1951]) получил неравенство, из которого следует, что $h(D) > 1000$ при $-D > 1.9 \cdot 10^{11}$. Однако даже эта сильно пониженная граница примерно в 100 раз превосходит предполагаемую. Вполне возможно, мы сможем получить примерно в 100 раз более точную границу при условии выполнения расширенной гипотезы Римана.

В вычислительном (и теоретическом) *tour de force* Уоткинс (см. [Watkins 2004]) безусловно показал, что $h(D) > 16$ при $-D > 31243$.

Следующая формула для $h(D)$ довольно привлекательна (но на самом деле не очень эффективна при больших $|D|$), так как бесконечная сумма для $L(1, \chi_D)$ в ней заменена конечной. Эту формулу получил Дирихле, см. [Narkiewicz 1986]. Если $D < 0$ и $D$ является фундаментальным дискриминантом (это означает, что либо $D \equiv 1 \pmod 4$ и $D$ бесквадратное, либо $D \equiv 8$ или $12 \pmod{16}$ и $D/4$ бесквадратное), то

$$h(D) = \frac{w}{D} \sum_{n=1}^{|D|} \chi_D(n) n.$$

Хоть эта формула и выглядит привлекательно, сумма состоит из $|D|$ слагаемых и пригодна для точного вычисления $h(D)$ лишь при малых значениях $|D|$, скажем, при $|D| < 10^8$. Существуют различные методы ускоренного вычисления таких сумм; например, в книге [Cohen 2000] можно найти формулы суммирования с вероятностной функцией ошибок и числом слагаемых всего $O(|D|^{1/2})$; такие формулы позволяют легко обрабатывать значения $|D| \approx 10^{16}$. Кроме того, можно показать, что прямым подсчетом примитивных приведенных форм $(a, b, c)$ отрицательного дискриминанта $D$ можно вычислить $h(D)$ за $O\left(|D|^{1/2+\epsilon}\right)$ операций. А метод **Шенкса** «детские шаги, гигантские шаги» понижает показатель степени с $1/2$ до $1/4$. Мы вернемся к вопросу о сложности вычисления $h(D)$ в следующем разделе.

## 5.6.4. Амбиговы формы и разложение на множители

Нетрудно выписать все элементы группы классов $\mathcal{C}(D)$, которые являются обратными сами себе. Для $D < 0$ приведенная форма каждого такого класса называется «амбиговой» формой. Амбиговы формы бывают трех типов: $(a, 0, c), (a, a, c), (a, b, a)$. Эти формы тесно связаны с разложением дискриминанта на два взаимно простых множителя.

Мы приведем классификацию амбиговых форм, но оставим ее простую проверку читателю.

**Лемма 5.6.8.** *Пусть $D$ — отрицательный дискриминант. Если $D$ четно, то амбиговы формы дискриминанта $D$ содержат формы $(u, 0, v)$, где $0 < u \leqslant v$, $\text{НОД}(u, v) = 1$, и $uv = -D/4$. Кроме того, если $uv = -D/4$, причем $\text{НОД}(u, v) = 1$ или 2 и число $\frac{1}{2}(u+v)$ нечетно, то мы получим формы $\left(\frac{1}{2}(u+v), v-u, \frac{1}{2}(u+v)\right)$, когда $\frac{1}{3}v \leqslant u < v$, и формы $\left(2u, 2u, \frac{1}{2}(u+v)\right)$, когда $0 < u < \frac{1}{3}v$. Если $D$ нечетно, то амбиговыми формами дискриминанта $D$ являются формы $\left(\frac{1}{4}(u+v), \frac{1}{2}(v-u), \frac{1}{4}(u+v)\right)$, когда $-D = uv$, $0 < \frac{1}{3}v \leqslant u \leqslant v$,*

НОД$(u,v) = 1$; *и формы* $\left(u, u, \frac{1}{4}(u+v)\right)$, *когда* $-D = uv$, $0 < u \leqslant \frac{1}{3}v$, НОД$(u,v) = 1$.

Заметим, что форма $(1, 0, |D|/4)$, если $D$ четно, и форма $(1, 1, (1-D)/4)$, если $D$ нечетно, являются амбиговыми. Как было сказано в предыдущем разделе, каждая из этих форм в соответствующем случае является приведенной формой из класса $1_D$. Они соответствуют тривиальному разложению на множители чисел $D/4$ или $D$. Если $D \equiv 12 \pmod{16}$ и $D \leqslant -20$, то амбигова форма $(2, 2, (4-D)/8)$ также соответствует тривиальному разложению на множители числа $D/4$. Тривиальному разложению числа $D/4$ соответствуют еще амбиговы формы $(4, 4, 1 - D/16)$, если $D \equiv 0 \pmod{32}$ и $D \leqslant -64$, и форма $(3, 2, 3)$ с дискриминантом $-32$. Однако все остальные амбиговы формы соответствуют нетривиальным разложениям чисел $D/4$ или $D$. Предположим, что число $D$ имеет $k$ различных нечетных простых делителей. Из леммы 5.6.8 следует, что существует $2^{k-1}$ амбиговых форм дискриминанта $D$. Если же $D \equiv 12 \pmod{16}$ или $D \equiv 0 \pmod{32}$, то существует $2^k$ и $2^{k+1}$ амбиговых форм, соответственно.

Предположим теперь, что $n$ — положительное нечетное целое число, делящееся как минимум на два различных простых числа. Если $n \equiv 3 \pmod{4}$, то $D = -n$ является дискриминантом, если же $n \equiv 1 \pmod{4}$, то $D = -4n$ является дискриминантом. В первом случае, если мы найдем амбигову форму, отличную от $(1, 1, (1+n)/4)$, то мы получим нетривиальное разложение числа $n$ на множители. Во втором случае, если мы найдем амбигову форму, отличную от $(1, 0, n)$ и $(2, 2, (1+n)/2)$, то мы получим нетривиальное разложение числа $n$ на множители. И в обоих случаях, если мы найдем все амбиговы формы, то мы сможем получить полное разложение числа $n$ на множители.

Следовательно, мы можем считать, что задача разложения на множители свелась к задаче нахождения амбиговых форм.

Покажем теперь, как можно найти амбиговы формы с данным отрицательным дискриминантом $D$. Пусть $h = h(D)$ обозначает число классов, т. е. порядок группы $\mathcal{C}(D)$ (см. разд. 5.6.3). Пусть $h = 2^t h_0$, где $h_0$ нечетно. Если $f = \langle a, b, c\rangle \in \mathcal{C}(D)$, то положим $F = f^{h_0}$. Тогда либо $F = 1_D$, либо одна из форм $F, F^2, F^4, \ldots, F^{2^{t-1}}$ имеет порядок 2 в этой группе. Приведенный элемент из класса порядка 2 является амбиговой формой (по определению). Поэтому, зная $h$ и $f$, мы можем довольно просто получить амбигову форму. Если полученная амбигова форма соответствует $1_D$ или равна $(2, 2, (1+n)/2)$ (в случае $n \equiv 1 \pmod{4}$), то соответствующее разложение на множители будет тривиальным. В остальных случаях мы получим нетривиальное разложение.

Так что, если приведенная схема не работает с одной $f$ из $\mathcal{C}(D)$, то мы, вероятно, можем попытаться еще раз с другой формой $f$. Если бы мы имели небольшое множество порождающих элементов группы классов, мы могли бы возобновлять попытку, выбирая каждый раз новый порождающий элемент, и в итоге разложили бы $n$. (На самом деле, в этом случае мы имели бы достаточное количество амбиговых форм для нахождения полного разложения $n$ на простые сомножители, если уточнять различные разложения с помощью

взятия НОД.) Если такого малого множества порождающих элементов нет в наличии, можно выбирать $f$ случайным образом.

Главным препятствием на пути применения этой схемы для разложения $n$ является то, что надо предоставить не подходящую $f$ из $\mathcal{C}(D)$, а число классов $h$. На самом деле можно потребовать несколько меньше. Все, что нам требуется при реализации вышеуказанной идеи, — это порядок $f$ в группе классов.

А теперь, на минуту забыв об этом, и фактически переходя к полному порядку $h$ группы классов, мы можем решить, что, поскольку у нас есть формула (5.3) для порядка этой группы, то цель достигнута. Однако в эту формулу входит бесконечная сумма, и не совсем ясно, сколько членов необходимо взять, чтобы достичь желаемой точности.

Заметим, что бесконечная сумма $L(1,\chi_D)$ в формуле (5.3) для числа классов может быть также записана в виде бесконечного произведения:

$$L(1,\chi_D)=\prod_p\left(1-\frac{\chi_D(p)}{p}\right)^{-1},$$

где произведение берется по всем простым числам. В работах [Shanks 1971], [Schoof 1982] показано, что если предположить справедливость расширенной гипотезы Римана (ERH) (см. гипотезу 1.4.2), и если

$$\tilde{L}=\prod_{p\leqslant n^{1/5}}\left(1-\frac{\chi_D(p)}{p}\right)^{-1},\quad \tilde{h}=(w/\pi)\sqrt{|D|}\tilde{L},$$

то существует эффективная постоянная $c$, такая, что $|h-\tilde{h}|<cn^{2/5}\ln^2 n$. Если мы возьмем на себя вычисление $\tilde{L}$ с некоторой точностью, то затем нам придется взять на себя оценку отклонения $\tilde{h}$ от числа классов $h$, которые могут отличаться не более чем на $cn^{2/5}\ln^2 n$. Далее, для нахождения величины, кратной порядку любой заданной $f\in\mathcal{C}(D)$, лежащей в интервале $(\tilde{h}-cn^{2/5}\ln^2 n,\tilde{h}+cn^{2/5}\ln^2 n)$, за время $O(n^{1/5}\ln n)$ может быть использован метод Шенкса «детские шаги, гигантские шаги» (см. разд. 7.5 и 5.3). Поскольку вычисление $\tilde{L}$ может быть выполнено за $O(n^{1/5})$ шагов, мы можем добиться разложения числа $n$ при подходящем выборе $f$ за $O(n^{1/5}\ln n)$ операций с целыми числами порядка $n$.

Если Вы желаете предположить справедливость гипотезы ERH, что считается вполне честной игрой в алгоритме разложения на множители (если методу не удастся разложить число, то Ваши усилия увенчаются опровержением ERH, что, по-видимому, представляет гораздо больший интерес, чем попытка разложения), то Вы могли бы спросить, какую еще информацию может дать ERH, кроме предсказуемой сходимости произведения для $L(1,\chi_D)$. На самом деле, она может оказать еще одну услугу. В предположении гипотезы ERH существует эффективная постоянная $c'$, такая, что классы примитивных приведенных форм $(a,b,c)$ дискриминанта $D$ при $a\leqslant c'\ln^2|D|$ порождают всю группу классов $\mathcal{C}(D)$ (см. [Schoof 1982]). Таким образом, в приведенном сценарии не обязана присутствовать неопределенность в выборе $f$. А именно, просто выбирайте $f$ с таким представителем $(a,b,c)$, что $a\leqslant c'\ln^2|D|$.

Собирая вместе эти составляющие, мы получаем детерминированный алгоритм разложения со сложностью $O\left(n^{1/5}\ln^3 n\right)$ операций с целыми числами порядка $n$. Обоснование корректности этого алгоритма опирается на недоказанную гипотезу ERH.

Шенкс идет еще дальше и показывает, что в предположении ERH можно вычислить число классов $h$ и структуру группы $\mathcal{C}(D)$ за время $O\left(|D|^{1/5+\epsilon}\right)$.

Сринивасан (см. [Srinivasan 1995]) показала, что существует вероятностный алгоритм, приближающий $L$, причем ожидается, что он даст достаточную точность приближения $h$, и погрешность вновь составит $O\left(|D|^{2/5+\epsilon}\right)$, после чего эстафету может принять метод Шенкса «детские шаги, гигантские шаги». Вероятностный метод Сринивасан получает приближенное значение за ожидаемое время $O\left(|D|^{1/5+\epsilon}\right)$ и как следствие становится вероятностным алгоритмом разложения с ожидаемым временем работы $O\left(n^{1/5+\epsilon}\right)$. Данный алгоритм является абсолютно строгим, не зависящим от недоказанных гипотез. Ее метод также вычисляет число классов и структуру группы за ожидаемое время $O\left(|D|^{1/5+\epsilon}\right)$. Однако в отличие от разложения на множители, корректность которого легко проверяется, не существует простого способа выяснить, является ли корректным предложенный Сринивасан способ вычисления числа классов, хотя почти наверняка это так. Как мы увидим в следующей главе, существуют более быстрые, строго обоснованные вероятностные алгоритмы разложения. Метод Сринивасан, однако, остается самым быстрым из известных строго обоснованных вероятностных методов вычисления числа классов группы $\mathcal{C}(D)$. (Хафнеру и Маккерли (см. [Hafner and McCurley 1989]) принадлежит субэкспоненциальный вероятностный метод, но его анализ опирается на ERH.)

## 5.7. Упражнения

**5.1.** На основе алгоритма Ленстры 4.2.11 разработайте детерминированный метод разложения на множители, требующий не более $n^{1/3+o(1)}$ операций для разложения числа $n$.

**5.2.** Пусть моделью итерации отображения $x^2+a \bmod p$ в ро-методе Полларда служит случайная функция $f$ из $\{0,1,\ldots,p-1\}$ в $\{0,1,\ldots,p-1\}$. Функция $f$ задает ориентированный граф на вычетах по модулю $p$, где из вычета $i$ исходит единственная стрелка, соединенная с $f(i)$. Покажите, что ожидаемая длина самого длинного пути $r_1, r_2, \ldots, r_k$, состоящего из различных вычетов, имеет порядок $\sqrt{p}$. Укажем возможную стратегию: если $s_1, s_2, \ldots, s_j$ есть путь, состоящий из различных вычетов, то вероятность того, что $f(s_j) \notin \{s_1,\ldots,s_j\}$ есть $(p-j)/p$. Так что вероятность того, что путь, ведущий из $s$, попадает в различные точки на протяжении $j$ или более шагов, равна произведению величин $(p-i)/p$ по $i=1,2,\ldots,j$. Искомое ожидание, таким образом, есть $\sum_{j=0}^{p-1}\prod_{i=1}^{j}(p-i)/p$. См. [Purdom and Williams 1968].

Затем исследуйте ситуацию, которая более адекватно отражает ро-метод Полларда для разложения на множители; в ней мы предполагаем, что случай-

ная функция $f$ отображает в соотношении $2:1$ или, в более общем случае, $2K:1$ (см. упр. 5.24). В связи с этим см. [Brent and Pollard 1981] и [Arney and Bender 1982].

**5.3.** Одним из фактов, используемых при анализе ро-метода Полларда, является то, что функция $f(x) = x^2 + a$ из $\mathbf{Z}_n$ в $\mathbf{Z}_n$ обладает следующим свойством: для каждого делителя $d$ числа $n$ мы имеем, что из $u \equiv v \pmod{d}$ следует, что $f(u) \equiv f(v) \pmod{d}$. Докажите, что любой многочлен $f(x)$ из $\mathbf{Z}_n[x]$ обладает этим свойством. Покажите, что обратное неверно. А именно, существуют натуральные $n$ и отличные от многочленов функции $f$ из $\mathbf{Z}_n$ в $\mathbf{Z}_n$, обладающие тем свойством, что $f(u) \equiv f(v) \pmod{d}$ как только выполнены условия $d|n$ и $u \equiv v \pmod{d}$. (Авторы приносят свою благодарность К. Харе, указавшему, что такие функции существуют.)

**5.4.** Пусть $G$ есть циклическая группа порядка $n$ с образующим $g$ и элементом $t$. Пусть нашей целью является нахождение дискретного логарифма $l$ элемента $t$, т. е. такого целого числа $l$, что $g^l = t$. Предположим, что мы каким-либо образом обнаружили, что $g^b = t^a$. Покажите, что искомый логарифм выражается по формуле

$$l = ((bu + kn)/d) \bmod n$$

для некоторого целого числа $k \in [0, d-1]$, где $d = \text{НОД}(a, n)$, а $u$ является решением расширенного соотношения Евклида $au + nv = d$.

Данное упражнение показывает, что нахождение логарифма нетривиальной степени элемента $t$ при не очень большом $d$ по существу эквивалентно исходной задаче дискретного логарифмирования (DL).

**5.5.** Пусть $G$ — конечная циклическая группа; Вам известен ее порядок $n$ и разложение числа $n$ на простые сомножители. Покажите, как можно использовать метод Шенкса «детские шаги, гигантские шаги» из разд. 5.3 для решения задач дискретного логарифмирования в $G$ за $O\left(\sqrt{p}\ln n\right)$ операций, где $p$ — наибольший простой делитель числа $n$. Получите такого же типа оценку для требуемой памяти.

**5.6.** Как мы выяснили в настоящей главе, в чистом виде процедуру Шенкса «детские шаги, гигантские шаги» можно сформулировать следующим образом: создайте соответствующие списки для детских шагов и гигантских шагов, отсортируйте один из списков, а затем найдите совпадение путем последовательного просмотра другого списка. Как нам известно, решение уравнения $g^l = t$ (где $g$ — порождающий элемент циклической группы порядка $n$ и $t$ — некоторый элемент) можно осуществить указанным способом за $O(n^{1/2}\ln n)$ операций (сравнений). Но существует так называемое построение с хэш-таблицей, которое несколько изменяет эту сложность (хотя и незначительно), и на практике работает весьма эффективно. В общих чертах такой метод работает следующим образом:

(1) Построить список детских шагов, но сделать это в виде хэш-таблицы.
(2) На каждом последовательном гигантском шаге осуществить (быстрый) просмотр соответствующей позиции в хэш-таблице в поиске совпадения.

Целью настоящего упражнения является изучение — с помощью компьютера — следующего примера решения реальной задачи дискретного логарифмирования (DL). Данный пример, в отличие от фундаментального алгоритма 5.3.1, использует некоторые трюки, основанные на способах функционирования машин, за счет чего происходит эффективное понижение сложности. Для простого числа $p = 2^{31} - 1$ и явно поставленной задачи DL, состоящей, скажем, в решении сравнения

$$g^l \equiv t \pmod{p},$$

поступаем следующим образом. По аналогии с алгоритмом 5.3.1 положим $b = \lceil \sqrt{p} \rceil$, но в дополнение к этому выберем специальный параметр $\beta = 2^{12}$ для создания «хэш-таблицы» детских шагов, $r$-я строка которой при $r \in [0, \beta - 1]$ состоит из всех тех вычетов $g^j \bmod p$ при $j \in [0, b - 1]$, для которых $r = (g^j \bmod p) \bmod \beta$. Это означает, что строка хэш-таблицы, в которую помещается степень $g^j \bmod p$, зависит *только* от младших $\lg \beta$ битов этого степенного вычета. Итак, примерно за $\sqrt{p}$ умножений (последовательно на $g$) мы составляем хэш-таблицу, состоящую из $\beta$ строк. Для проверки в случае попытки программирования укажем, что при конкретном выборе $g = 7$ ($r = 1271$)-я строка должна выглядеть как

$$((704148727, 507),\ (219280631, 3371),\ (896259319, 4844)\ldots),$$

что означает, к примеру,

$$\begin{aligned} 7^{507} \bmod p &= 704148727 = (\ldots 010011110111)_2, \\ 7^{3371} \bmod p &= 219280631 = (\ldots 010011110111)_2, \end{aligned}$$

и т. д. После того как хэш-таблица детских шагов построена, можно просматривать члены гигантских шагов $tg^{-ib}$ при $i \in [0, b - 1]$ и, анализируя лишь младшие 12 битов каждого из таких членов, осуществлять прямой доступ к таблице по индексу и поиск совпадения. Например, при $t = 31$ это непосредственно приводит к решению задачи DL

$$7^{723739097} \equiv 31 \pmod{2^{31} - 1}.$$

Данное упражнение служит хорошей отправной точкой для разработки общей программы решения задачи DL, получающей на входе произвольные $p, g, l, t$, а затем выбирающей оптимальные параметры вроде $\beta$. Кстати говоря, подходы с хэш-таблицей, подобные этому, обладают интересной особенностью: в сущности, им требуется память для хранения *одного* списка, а не двух. Более того, если доступ к хэш-таблице по индексу рассматривать как одну фундаментальную операцию, то сложность алгоритма в операциях равна $O(p^{1/2})$, т. е. убирается множитель $\ln p$. Отметим также еще одно преимущество: однажды построенная хэш-таблица может повторно использоваться для других DL-вычислений (при условии, что $g$ не меняется).

**5.7.** [E. Teske]. Пусть $g$ — образующий элемент конечной циклической группы $G$, и пусть $h \in G$. Положим $\#G = 2^m \cdot n$, где $m \geqslant 0$ и $n$ нечетно. Рассмотрим следующее блуждание:

$$h_0 = g * h, \qquad h_{k+1} = {h_k}^2.$$

Члены $h_k$ вычисляются до тех пор, пока мы не получим $h_k = h_j$ для некоторого $j < k$ или $h_k = 1$. Исследуйте, будет ли это блуждание полезным для вычисления дискретных логарифмов.

(1) Пусть $(\alpha_k)$ и $(\beta_k)$ — последовательности показателей для $g$ и $h$ соответственно. Таким образом, $h_k = g^{\alpha_k} * h^{\beta_k}$ для каждого $k$. Определите явные формулы для $\alpha_k$ и $\beta_k$.
(2) Определите все возможные элементы группы $h$, для которых может оказаться, что $h_k = 1$ для некоторого $k$. Определите наибольшее возможное значение $k$, для которого это может случиться.
(3) Определите период $\lambda$ последовательности $(h_k)$ в предположении, что $\#G$ — простое число.
(4) Рекомендовали бы Вы использовать данное блуждание для вычисления дискретного логарифма? Если да, то почему? Если нет, то почему нет?

**5.8.** Здесь представлены задачи для практического тестирования произвольной реализации $(p-1)$-метода (алгоритм 5.4.1).

(1) Используйте базовый алгоритм с границей поиска $B = 1000$ для получения разложения

$$n = 67030894509517639 = 179424673 \cdot 373587943.$$

(2) Объясните, отталкиваясь от разложения числа 373587942, *почему* Ваше значение $B$ сработало.
(3) Снова, отталкиваясь от разложения числа 373587942, напишите версию алгоритма со второй стадией, и на этот раз найдите делитель при $B = 100$, в то время как граница второй стадии $B' = 1000$. Конечно, эта программа должна работать быстрее первой.
(4) Найдите нетривиальный делитель числа $M_{67} = 2^{67} - 1$, используя $B = 100$, $B' = 3000$ (или просто используйте $B = 3000$ без второй стадии).
(5) Модифицируйте алгоритм 5.4.1 так, что для каждого простого числа $p_i \leqslant B$ мы берем в качестве показателя $a_i$ наибольшее целое число, такое, что $p_i^{a_i} < n$. Используйте этот вариант алгоритма с $B = 100$ и $B' = 1000$ для разложения на множители целого числа $n = 670308837440379259$.

**5.9.** Здесь мы описываем интересный способ реализации второй стадии и заканчиваем постановкой вычислительного вопроса, также представляющего интерес. Мы видели, что вторая стадия имеет смысл, если неизвестный простой делитель $p$ числа $n$ имеет вид $p = zq + 1$, где $z$ является $B$-гладким и $q \in (B, B']$ — единственное простое число, выходящее за предел. Один новый подход (см. [Montgomery 1992a], [Crandall 1996a]) к реализации второй стадии состоит в следующем: после вычисления $b = a^{M(B)} \bmod n$ на первой стадии согласно описанному в тексте, в качестве второй стадии можно накапливать некоторое произведение (здесь $g, h$ пробегают некоторый фиксированный промежуток или соответствующие множества), например,

$$c = \prod_{g \neq h} \left( b^{g^K} - b^{h^K} \right) \bmod n$$

и вычислять НОД$(n, c)$ в надежде обнаружить нетривиальный делитель. Возникающая здесь теоретическая задача — объяснить, *почему* данный метод позволяет обнаружить выходящее за предел простое число $q$ и указать примерную вероятность (в терминах $q, K$ и интервалов для $g, h$) обнаружения делителя благодаря сравнению $g^K \equiv h^K \pmod{q}$.

Интересный вычислительный вопрос, возникающий в связи с «$g^K$-методом», состоит в том, как быстро можно вычислить цепочку

$$b^{1^K}, b^{2^K}, b^{3^K}, \ldots, b^{A^K},$$

где каждый член, как обычно, берется по модулю $n$? Найдите алгоритм, который фактически генерирует указанную цепочку «гиперстепеней» при фиксированном $K$ всего за $O(A)$ операций в кольце $\mathbf{Z}_N$.

**5.10.** Покажите, что эквивалентность квадратичных форм есть отношение эквивалентности.

**5.11.** Если две квадратичные формы $ax^2 + bxy + cy^2$ и $a'x^2 + b'xy + c'y^2$ имеют одинаковые интервалы значений, обязательно ли коэффициенты $(a', b', c')$ связаны с коэффициентами $(a, b, c)$ по формуле (5.1), где $\alpha, \beta, \gamma, \delta$ — целые числа и $\alpha\delta - \beta\gamma = \pm 1$?

**5.12.** Покажите, что эквивалентные квадратичные формы имеют одинаковый дискриминант.

**5.13.** Покажите, что квадратичная форма, получающаяся на выходе алгоритма 5.6.2, эквивалентна квадратичной форме, задаваемой на входе.

**5.14.** Покажите, что, если $(a, b, c)$ — приведенная квадратичная форма дискриминанта $D < 0$, то $a \leqslant \sqrt{|D|/3}$.

**5.15.** Покажите, что если на входе задано $(A, B, C)$, то сложность алгоритма 5.6.2 в операциях равна $O(1 + \ln(\min\{A, C\}))$, причем операции производятся над целыми числами, не превосходящими $4AC$.

**5.16.** Покажите, что натуральное число $n$ есть сумма двух квадратов тогда и только тогда, когда не существует простого числа $p \equiv 3 \pmod 4$, которое входит в разложение $n$ в нечетной степени. Используя тот факт, что сумма обратных величин к простым числам, сравнимым с 3 (mod 4), расходится (теорема 1.1.5), докажите, что множество натуральных чисел, представимых в виде суммы двух квадратов, имеет асимптотическую плотность 0. (См. упр. 1.10, 1.91 и 3.17.)

**5.17.** Покажите, что, если $p$ — простое и $p \equiv 1 \pmod 4$, то существует вероятностный алгоритм, позволяющий записать $p$ в виде суммы двух квадратов, ожидаемое время работы которого полиномиально. В случае $p \equiv 5 \pmod 8$ покажите, как можно превратить этот алгоритм в детерминированный. С использованием детерминированного полиномиального метода для нахождения квадратных корней из $-1$ по модулю $p$ из работы [Schoof 1985] покажите, как можно превратить этот алгоритм в детерминированный в общем случае, сохранив полиномиальное время работы.

**5.18.** Пусть $(a,b,c)$, $(a',b',c')$ — эквивалентные квадратичные формы, $n$ — натуральное число, $ax^2+bxy+cy^2=n$, и в результате эквивалентности $x,y$ заменяются на $x',y'$. Пусть $u=2ax+by$, $u'=2a'x'+y'$. Покажите, что $uy'\equiv u'y \pmod{2n}$.

**5.19.** Покажите, что, если $(a,b,c)$ — квадратичная форма, то для каждого целого числа $b'\equiv b \pmod{2a}$ существует целое число $c'$, такое, что форма $(a,b,c)$ эквивалентна форме $(a,b',c')$.

**5.20.** Пусть $\langle a,b,c\rangle \in \mathcal{C}(D)$. Докажите, что $\langle a,b,c\rangle$ есть единица $1_D$ в $\mathcal{C}(D)$ тогда и только тогда, когда форма $(a,b,c)$ представляет 1. Выведите отсюда, что $\langle a,b,c\rangle * \langle c,b,a\rangle = 1_D$.

**5.21.** Исследуйте и реализуйте алгоритм разложения Макки со сложностью $O(n^{1/4+\epsilon})$, как описано в работе [McKee 1999]. Метод является вероятностным и представляет собой своего рода оптимизацию знаменитого метода Ферма.

**5.22.** На основании формулы Дирихле для числа классов (5.3) выведите следующие формулы для числа $\pi$:

$$\pi = 2\prod_{p>2}\left(1+\frac{(-1)^{(p-1)/2}}{p}\right)^{-1} = 4\prod_{p>2}\left(1-\frac{(-1)^{(p-1)/2}}{p}\right)^{-1}$$

Просто используя тот факт, что эти формулы верны, убедитесь, что существует бесконечно много простых чисел вида $p=4k+1$ и вида $p=4k+3$. (Ср. упр. 1.7.) Вопрос вычислительного характера: сколько примерно простых чисел необходимо взять, чтобы вычислить приближенное значение $\pi$ с заданным числом десятичных знаков после запятой?

## 5.8. Проблемы для исследования

**5.23.** Покажите, что для $p=257$ $\rho$-итерация $x=x^2-1 \bmod p$ имеет лишь три возможные длины цикла, а именно, 2, 7, 12. Для $p=7001$ покажите, что итерация $x=x^2+3 \bmod p$ имеет лишь следующие 8 длин циклов: 3, 4, 6, 7, 19, 28, 36, 67. Найдите также число различных компонент связности у цикловых графов этих двух итераций. Верно ли, что число различных длин циклов, а также число компонент связности (которое всегда по меньшей мере так же велико) есть $O(\ln p)$? Результат, похожий на этот, доказан для случайной функции; см. [Flajolet and Odlyzko 1990].

**5.24.** Если в качестве итерации в ро-методе Полларда взять не $x=(x^2+a) \bmod N$, а

$$x=\left(x^{2K}+a\right) \bmod N,$$

то эвристически установлено, что ожидаемое количество итераций для нахождения простого делителя $p$ числа $N$ понижается с $c\sqrt{p}$ до

$$\frac{c\sqrt{p}}{\sqrt{\text{НОД}\,(p-1,2K)-1}}.$$

Прежде чем исследовать это понижение количества итераций, возможно, было бы полезным сначала изучить этот эвристический подход и исследовать некоторые возможные реализации метода, основанные на этом НОД-понижении (см. [Brent and Pollard 1981], [Montgomery 1987], [Crandall 1999d]). Заметим, что если мы *знаем* что-либо о числе $K$, то мы получим заметное ускорение алгоритма, как это происходит в приложениях ро-методов Полларда к числам Ферма или Мерсенна. (Если число $K$ мало, то использование итерации $x = x^{2K} + a$ может привести к обратным результатам, даже если мы знаем, что $p \equiv 1 \pmod{2K}$, так как затраты на итерацию могут перевесить выгоду от получения более короткого цикла.) Однако если мы *ничего* не знаем о числе $K$, то возникают по-настоящему серьезные вопросы относительно сложности.

Поэтому интересной нерешенной проблемой является следующая: $M$ компьютеров применяют ро-метод Полларда к некоторому числу при отсутствии каких-либо дополнительных сведений о числе $K$. Как оптимально задать соответствующие значения $\{K_m : m \in [1, \ldots, M]\}$ этим компьютерам? Возможно, ответом будет просто $K_m = 1$ для $m$-го компьютера, или значения $K_m$ будут равны различным малым простым числам. Также неясно, как будут меняться значения $K$ (и будут ли вообще), если мы от модели «независимых компьютеров» будем переходить к модели «параллельного вычисления», которая обсуждается в упр. 5.25. Один интуитивный взгляд на проблему таков: делитель Макинтоша—Тардифа числа $F_{18}$, а именно

$$81274690703860512587777 = 1 + 2^{23} \cdot 29 \cdot 293 \cdot 1259 \cdot 905678539$$

(который был найден при помощи метода ECM), можно найти ро-методом Полларда, особенно если какой-нибудь «удачливый» компьютер будет производить итерации по формуле

$$x = x^{2^{23} \cdot 29} \bmod F_{18}.$$

При анализе сложности алгоритма имейте в виду, что количество операций для совершения *одной итерации* растет как $O(\ln K_m)$, эта величина есть количество операций в схеме возведения в степень.

**5.25.** Проанализируйте одну идею о распараллеливании ро-метода Полларда для разложения на множители (это — не метод распараллеливания для дискретных логарифмов, который обсуждался в тексте). Пусть $j$-й из $M$ компьютеров вычисляет последовательность Полларда для итераций $x = x^2 + a \bmod N$ с *общим* параметром $a$, но с различным для каждого компьютера начальным значением $x_1^{(j)}$, т. е.

$$\left\{ x_i^{(j)} : \ i = 1, 2, \ldots, n \right\};$$

так мы получим полную последовательность длины $n$ для каждого $j \in [1, M]$. Покажите, что если мы сможем вычислить произведение

$$Q = \prod_{i=1}^{n} \prod_{j=1}^{M} \prod_{k=1}^{M} \left( x_{2i}^{(j)} - x_i^{(k)} \right)$$

по модулю числа $N$, подлежащего разложению, то это полное произведение

будет иметь порядка $n^2M^2$ алгебраических сомножителей, и значит, в свою очередь, нам потребуется около $p^{1/2}/M$ *параллельных* итераций для нахождения неизвестного делителя $p$. Так что вопрос сводится к следующему: можем ли мы распараллелить указанное произведение, используя какую-нибудь быструю схему вычисления значений многочлена? Ответ на этот вопрос — положительный, и является предметом некоторых эвристических обсуждений, подробно описанных в [Crandall 1999d], в которых утверждается, что при помощи $M$ компьютеров можно найти неизвестный делитель $p$ за

$$O\left(\sqrt{p}\,\frac{\ln^2 M}{M}\right)$$

параллельных операций.

**5.26.** Вспомним, что особенностью подхода к решению задачи дискретного логарифмирования (DL) на основе ро-метода Полларда являются минимальные требования к памяти. Более того, существует большое разнообразие вариантов основного ро-подхода. Настоящее упражнение состоит в исследовании одного очень простого варианта, направленного на решение конкретной задачи DL

$$g^l \equiv t \pmod p,$$

где число $t$ и первообразный корень $g$ заданы. Сначала определим псевдослучайную функцию на вычетах $z \bmod p$, например,

$$f(z) = 2 + 3\theta(z - p/2),$$

т. е. $f(z) = 2$ при $z < p/2$, и $f(z) = 5$ в противном случае. Теперь определим последовательность $x_1 = t, x_2, x_3, \ldots$, такую что

$$x_{n+1} = g^{f(x_n)}x_n t$$

для $n \geqslant 1$. Замечательным фактом является то, что мы можем использовать две последовательности $(w_n = x_{2n})$, $(x_n)$, так же, как в алгоритме 5.2.1, где одна последовательность обгоняет другую с двойным ускорением. Затем мы выполним эти итерации, надеясь на равенство

$$x_{2n} \equiv x_n \pmod p,$$

поскольку это равенство дает соотношение

$$t^a \equiv g^b \pmod p.$$

И затем, используя результат упражнения 5.4, мы найдем нужный дискретный логарифм. Вычислите на компьютере этим методом, используя данную нами псевдослучайную функцию $f$, дискретные логарифмы для

$$\begin{aligned} 11^{4950\,11427} &\equiv 3 \pmod{2^{31}-1}, \\ 17^{1629} &\equiv 3 \pmod{2^{17}-1}. \end{aligned}$$

Интересный вопрос для исследования таков: насколько разнообразны варианты ро-метода Полларда? Мы только что убедились, что существует несколько способов получения последовательностей Полларда в виде комбинации сте-

пеней чисел $x$ и $g$, но можно также рассматривать и дробные степени. Например, если в ро-методе Полларда мы получим цепочку корней вида

$$\sqrt{g^{e_n}\ldots\sqrt{g^{e_2}\sqrt{g^{e_1}t}}} \equiv \sqrt{g^{e_{2n}}\ldots\sqrt{g^{e_2}\sqrt{g^{e_1}t}}} \pmod{p},$$

где степени $e_n$ — случайные числа (но такие, что каждое звено цепочки допускает извлечение квадратного корня), то каждую сторону сравнения можно достаточное количество раз возвести в квадрат и получить смешанное соотношение для $g, t$, как и раньше. Хотя извлечение квадратного корня не является «дешевой» операцией, этот подход может быть интересен, если мы каким-либо образом сможем порождать в среднем короткие циклы для цепочек корней.

**5.27.** В связи с $(p-1)$-методом Полларда покажите, что если число $n$ составное и не является степенью, и если мы знаем целое число $m < n^2$, такое что $p-1|m$ для некоторого простого $p|n$, то мы можем использовать это число $m$ в вероятностном алгоритме для нахождения нетривиального разложения на множители числа $n$. Покажите, что этот алгоритм закончит свою работу за полиномиальное время (количество арифметических операций с целыми числами размера $n$ ограничено степенью $\ln n$).

**5.28.** Здесь мы исследуем «круговую группу», определенную для нечетного простого числа $p$ как множество

$$C_p = \{(x,y) : x,y \in [0,p-1]; x^2+y^2 \equiv 1 \pmod{p}\}$$

с операцией «$\oplus$», определенной как

$$(x,y)\oplus(x',y') = (xx'-yy', xy'+yx') \bmod p.$$

Покажите, что порядок круговой группы равен

$$\#C_p = p - \left(\frac{-1}{p}\right).$$

Докажите следствие, что порядок всегда делится на 4. Объясните, как операция $\oplus$ связана с комплексным умножением (для гауссовых целых чисел), и найдите алгебраические связи между круговой группой и полем $\mathbf{F}_{p^2}$.

Опишем теперь алгоритм разложения на множители (который можно назвать «$(p\pm1)$-методом»), основанный на круговой группе. Мы начинаем с точки $P_0 = (x_0, y_0)$ и вычисляем ее кратные $[n]P_0$ по существу так же, как в алгоритме ECM. Но как же найти начальную точку? (В связи с этим вопросом см. упр. 5.16.) Насколько эффективным является этот метод по сравнению со стандартным $(p-1)$-методом? Оценивая эффективность, учтите, что точка может быть удвоена всего за два умножения в поле. Сколько надо произвести умножений, чтобы сложить две произвольные точки?

Проанализируйте, имеет ли смысл метод разложения на множители, основанный на «гиперсферической» группе. Такой группой будет множество

$$H_p = \left\{(x,y,z,w) \;:\; x,y,z,w \in [0,p-1]; x^2+y^2+w^2+z^2 \equiv 1 \pmod{p}\right\}$$

с операцией умножения гиперкомплексных кватернионов. Покажите, что порядок группы равен

$$\#H_p = p^3 - p.$$

При оценке эффективности такого метода разложения на множители вы должны рассмотреть по меньшей мере следующие вопросы. Как найти в этом случае начальную точку $(x_0, y_0, w_0, z_0)$ в нашей группе? Сколько операций в поле требуется для удвоения точки, для сложения двух произвольных точек?

Рассмотрите алгебраические связи круговой и гиперсферической группы (и возможно других родственных им групп) с группами матриц $(\mathrm{mod}\ p)$. Например, все матрицы размера $n \times n$ с определителем 1 по модулю $p$ образуют группу, которая с тем или иным успехом может быть использована для создания некоторого алгоритма разложения на множители. Эти связи хорошо известны, включая также связь с так называемым циклотомическим разложением на множители. Интересное направление исследований основано на следующем вопросе: как разработать *эффективные* алгоритмы разложения на множители, если таковые вообще есть, основанные на этих идеях о группах/матрицах? Мы уже знаем, что комплексное умножение, например, можно произвести за три умножения вместо четырех. Умножение больших матриц может быть выполнено при помощи своих собственных специальных ускорений, таких как рекурсия Штрассена (см. [Crandall 1994b]) и теоретико-числовые преобразования (см. [Yagle 1995]); см. также упр. 9.84.

**5.29.** Рассмотрите возможность модификации схемы Полларда—Штрассена для вычисления значений многочлена в приложении к разложению на множители чисел Ферма $F_n = 2^{2^n} + 1$. Так как мы можем ограничиться поиском простых делителей среди чисел вида $p = k2^{n+2} + 1$, рассмотрите следующий подход. Построим произведение

$$P = \prod_i \left(k_i 2^{n+2} + 1\right)$$

(по модулю $F_n$), где $\{k_i\}$ составляют некоторое множество подходящим образом выбранных целых чисел, чтобы в итоге вычислить НОД $(F_n, P)$. Концепция Полларда—Штрассена о вычислении произведения последовательных целых чисел меняется: сейчас мы хотим сформировать произведение по специальному семейству множителей. Рассмотрите возможные способы эффективного получения произведения $P$. Существует интересная мысль, что нам следует как-то предварительно просеять числа $\{k_i\}$ или даже изменять степени $n + 2$ в зависимости от $i$. Имеет ли смысл описать семейство множителей $\{k_i\}$ как объединение непересекающихся арифметических прогрессий (которые получатся в результате предварительного просеивания)? Одним из важных практических вопросов является следующий: есть ли надежда, что такая вариация метода Полларда—Штрассена превзойдет по производительности обычное прямое решето (в котором просто проверяется $2^{2^n} \pmod p$ для различных $p = k2^{n+2} + 1$)? Эта проблема заслуживает внимания, так как начиная с $F_{20}$ или около того, прямое решето является единственным на данный момент способом нахождения делителей больших чисел $F_n$.

Глава 6

# СУБЭКСПОНЕНЦИАЛЬНЫЕ АЛГОРИТМЫ РАЗЛОЖЕНИЯ НА МНОЖИТЕЛИ

Методы настоящей главы включают в себя две из трех главных «рабочих лошадок» современного разложения на множители — квадратичное решето (quadratic sieve — QS) и решето числового поля (number field sieve — NFS). (Третья «лошадка» — метод эллиптических кривых (elliptic curve method — ECM) — описывается в гл. 7.) Квадратичное решето и решето числового поля являются прямыми наследниками метода разложения на множители с помощью непрерывных дробей Бриллхарта и Моррисона, который был самым первым субэкспоненциальным алгоритмом разложения. Метод разложения с помощью непрерывных дробей, предложенный в начале 70-х годов XX в., позволял полностью раскладывать числа, имеющие до 50 знаков, в то время как до этого пределом было 20 знаков. Квадратичное решето и решето числового поля, каждое из которых имеет свои плюсы и господствует в своей области, к настоящему времени позволили увеличить размеры чисел, допускающих методичное разложение, с 50-и до более чем 150-и знаков. С другой стороны, метод эллиптических кривых позволяет обнаруживать простые делители с более чем 50-ю знаками, и, оказывается, слабо зависит от размера разлагаемых чисел. Мы включили в эту главу небольшое обсуждение строгих методов разложения на множители, которые также по-своему отражают современное состояние теории. Мы также кратко обсудим некоторые субэкспоненциальные алгоритмы дискретного логарифмирования для мультипликативных групп конечных полей.

## 6.1. Разложение с помощью метода квадратичного решета

Впервые предложенный в работе [Pomerance 1982] метод квадратичного решета (QS), однако, многим обязан своим предшественникам, включая метод непрерывных дробей Бриллхарта и Моррисона (см. [Morrison and Brillhart 1975]). Историю метода QS, а также решета числового поля см. в работе [Pomerance 1996].

### 6.1.1. Базовый метод QS

Пусть $n$ — нечетное число, у которого в точности $k$ различных простых делителей. Тогда существует в точности $2^k$ квадратных корней из 1 по модулю $n$.

Это легко показать в случае $k = 1$, а в общем случае это следует из китайской теоремы об остатках (см. разд. 2.1.3). Два из этих $2^k$ квадратных корней из 1 — наши старые знакомые, $\pm 1$. Все остальные интересны тем, что их можно использовать для разложения $n$. В самом деле, если $a^2 \equiv 1 \pmod{n}$ и $a \not\equiv \pm 1 \pmod{n}$, то НОД$(a-1, n)$ должен быть нетривиальным делителем $n$. Чтобы это увидеть, заметим, что $n|(a-1)(a+1)$, но $n$ не делит ни один из множителей, так что одна часть числа $n$ должна делить $a-1$, а другая часть — $a+1$.

Возьмем, к примеру, случай $a = 11$ и $n = 15$. Имеем $a^2 \equiv 1 \pmod{n}$, и НОД$(a-1, n) = 5$, нетривиальному делителю числа 15.

Рассмотрим следующие три несложные задачи: нахождение делителя четного числа, разложение нетривиальных степеней целых чисел, вычисление НОД. По поводу первой задачи комментарии излишни. Вторая решается взятием $\lfloor n^{1/k} \rfloor$ и проверкой, что $k$-я степень этого числа есть $n$; корень извлекается методом Ньютона, $k$ меняется до $\lg n$. Третья несложная задача решается посредством алгоритма 2.1.2. Итак, мы можем «свести» задачу о разложении к нахождению нетривиальных квадратных корней из 1 для нечетных составных чисел, не являющихся степенями. Мы пишем «свести» в кавычках, поскольку это не совсем сведение — с вычислительной точки зрения эти две задачи, по существу, являются эквивалентными. В самом деле, если мы можем разложить $n$, нечетное составное, не являющееся степенью, то нетрудно поработать с этим разложением и с НОД и получить разложение $n = AB$, где $A, B$ больше 1 и взаимно просты (см. упражнения). Далее, пусть $a$ есть решение задачи из условия китайской теоремы об остатках, поставленной следующим образом:

$$a \equiv 1 \pmod{A}, \quad a \equiv -1 \pmod{B}.$$

Мы тем самым построили нетривиальный квадратный корень из 1 по модулю $n$.

Сейчас мы нацелены на задачу нахождения нетривиального квадратного корня из 1 по модулю $n$, где $n$ — нечетное составное число, не являющееся степенью. Эта задача, в свою очередь, эквивалентна нахождению решения сравнения $x^2 \equiv y^2 \pmod{n}$, где $xy$ — взаимно простые с $n$ и $x \not\equiv \pm y \pmod{n}$. Отсюда $xy^{-1} \pmod{n}$ есть нетривиальный квадратный корень из 1. С другой стороны, как мы видели, любое решение сравнения $x^2 \equiv y^2 \pmod{n}$, где $x \not\equiv \pm y \pmod{n}$, может быть использовано для разложения $n$.

Основная идея алгоритма QS состоит в нахождении сравнений вида $x_i^2 \equiv a_i \pmod{n}$, причем $\prod a_i$ есть квадрат, скажем, $y^2$. Если $x = \prod x_i$, то $x^2 \equiv y^2 \pmod{n}$. Дополнительное требование $x \not\equiv \pm y \pmod{n}$ первоначально игнорируется. Если это условие оказалось выполненным, то нам повезло, и мы можем разложить $n$. Если оно не выполнено, мы пробуем снова применить этот метод. Мы увидим, что на самом деле можно получить целую серию пар сравнимых между собой квадратов, и если предположить определенного рода статистическую независимость, более половины из них должны привести к разложению числа $n$. Однако следует с самого начала отметить, что QS, строго говоря, не является вероятностным алгоритмом. Говоря о статистической независимости, мы говорим о ней, как об эвристическом соображении. Числа, которые мы пы-

таемся раскладывать, кажется, не слишком «возражают» против отсутствия у нас строгого обоснования и раскладываются, не взирая ни на что. Давайте испытаем этот метод для числа $n = 1649$, которое является составным и не является степенью. Начиная так же как и в методе Ферма, мы берем в качестве $x_i$ числа, идущие сразу после $\sqrt{n}$ (см. разд. 5.1.1):

$$\begin{aligned} 41^2 &= 1681 \equiv 32 \pmod{1649}, \\ 42^2 &= 1764 \equiv 115 \pmod{1649}, \\ 43^2 &= 1849 \equiv 200 \pmod{1649}. \end{aligned}$$

В методе Ферма мы продолжали бы эти вычисления, пока не достигли бы $57^2$, но наша новая идея объединения сравнений позволяет ограничиться этими тремя вычислениями. В самом деле, $32 \cdot 200 = 6400 = 80^2$, так что мы имеем

$$(41 \cdot 43)^2 \equiv 80^2 \pmod{1649}.$$

Заметим, что $41 \cdot 43 = 1763 \equiv 114 \pmod{1649}$ и $114 \not\equiv \pm 80 \pmod{1649}$, следовательно, мы на правильном пути. В самом деле, $\text{НОД}(114 - 80, 1649) = 17$, и мы находим $1649 = 17 \cdot 97$.

Можно ли распространить эту идею на бóльшие числа? Пусть мы рассматриваем вычеты $x^2 \bmod n$ для $x$, пробегающих целые числа, начиная с $\lceil \sqrt{n} \rceil$. Нам бы хотелось найти среди них непустое подмножество, произведение элементов которого является квадратом. Возникает естественный вопрос — как это сделать?

Для начала приведем некоторые соображения на этот счет. Во-первых, заметим, что, если некоторое $x^2 \bmod n$ имеет большой простой делитель, входящий в его разложение в первой степени, то если мы включим данный вычет в наше подмножество с произведением, равным квадрату, то там должно быть и другое $x'^2 \bmod n$, имеющее тот же самый большой простой делитель. В рассмотренном выше упрощенном примере с числом 1649 вторым вычетом является число 115, которое имеет относительно большой простой делитель 23 (большой по сравнению с простыми делителями двух других вычетов), и действительно, мы исключили соответствующее сравнение и не использовали его в нашем произведении. А что, если мы будем поступать так систематически, выбрасывая все $x^2 \bmod n$, имеющие простые делители, превосходящие, скажем, $B$? Т. е. допустим, что мы сохраняем только $B$-гладкие числа (см. опр. 1.4.8). Мы приходим к следующему вопросу:

> Сколько положительных $B$-гладких чисел необходимо для того, чтобы произведение их непустого подмножества заведомо было квадратом?

Недолгое размышление приводит нас к осознанию того, что данный вопрос, на самом деле, из области линейной алгебры! Сопоставим «вектор показателей» $B$-гладкому числу $m = \prod p_i^{e_i}$, где $p_1, p_2, \ldots, p_{\pi(B)}$ — простые числа, не превосходящие $B$, и каждый показатель $e_i \geqslant 0$. Вектор показателей есть

$$\vec{v}(m) = (e_1, e_2, \ldots, e_{\pi(B)}).$$

Если все числа $m_1, m_2, \ldots, m_k$ являются $B$-гладкими, то $\prod_{i=1}^{k} m_i$ есть квадрат тогда и только тогда, когда у вектора $\sum_{i=1}^{k} \vec{v}(m_i)$ все координаты — четные.

Последнее соображение подсказывает, что мы должны проводить редукцию векторов показателей по модулю 2 и рассматривать их как элементы векторного пространства $\mathbf{F}_2^{\pi(B)}$. Полем скаляров данного векторного пространства является $\mathbf{F}_2$, содержащее всего два элемента — 0, 1. Так что линейная комбинация различных векторов данного векторного пространства есть не что иное, как сумма по подмножеству. Это подмножество соответствует тем векторам в линейной комбинации, которые имеют коэффициент 1. Тем самым поиск непустого подмножества целых чисел с произведением, равным квадрату, сводится к поиску линейной зависимости в множестве векторов.

У данной точки зрения есть два больших преимущества. Во-первых, мы сразу имеем теорему из линейной алгебры о том, что множество векторов линейно зависимо, если их число больше, чем размерность пространства. Так мы получаем ответ: для составления произведения, равного квадрату, потребуется самое большее $\pi(B) + 1$ положительных $B$-гладких чисел. Во-вторых, в предмет линейной алгебры также входят эффективные алгоритмы, например алгоритмы приведения матриц. Тем самым вопрос о нахождении линейной зависимости в множестве векторов сводится к построчному приведению матрицы, составленной из этих векторов.

Итак, мы, кажется, ответили на поставленный выше «естественный вопрос», позволяющий сделать рывок от числа 1649 к бóльшим числам. У нас есть метод систематической обработки наших вычетов $x^2 \bmod n$, теорема, говорящая о том, в каком случае мы имеем достаточное их количество, и алгоритм нахождения среди них подмножества с произведением, равным квадрату.

Мы, однако, не уточнили, как выбирается граница гладкости $B$, и из приведенных рассуждений на самом деле не следует, что данная схема будет сколько-нибудь быстрее метода Ферма.

Если мы выбираем $B$ малым, то преимущество состоит в том, что нам не нужно много $B$-гладких вычетов для нахождения произведения элементов, равного квадрату. Но если $B$ слишком мало, то свойство $B$-гладкости будет настолько редким, что нам может *вообще* не встретиться $B$-гладких чисел. Так что нам нужно уравновесить эти две силы, действующие на границу гладкости $B$: эта граница должна быть достаточно малой, чтобы нам для успеха не нужно было слишком много $B$-гладких чисел, но в то же время $B$ должно быть достаточно большим, чтобы $B$-гладкие числа появлялись с достаточной частотой.

Чтобы попытаться решить эту задачу, нам следует вычислить, чему будет равна частота $B$-гладких чисел как функция от $B$ и от $n$. Вероятно, мы можем попытаться использовать (1.44) и предположить, что «вероятность» $B$-гладкости $x^2 \bmod n$ есть примерно $u^{-u}$, где $u = \ln n / \ln B$.

В связи с этим подходом возникают два вопроса. Во-первых, (1.44) применимо лишь ко всей совокупности чисел до определенной границы, а не к специальному подмножеству. Можно ли быть уверенным, что члены наше-

го подмножества являются гладкими с той же вероятностью, что и типичное число? Во-вторых, чему равен точный порядок величины чисел в нашем подмножестве? В предыдущем абзаце мы просто использовали границу $n$, когда выражали число $u$.

Мы не будем учитывать первую из этих трудностей, поскольку мы строим эвристический метод разложения на множители. Если метод работает, то «гипотеза» о том, что наши специальные числа с точки зрения гладкости ведут себя как типичные числа, заслуживает определенного доверия. Вторую из этих трудностей, немного подумав, фактически можно разрешить в нашу пользу. А именно, мы допустили неточность в определении порядка вычетов $x^2 \bmod n$; на самом деле они *меньше*, чем $n$, значительно меньше.

Вспомним, что мы решили пробегать натуральные числа, начиная с $x = \lceil\sqrt{n}\rceil$. Но пока мы не дошли до $\lceil\sqrt{2n}\rceil$, вычет $x^2 \bmod n$ задается простой формулой $x^2 - n$. И если $\sqrt{n} < x < \sqrt{n} + n^\epsilon$, где $\varepsilon > 0$ мало, то $x^2 - n$ имеет порядок величины $n^{1/2+\epsilon}$. Так что нам следовало бы заменить нашу эвристическую оценку вероятности того, что $x$ ведет к $B$-гладкому числу на $u^{-u}$, где $u$ сейчас порядка $\frac{1}{2}\ln n/\ln B$.

Приведем еще одно обстоятельство, прежде чем использовать оценку $u^{-u}$ для выбора оптимального $B$ и для оценки необходимого числа значений $x$. А именно, сколько операций необходимо выполнить для отдельно взятого числа $x$, чтобы выяснить, является ли $x^2 - n$ $B$-гладким? Сначала можно подумать, что ответ порядка $\pi(B)$, поскольку пробные деления на простые числа до $B$ являются очевидным способом проверки $B$-гладкости числа. На самом деле для этого есть гораздо лучший способ, и этот способ существенно меняет дело. Мы можем использовать методы просеивания из разделов 3.2.5 и 3.2.6, так что среднее число арифметических операций, потраченное на одно значение $x$, будет лишь порядка $\ln\ln B$, что в самом деле очень мало. Данные методы требуют от нас просеивания по простым и степеням простых, где показатель степени по величине таков, что она может делить одно из значений $x^2 - n$. Простые числа $p$, которые являются основаниями этих степеней, — это те простые, для которых сравнение $x^2 - n \equiv 0 \pmod p$ разрешимо, а именно, простое $p = 2$ и нечетные простые $p \leqslant B$, для которых символ Лежандра $\left(\frac{n}{p}\right) = 1$. И для каждого такого нечетного простого числа $p$ и каждой подходящей степени $p$ имеются два класса вычетов для просеивания по ним значений $x$. Пусть $K$ есть число простых чисел до $B$, по которым мы производим просеивание. Тогда $K$ эвристически оценивается как $\frac{1}{2}\pi(B)$. Линейная зависимость между векторами показателей будет обеспечена, как только мы наберем $K+1$ вектор.

Если вероятность значения $x$, ведущего к $B$-гладкому числу есть $u^{-u}$, то ожидаемое число значений $x$ до наступления одного успеха есть $u^u$, а ожидаемое число значений до наступления $K+1$ успеха есть $u^u(K+1)$. Умножим эту величину на $\ln\ln B$ — среднее количество операций, потраченное на каждое значение $x$. Итак, допустим, что все наши расчеты верны и рассмотрим выражение

$$T(B) = u^u(K+1)\ln\ln B, \text{ где } u = \frac{\ln n}{2\ln B}.$$

Теперь пытаемся найти $B$ как функцию от $n$, с тем чтобы манипулировать $T(B)$. Поскольку $K \approx \frac{1}{2}\pi(B)$ есть величина порядка $B/\ln B$ (см. теорему 1.1.4), мы имеем, что $\ln T(B) \sim S(B)$, где $S(B) = u\ln u + \ln B$. Подставляя вместо $u$ его выражение, получаем, что производная задается формулой

$$\frac{dS}{dB} = \frac{-\ln n}{2B\ln^2 B}(\ln\ln n - \ln\ln B - \ln 2 + 1) + \frac{1}{B}.$$

Приравнивая это к нулю находим, что $\ln B$ лежит между константой, умноженной на $\sqrt{\ln n}$, и константой, умноженной на $\sqrt{\ln n \ln\ln n}$, так что $\ln\ln B \sim \frac{1}{2}\ln\ln n$. Так мы находим, что критическое значение $B$ и другие величины ведут себя следующим образом:

$$\ln B \sim \frac{1}{2}\sqrt{\ln n \ln\ln n}, \quad u \sim \sqrt{\ln n/\ln\ln n}, \quad S(B) \sim \sqrt{\ln n \ln\ln n}.$$

Мы заключаем, что оптимальным выбором границы гладкости $B$ будет величина порядка $\exp\left(\frac{1}{2}\sqrt{\ln n \ln\ln n}\right)$, и что время работы при данном выборе $B$ будет порядка $B^2$, другими словами, время работы указанной схемы разложения числа $n$ должно быть порядка $\exp\left(\sqrt{\ln n \ln\ln n}\right)$.

Мы будем обозначать последнюю функцию от $n$ следующим образом:

$$L(n) = e^{\sqrt{\ln n \ln\ln n}}. \tag{6.1}$$

Приведенные рассуждения не учитывают сложность этапа применения линейной алгебры, но можно показать, что она также порядка $B^2$ (см. разд. 6.1.3). В предположении всех сделанных эвристических допущений мы описали детерминированный алгоритм разложения нечетного составного числа $n$, не являющегося степенью. Время его работы $L(n)^{1+o(1)}$. Эта функция от $n$ — субэкспоненциальная, т. е. имеет вид $n^{o(1)}$, и будучи таковой растет медленнее, чем любая оценка сложности алгоритмов, описанных в гл. 5.

## 6.1.2. Базовый алгоритм QS. Резюме

Выше мы описали базовый алгоритм QS. Резюмируем сказанное.

**Алгоритм 6.1.1** (базовое квадратичное решето). Нам дано нечетное составное число $n$, не являющееся степенью. Данный алгоритм пытается выдать нетривиальное разложение числа $n$.

1. [Инициализация]
   $B = \lceil L(n)^{1/2} \rceil$; // Или выбрать $B$ «на свой вкус».
   Установить $p_1 = 2$ и $a_1 = 1$;
   Искать нечетные простые числа $p \leqslant B$, для которых $\left(\frac{n}{p}\right) = 1$, и присвоить им номера: $p_2, \ldots, p_K$;
   для $(2 \leqslant i \leqslant K)$ искать корни $\pm a_i$, такие, что $a_i^2 \equiv n \pmod{p_i}$;
   // Искать эти корни посредством алгоритма 2.3.8 или 2.3.9.
2. [Просеивание]

Просеять последовательность $(x^2-n)$, $x=\lceil\sqrt{n}\rceil, \lceil\sqrt{n}\rceil+1, \ldots$, на предмет $B$-гладких чисел, пока мы не добавим в множество $S$ $K+1$ подходящую пару $(x, x^2-n)$;
// См. разд. 3.2.5, 3.2.6 и замечания (2), (3), (4).

3. [Линейная алгебра]
for$((x, x^2-n) \in S)$ {
Установить разложение на простые сомножители $x^2-n=\prod_{i=1}^{K} p_i^{e_i}$;
$\vec{v}(x^2-n)=(e_1, e_2, \ldots, e_K)$; // Вектор показателей.
}
Составить $(K+1)\times K$-матрицу, строками которой являются всевозможные векторы $\vec{v}(x^2-n)$, взятые по $\bmod 2$;
Использовать алгоритмы линейной алгебры для нахождения нетривиального подмножества строк этой матрицы, дающих в сумме 0-вектор $(\bmod 2)$, скажем, $\vec{v}(x_1)+\vec{v}(x_2)+\ldots+\vec{v}(x_k)=\vec{0}$;

4. [Разложение на множители]
$x=x_1x_2\ldots x_k \bmod n$;
$y=\sqrt{(x_1^2-n)(x_2^2-n)\ldots(x_k^2-n)} \bmod n$;
// Извлечь этот корень, исходя непосредственно из известного разложения на простые сомножители полного квадрата $(x_1^2-n)(x_2^2-n)\ldots(x_k^2-n)$, см. замечание (6).
$d=\text{НОД}(x-y, n)$;
return $d$;

Есть несколько замечаний, которые необходимо сделать относительно этого алгоритма:

(1) На практике обычно используется несколько меньшее значение $B$, чем то, которое задается формулой на шаге [Инициализация]. Любое значение $B$, по порядку величины равное $L(n)^{1/2}$, даст ту же суммарную сложность, и существуют разные практические доводы, говорящие в пользу уменьшения этого значения; среди них размер матрицы, с которой приходится работать на шаге [Линейная алгебра], и размер модулей, с помощью которых производится просеивание, в сравнении с размером кэш-памяти компьютера, используемого на шаге [Просеивание]. Оптимальное значение $B$, скорее, из области искусства, чем науки, и, вероятно, его лучше всего оставить экспериментаторам.

(2) Для просеивания нужно знать, какие классы вычетов просеивать для каждого $p_i$, найденного на шаге [Инициализация]. (Для простоты мы не будем рассматривать задачу просеивания с более высокими степенями этих простых. Такое просеивание легко осуществить — например, можно использовать алгоритм 2.3.11, но повторяем, на практике его можно не рассматривать, поскольку оно не вносит весомого вклада в нахождение $B$-гладких чисел.) Для нечетных простых $p_i$ на шаге [Инициализация] мы решаем сравнение $x^2 \equiv n \pmod{p_i}$. Оно разрешимо, поскольку на этом шаге выбираются в точности те $p_i$, которые обладают указанным свойством. Для решения сравнения можно использовать как алгоритм

2.3.8, так и алгоритм 2.3.9. Разумеется, для каждого решения вторым решением будет найденный класс вычетов со знаком «минус», так что мы просеиваем два класса вычетов для каждого нечетного $p_i$. (Хоть мы и можем просеивать по числу $p_1 = 2$, как это указано в нашем псевдокоде, нам вовсе не обязательно просеивать и по числу 2, и по другим очень малым простым числам; см. соответствующие замечания в разд. 3.2.5.)

(3) Важно отметить, что арифметика реального просеивания может осуществляться путем сложений приближенных логарифмов просеивающих простых чисел, о чем говорилось в разд. 3.2.5. А именно, мы могли бы разместить в памяти и заполнить нулями массив некоторого подходящего размера, равного, скажем, $b$ байтам, которые соответствуют первым $b$ значениям $x$. Затем добавлять приращение $\lg p_i$ (округленное до ближайшего целого), начиная со смещений $x_i, x_i'$ — это наименьшие целые числа $\geqslant \lceil\sqrt{n}\rceil$, которые сравнимы по $(\mathrm{mod}\ p_i)$ с $a_i, -a_i$, соответственно, и далее через каждые $p_i$ элементов в нашем массиве. Если необходимо (т. е. если не найдено достаточное количество гладких чисел), заполнить нулями новый массив, с первым элементом, отвечающим $\lceil\sqrt{n}\rceil + b$, и т. д. Порог для выдачи предполагаемой ячейки с $B$-гладким значением устанавливается равным $\lfloor \lg|x^2 - n| \rfloor$ минус некоторый большой запас, например 20, чтобы компенсировать ошибки в приближенных логарифмах и другие ошибки, которые могут возникнуть из-за отсутствия просеивания по очень малым простым и высоким степеням. Любое выданное значение нужно проверить с помощью пробных делений и выяснить, является ли оно в самом деле $B$-гладким. Это разложение понадобится на шаге [Линейная алгебра]. (Получению правильно работающей реализации может помочь проверка ячеек логарифмического массива с помощью настоящих трудоемких разложений.)

(4) Вместо того чтобы начинать с $\lceil\sqrt{n}\rceil$ и пробегать дальше целые числа, рассмотрим такую возможность: $x$ пробегает последовательность целых чисел *с центром* $\sqrt{n}$. У этой идеи есть преимущество и недостаток. Преимущество в том, что значения многочлена $x^2 - n$ будут в среднем несколько меньше, так что, по-видимому, они с большей вероятностью будут $B$-гладкими. Недостаток в том, что некоторые значения будут отрицательными, а при формировании квадратов нужно учитывать знак. У квадратов не только все простые сомножители входят в разложение с четными показателями, квадраты еще и положительны. Однако недостаток легко исправить. Мы увеличим длину векторов показателей на одну координату, положив новую координату, скажем нулевую, равной 1, если наше целое число отрицательно, и 0, если оно положительно. Таким образом, как и в случае со всеми остальными координатами, мы хотим получить четное число единиц. Все это приводит к увеличению размерности нашего векторного пространства ($K$ меняется на $K + 1$). Значит, недостаток использования отрицательных чисел в том, что наши векторы станут на 1 бит длиннее, и нам потребуется на один вектор больше, чтобы обеспечить линейную зависимость. Это несущественный недоста-

ток; он компенсируется преимуществом просеивания меньших чисел. Поэтому, допуская отрицательные значения многочлена, можно двигаться дальше.

(5) Мы оставили в стороне следующую проблему: нет гарантии, что число $d$, сгенерированное на шаге [Разложение на множители], является нетривиальным делителем числа $n$. Если предположить определенного рода случайность (которая, конечно, не имеет места, но может считаться разумным эвристическим допущением), то «вероятность» того, что $d$ является нетривиальным делителем, равна $1/2$ или выше (см. упр. 6.2). Если найти несколько больше зависимостей между нашими векторами показателей и снова предположить статистическую независимость, то можно поднять наши шансы на успех. Например, пусть мы просеиваем на шаге [Просеивание] до тех пор, пока не будет найдено $K+11$ значений многочлена, являющихся $B$-гладкими. Если предположить, что размерность нашего пространства теперь равна $K+1$ (поскольку мы допускаем отрицательные значения многочлена, см. выше), то будет как минимум 10 независимых линейных зависимостей. Шансы, что ни одна из них не даст нетривиального разложения числа $n$, меньше, чем 1 к 1000. И если эта вероятность неудачи все еще слишком велика для Вас, Вы можете накопить еще больше $B$-гладких чисел для получения приемлемого результата.

(6) На шаге [Разложение на множители] мы должны извлечь квадратный корень из, вероятно, очень большого квадрата, а именно $Y^2 = (x_1^2 - n)(x_2^2 - n)\ldots(x_k^2 - n)$. Однако нас интересует лишь $y = Y \bmod n$. Мы можем использовать тот факт, что на самом деле нам известно разложение $Y^2$ на простые сомножители, так что нам известно и разложение на простые сомножители числа $Y$. Мы можем тем самым вычислить $y$, используя алгоритм 2.1.5 для нахождения вычета по модулю $n$ каждой степени простого, входящей в разложение $Y$, а затем перемножить эти вычеты, вновь выполняя редукцию по модулю $n$. Мы обнаружим, что в методе решета числового поля проблему извлечения квадратного корня нельзя решить с такой легкостью.

В нескольких последующих разделах мы обсудим некоторые из принципиальных улучшений базового алгоритма квадратичного решета.

### 6.1.3. Быстрые матричные методы

При $B = \exp\left(\frac{1}{2}\sqrt{\ln n \ln\ln n}\right)$ мы увидели, что время выполнения просеивающей стадии QS равно (эвристически) $B^{2+o(1)}$. После этой стадии имеется порядка $B$ векторов длины порядка $B$ с координатами в конечном поле $\mathbf{F}_2$ из двух элементов, и требуется найти среди них непустое подмножество векторов, дающих в сумме нулевой вектор. Чтобы достичь для алгоритма QS суммарной сложности вида $B^{2+o(1)}$, нам потребуется подпрограмма решения задачи линейной алгебры, которая может найти нужное непустое подмножество, не увеличивая временную границу.

Для начала заметим, что составление матрицы из наших векторов и использование метода Гаусса для нахождения подмножества векторов, дающих в сумме нулевой вектор, имеет оценку сложности вида $O\left(B^3\right)$ (предполагая, что матрица имеет размер $B \times B$). Несмотря на это, на практике метод Гаусса удобно использовать при разложении не очень больших чисел. Существует несколько причин, почему бóльшая по порядку оценка сложности не является проблемой на практике.

(1) Поскольку матричную арифметику мы проводим над полем $\mathbf{F}_2$, она допускает естественную реализацию на компьютере. Если $w$ является длиной машинного слова (чаще всего это 8 или 16 бит на старых машинах, и 32 или 64 или даже больше — на новых), то мы можем обрабатывать каждую строку блоками по $w$ координат в каждом, где каждый шаг есть просто логическая операция, требующая очень малого числа тактов процессора.

(2) Первоначальная матрица достаточно разрежена, так что в самом начале, перед тем как произойдет ее заполнение, надо проделывать очень небольшое число операций, что несколько уменьшает временную границу в худшем случае.

(3) Если разлагаемое нами число не слишком велико, мы можем нагрузить алгоритм на просеивающей стадии и разгрузить на матричной стадии. То есть мы можем выбрать границу $B$ несколько заниженной, тем самым заставляя просевающую стадию выполняться дольше, но облегчая себе задачу на матричной стадии. Трудности с памятью при больших значениях $B$ составляют другую практическую причину, по которой $B$ выбирается меньше, чем теоретически оптимальное значение.

Что касается пункта (2), были найдены способы «рационального» использования метода Гаусса, с тем чтобы сохранять разреженность как можно дольше (см. [Odlyzko 1985], [Pomerance and Smith 1992]). Эти методы иногда называют «структурированные методы Гаусса».

По мере того как раскладываемые нами числа растут, матричная стадия QS (и особенно решета числового поля; см. разд. 6.2) принимает угрожающие размеры. Неблагоприятная оценка сложности метода Гаусса сводит на нет наши суммарные оценки, которые получены в предположении, что матричная стадия не является «узким местом». Кроме того, из-за трудности работы с гигантскими матрицами могут потребоваться большие и дорогие компьютеры, которые не так-то просто заполучить на длительный срок.

Было предложено по меньшей мере три альтернативных метода для разреженных матриц, предназначенных для замены метода Гаусса, два из которых уже были хорошо изучены в численном анализе. Эти два метода — метод сопряженного градиента и метод Ланцоша — были адаптированы для матриц с элементами в конечном поле. Третья возможность — координатно-рекуррентный метод Видемана (см. [Wiedemann 1986]). Этот метод основан на алгоритме Берлекэмпа—Мэсси для нахождения наименьшего линейного рекуррентного соотношения в последовательности элементов конечного поля. Каж-

дый из этих методов может работать с разреженной кодировкой матрицы, т. е. с кодировкой, которая перечисляет лишь местоположения ненулевых элементов. Так, если матрица имеет $N$ ненулевых элементов, требуемый объем памяти есть $O(N \ln B)$. Поскольку наши матрицы разложений имеют не более $O(\ln n)$ ненулевых элементов в каждой строке, объем памяти, требуемый для матричной стадии алгоритма с использованием разреженной кодировки, составляет $O\left(B \ln^2 n\right)$.

Можно провести строгий анализ методов Видемана и Ланцоша. Время работы этих методов составляет $O(BN)$, где $N$ есть число ненулевых элементов матрицы. Таким образом, оценка времени работы матричной стадии алгоритмов разложения типа QS составляет $B^{2+o(1)}$, что равняется оценке времени работы просеивания.

Обсуждение метода сопряженного градиента и метода Ланцоша см. в работе [Odlyzko 1985]. Изучение метода Ланцоша с теоретических позиций см. в статье [Teitelbaum 1998]. Некоторые практические усовершенствования метода Ланцоша см. в работе [Montgomery 1995].

### 6.1.4. Вариации больших простых

Как говорилось выше и в разд. 3.2.5, просеивание обходится очень дешево. В отличие от пробного деления, на которое тратится время, пропорциональное числу пробных делителей, т. е. одна «порция» времени на одно пробное простое, при просеивании на одно простое тратится времени меньше, если его величина больше. Фактически, время, потраченное на каждую позицию в решете, для каждого простого модуля $p$ в среднем пропорционально $1/p$. Однако существуют скрытые издержки, связанные с увеличением списка простых $p$, по которым мы просеиваем. Одна состоит в том, что мы вряд ли сможем поместить весь просеиваемый массив в память компьютера, так что мы разбиваем его на части. Если простое число $p$ превосходит длину сегмента нашего решета, мы должны потратить порцию времени на каждый сегмент, чтобы выяснить, попадает в него это простое число или нет. Таким образом, как только простое число выходит за эту границу, «$1/p$-философия» решета остается позади, и мы тратим по существу одинаковое время на каждое из этих бóльших простых: просеивание становится похожим на пробное деление. Другая скрытая издержка, возможно, не такая уж и скрытая. Когда мы обращаемся к стадии нашего алгоритма, которая связана с линейной алгеброй, матрица будет существенно больше, если использовать больше простых. Допустим, мы используем $10^6$ простых; это число вполне могло бы возникнуть на просеивающей стадии. Тогда наша матрица, если ее закодировать как двоичную (0,1)-матрицу, будет иметь $10^{12}$ битов. Да, это действительно большой объект, к которому нужно применять линейную алгебру! На самом деле, некоторые из подпрограмм линейной алгебры, которые будут использоваться (см. разд. 6.1.3), работают с разреженной кодировкой матрицы, т. е. со списком мест, где встречаются единицы, поскольку большинство элементов — нули. Тем не менее, объем памяти для матрицы играет существенную роль и налагает ограничения на величину используемой границы гладкости.

Проведенный в разд. 6.1.1 анализ указывает третью причину, по которой не следует брать границу гладкости слишком большой. А именно, это увеличило бы количество чисел, которое необходимо выдать для нахождения линейной зависимости. Однако эта причина в некотором смысле несостоятельна. Если уже существует зависимость внутри подмножества наших данных, наличие большего объема данных не уничтожит ее, но, вероятно, ее будет несколько труднее отыскать. Так что нам не следует воспринимать завышение границы гладкости как серьезное препятствие, если мы сможем разобраться с двумя трудностями, упомянутыми в предыдущем абзаце.

В своем простейшем виде вариация больших простых предоставляет нам дешевый способ несколько увеличить нашу границу гладкости; она без затрат предоставляет нам большое количество чисел, которые являются почти $B$-гладкими, но которые не прошли тест по причине того, что они имеют один больший простой делитель. Это большее простое число можно взять лежащим в интервале $(B, B^2]$. Следует с самого начала отметить, что взять числа, которые $B$-гладки за исключением того, что имеют один простой делитель в интервале $(B, B^2]$, — это *не то же самое*, что взять $B^2$-гладкие числа. При $B$ порядка $L(n)^{1/2}$, как мы положили в разд. 6.1.1, типичное $B^2$-гладкое число в окрестности $n^{1/2+\epsilon}$ в действительности имеет много простых делителей в интервале $(B, B^2]$, а не только один.

Как бы то ни было, вариация больших простых действительно дает нам то, чего у нас раньше не было. Допуская выдачу решетом сигналов для чисел, которые близки к порогу $B$-гладкости, но все же не достигают его, мы можем обнаружить числа, которые имеют один несколько больший простой делитель. Фактически, если из разложения числа выбросить все его простые сомножители, не превосходящие $B$, и оставшееся число будет меньше $B^2$, но больше 1, то это оставшееся число должно быть простым. Как раз эта идея работает в вариации больших простых. Наше решето несовершенно, поскольку мы используем приближенные логарифмы и, возможно, не просеиваем по малым простым (см. разд. 3.2.5), но дополнительная нечеткость не играет большой роли в общей массе рассматриваемых чисел. Некоторые числа с большим простым делителем, которые можно было бы выдать, вероятно, будут пропущены, другие числа будут выданы, хотя их и не следует выдавать, но ни то, ни другое не является существенной проблемой.

Итак, если мы можем получить эти числа с большим простым делителем безо всяких затрат, как нам использовать их в нашем алгоритме на стадии линейной алгебры? В действительности нам не следует рассматривать эти числа с большим простым делителем как имеющие более длинные векторы показателей, так как это может привести к слишком большой матрице. Существует очень дешевый способ обработки выданных чисел с большим простым делителем. Просто отсортируйте их по значению этого большого простого делителя. Если какое-либо большое простое встречается в отсортированном списке лишь один раз, то мы, по-видимому, не сможем использовать соответствующее число для составления квадрата, так что оно отбрасывается. Пусть теперь у нас есть $k$ выданных чисел с одним и тем же простым: $x_i^2 - n = y_i P$ для $i = 1, 2, \ldots, k$.

Тогда

$$(x_1 x_i)^2 \equiv y_1 y_i P^2 \pmod{n}, \text{ для } i = 2, \ldots, k.$$

Так что при $k \geqslant 2$ мы можем использовать векторы показателей для $k-1$ числа $y_1 y_i$, поскольку вклад множителя $P^2$ в вектор показателей, координаты которого взяты по mod 2, равен 0. Другими словами, спаренные большие простые приводят к векторам показателей по простым, которые не превосходят $B$. Поскольку отсортировать список можно очень быстро, получение этих новых векторов показателей выглядит как подарок свыше.

Существует одна неприятность от использования этих новых векторов показателей, которая, однако, не слишком велика. Вектор показателей для $y_1 y_i$ (см. выше) обычно не такой разреженный, как вектор показателей для полностью гладкого числа. Следовательно, матричные стратегии, которые используют преимущества разреженности, слегка «хромают». Повторяем, эта неприятность не слишком велика, и каждая серьезная реализация метода QS использует вариацию больших простых.

Может возникнуть вопрос, как велика вероятность получения пары совпадающих больших простых чисел? Другими словами, когда мы отсортируем наш список, может ли случиться так, что в нем очень мало совпадений, и что почти все будет отброшено, поскольку простые в нем встречаются по одному разу? Парадокс дней рождения из теории вероятностей подсказывает, что совпадения будут нередки при достаточном количестве чисел с большими простыми делителями. Реальный опыт исследователей показывает, что значение больших простых делителей ничтожно мало в начале работы, поскольку число совпадений очень мало, но по мере накопления данных начинает сказываться действие парадокса дней рождения, и совпадения больших простых чисел начинают появляться в изобилии, становясь существенным источником строк окончательной матрицы.

На практике было замечено, и это подкрепляется теорией, что чем больше большое простое число, тем менее вероятно, что оно совпадет. Так что большинство практических исследователей обычно избегают большого промежутка для больших простых, оставляя, к примеру, лишь те из них, которые лежат в интервале $(B, 20B]$ или $(B, 100B]$.

За прошедшие годы многие говорили о том, что если одно большое простое число — это хорошо, то два, вероятно, — лучше. Эта мысль была развита в работе [Lenstra and Manasse 1994]; ее авторам, по-существу, действительно удалось обнаружить лучшую производительность в задачах разложения больших чисел при использовании двух больших простых. Историческое разложение на множители знаменитого числа RSA129, о котором говорилось в разд. 1.1.2, было получено с помощью такой двойной вариации больших простых.

В двойной вариации больших простых присутствуют различные усложнения, которые не имели места в описанной выше одинарной вариации больших простых. Если у целого числа из интервала $(1, B^2]$ все простые делители превосходят $B$, то оно должно быть простым. Это фундаментальное наблюдение, которое используется в одинарной вариации больших простых. А что если це-

лое число из интервала $(B^2, B^3]$ не имеет простых делителей $\leqslant B$? Тогда оно либо простое, либо является произведением двух простых, каждое из которых превосходит $B$. В сущности, двойная вариация больших простых допускает выдачу чисел, в которых неразложенная часть доходит до $B^3$. Если эта неразложенная часть $m$ превосходит $B^2$, то применяется быстрый тест на псевдопростоту, скажем, проверка сравнения $2^{m-1} \equiv 1 \pmod{m}$; см. разд. 3.4.1. Если $m$ удовлетворяет этому сравнению, то оно *отбрасывается*, поскольку тогда оно, вероятно, простое, и, кроме того, оно слишком велико для того, чтобы совпасть с другим большим простым. Если с помощью сравнения доказано, что $m$ — составное, то его раскладывают на множители, скажем, ро-методом Полларда (см. разд. 5.2.1). Это приведет к выдаче чисел, которые $B$-гладки за исключением двух простых делителей, которые больше $B$ (но не намного больше).

Очевидно, это уже требует намного больше работы, чем одинарная вариация больших простых. Но это еще не все. Нужно исследовать выданные числа с одним или двумя большими простыми на предмет циклов, т. е. подмножеств, произведение по которым $B$-гладко за исключением больших простых, каждое из которых входит в произведение в четной степени. Например, пусть нам выданы числа $y_1P_1, y_2P_2, y_3P_1P_2$, где $y_1, y_2, y_3$ $B$-гладки и $P_1, P_2$ — простые, превосходящие $B$ (так что здесь мы описываем цикл, состоящий из двух чисел с одним большим простым и одного числа с двумя большими простыми). Произведение этих трех чисел равно $y_1y_2y_3P_1^2P_2^2$, и его вектор показателей по модулю 2 точно такой же как и у $B$-гладкого числа $y_1y_2y_3$. Конечно, могут быть и более сложные циклы, чем этот, часть из которых может содержать лишь разложения с двумя большими простыми (хотя такой тип циклов встречается нечасто). Уже не так просто, как раньше, исследовать наше множество данных на предмет таких циклов. А именно, множество данных будет значительно больше, чем раньше, и существует возможность переполнения данными. Эти проблемы обсуждаются в работе [Lenstra and Manasse 1994]. Ее авторы обнаружили, что на больших числах ими получено более чем двукратное ускорение при использовании двойной вариации больших простых. Однако они также допускают, что ими было использовано значение $B$, которое, вероятно, меньше, чем выбрали бы другие. Было бы интересно увидеть эксперимент, допускающий вариацию всех необходимых параметров, для выяснения той комбинации, которая является наилучшей для чисел различных размеров.

А что же насчет трех больших простых? Можно представить, что дополнительные трудности будут еще больше, чем в случае двух больших простых. Возможно, это того стоит, но, вероятно, использование вместо этого большего значения $B$ будет более выгодно.

### 6.1.5. Многие полиномы

В базовом методе QS $x$ у нас пробегало целые числа в окрестности $\sqrt{n}$, и мы искали среди значений $x^2 - n$ те, которые являются $B$-гладкими. Причина, по которой мы брали $x$ в окрестности $\sqrt{n}$, состоит в минимизации величины

$x^2 - n$, так как меньшие числа с большей вероятностью являются гладкими, чем бóльшие. Но для $x$, близких к $\sqrt{n}$, мы имеем $x^2 - n \approx 2(x - \sqrt{n})\sqrt{n}$, и как только $x$ удаляется от $\sqrt{n}$, то же самое делают и числа $x^2 - n$, причем делают это равномерно и быстро. Таким образом, в базовый метод QS заложено определенное уменьшение отдачи по мере работы алгоритма при, вероятно, приличной скорости выхода гладких чисел на начальном отрезке просеивания и заметном уменьшении этой скорости при дальнейшем просеивании.

Вариация метода QS с многими полиномами позволяет обойти эту проблему, благодаря использованию семейства полиномов, а не только полинома $x^2 - n$. Различные версии использования многих полиномов были предложены независимо Дэвисом, Холдриджем и Монтгомери (см. [Pomerance 1985]). Метод Монтгомери чуть лучше, и в настоящее время алгоритм QS используют именно так. В сущности, Монтгомери заменил переменную $x$ на специальным образом подобранную линейную функцию от $x$.

Пусть $a, b, c$ — целые числа, такие что $b^2 - ac = n$. Рассмотрим квадратичный полином $f(x) = ax^2 + 2bx + c$. Тогда

$$af(x) = a^2x^2 + 2abx + ac = (ax + b)^2 - n, \tag{6.2}$$

так что

$$(ax + b)^2 \equiv af(x) \pmod{n}.$$

Если мы имеем значение $a$, которое является квадратом, умноженным на $B$-гладкое число, и значение $x$, для которого $f(x)$ $B$-гладко, то вектор показателей для $af(x)$, взятый по модулю 2, дает нам строку нашей матрицы. Более того, возможные нечетные простые $p$, которые могут делить $f(x)$ (но не делят $n$) — это те, для которых $\left(\frac{n}{p}\right) = 1$, т. е. те же самые простые, которые используются в базовом алгоритме QS. (Важно иметь множество встречающихся простых, не зависящее от используемого многочлена, поскольку иначе в нашей матрице будет больше столбцов, и потому потребуется больше строк для генерации зависимости.)

Мы требуем, чтобы тройка $a, b, c$ удовлетворяла равенству $b^2 - ac = n$, и чтобы $a$ было $B$-гладким числом, умноженным на квадрат. Однако причина, по которой мы используем многочлен $f(x)$, состоит в том, что его значения могут быть малы и потому могут быть гладкими с большей вероятностью. Какие условия нам следует наложить на $a, b, c$, чтобы иметь малые значения $f(x) = ax^2 + 2bx + c$? Это зависит от длины интервала просеивания для данного полинома. Давайте заранее решим, что мы будем просеивать полином лишь для аргументов $x$, пробегающих интервал длины $2M$. Кроме того, в силу (6.2) можно договориться брать коэффициент $b$ таким, что он удовлетворяет неравенству $|b| \leqslant \frac{1}{2}a$ (в предположении, что $a$ положительно). Т. е. мы обеспечиваем то, что нашим интервалом длины $2M$ для $x$ будет в точности интервал $[-M, M]$. Заметим, что наибольшее значение $f(x)$ принимает на концах этого интервала, и оно приблизительно равно $(a^2M^2 - n)/a$, а наименьшим значением будет значение в точке $x = 0$, равное приблизительно $-n/a$. Сделаем модули этих двух выражений примерно равными между собой, что даст приближенное уравнение $a^2M^2 \approx 2n$, так что $a \approx \sqrt{2n}/M$.

Если $a$ удовлетворяет этому приближенному равенству, то модуль $f(x)$ на интервале $[-M, M]$ ограничен величиной $(M/\sqrt{2})\sqrt{n}$. Это следовало бы сравнить с тем, что дает первоначальный полином $x^2 - n$, используемый в базовом методе QS. На интервале $[\sqrt{n} - M, \sqrt{n} + M]$ его значения ограничены приближенно величиной $2M\sqrt{n}$. Так что в этой величине мы сэкономили множитель $2\sqrt{2}$. Но на самом деле мы сэкономили гораздо больше. В базовом методе QS значения продолжают расти — мы не можем остановиться на предварительно заданном значении $M$. Но когда мы используем семейство полиномов, мы можем их постоянно менять. Грубый анализ с использованием разд. 6.1.1 показывает, что мы можем взять $M = B = L(n)^{1/2}$ при использовании многих полиномов, тогда как должны взять $M = B^2 = L(n)$ при использовании лишь одного полинома. Так что числа, которые «претендуют на гладкость», при использовании многих полиномов в среднем меньше в $B$ раз. Эвристический анализ показывает, что использование многих полиномов ускоряет метод квадратичного решета примерно в $\frac{1}{2}\sqrt{\ln n \ln\ln n}$ раз. Когда $n$ имеет порядка 100 цифр, это дает примерно 17-кратную экономию, т. е. QS с многими полиномами работает в 17 раз быстрее базового метода QS. (Этот «мысленный эксперимент» не был подтвержден реальными численными расчетами, хотя не может быть никаких сомнений в том, что использование многих полиномов на практике дает значительное ускорение.)

Однако существует одно последнее требование относительно старшего коэффициента $a$: нам нужно найти значения $b, c$, которые с ним связаны. Если мы можем решить сравнение $b^2 \equiv n \pmod{a}$ относительно $b$, то мы можем обеспечить неравенство $|b| \leqslant a/2$ и можем получить $c = (b^2 - n)/a$. Заметим, что методы из разд. 2.3.2 позволяют нам решить это сравнение при условии, что мы выбираем $a$ таким, что: $a$ нечетное; нам известно разложение $a$ на простые сомножители; для каждого простого $p|a$ мы имеем $\left(\frac{n}{p}\right) = 1$. Один из эффективных способов осуществить это — брать различные простые $p \approx (2n)^{1/4}/M^{1/2}$, такие, что $\left(\frac{n}{p}\right) = 1$, и выбирать $a = p^2$. Тогда такие значения $a$ удовлетворяют всем требованиям, которые мы к ним предъявляем:

(1) Мы получаем $a$, равное квадрату, умноженному на $B$-гладкое число.
(2) Мы получаем $a \approx \sqrt{2n}/M$.
(3) Мы можем эффективно решить сравнение $b^2 \equiv n \pmod{a}$ относительно $b$.

Сравнение $b^2 \equiv n \pmod{a}$ имеет два решения, если, как выше, мы выберем $a = p^2$. Однако эти два решения приводят к эквивалентным полиномам, так что мы используем лишь одно из решений, например то, для которого $0 < b < \frac{1}{2}a$.

### 6.1.6. Автоматическая инициализация

В разд. 6.1.5 мы уяснили, что часто менять полиномы — это хорошо. Вопрос в том, насколько часто? Одно из ограничений, о котором уже неявно упоминалось, состоит в том, что длина $2M$ интервала, на котором мы просеиваем полином, должна быть не меньше $B$, границы простых модулей, по которым

мы просеиваем. Если есть только это ограничение, то разумным выбором было бы такое $M$, что $2M = B$.

Для чисел в диапазоне от 50 до 150 цифр типичные варианты значений $B$ лежат приблизительно в диапазоне от $10^4$ до $10^7$. Оказывается, просеивание является настолько быстрой операцией, что если бы мы меняли полином всякий раз, как просеяли $B$ чисел, временные затраты на это изменение могли бы оказаться настолько заметны, что общая эффективность пострадала бы. Эти затраты состоят, главным образом, в решении *проблемы инициализации*. Т. е. нам даны $a, b, c$ (см. разд. 6.1.5), и для каждого нечетного простого числа $p \leqslant B$, такого, что $\left(\frac{n}{p}\right) = 1$, мы должны решить сравнение

$$ax^2 + 2bx + c \equiv 0 \pmod{p}$$

и найти два корня $r(p) \bmod p$ и $s(p) \bmod p$ (здесь мы предполагаем, что $p$ не делит $an$). Так, мы имеем

$$r(p) = (-b + t(p))a^{-1} \bmod p,\ \ s(p) = (-b - t(p))a^{-1} \bmod p, \tag{6.3}$$

где

$$t(p)^2 \equiv n \pmod{p}.$$

Для каждого полинома мы можем использовать один и тот же вычет $t(p)$ всякий раз, как нам приходится находить $r(p), s(p)$. Так что основная часть работы при использовании (6.3) состоит в вычислении $a^{-1} \bmod p$ для каждого $p$ (например, посредством алгоритма 2.1.4) и выполнении двух умножений по модулю $p$. Если имеется много простых $p$, для которых это необходимо сделать, то это уже немалая работа, которую нам не хотелось бы выполнять слишком часто.

Идея автоматической инициализации состоит в том, чтобы выполнить часть работы по вычислению корней (6.3) сразу для нескольких многочленов с одним и тем же значением $a$. Для каждого значения $a$ мы выбираем $b$ так, что $b^2 \equiv n \pmod{a}$ и $0 < b < a/2$ (см. разд. 6.1.5). Для каждого такого $b$ мы можем выписать полином $ax^2 + 2bx + c$ для использования в алгоритме QS, полагая $c = (b^2 - n)/a$. Число вариантов значения $b$ при заданном значении $a$ равно $2^{k-1}$, где $a$ имеет $k$ различных простых делителей (при условии, что $a$ нечетно, и для каждого простого $p|a$ мы имеем $\left(\frac{n}{p}\right) = 1$). Так что если выбрать в качестве $a$ квадрат простого числа, как было предложено в разд. 6.1.5, то для $b$ останется одна единственная возможность. Пусть вместо этого мы выберем в качестве $a$ произведение 10 различных простых чисел $p$. Тогда существует $512 = 2^9$ вариантов значений $b$, соответствующих данному $a$, и тем самым вычисление $a^{-1} \pmod{p}$ надо провести лишь один раз, а затем использовать его для всех 512-и полиномов. Более того, если ни одно из 10 простых, используемых в $a$, не превосходит $B$, то они не обязаны входить в $a$ возведенными в квадрат, поскольку обнуление для них уже заложено в матричную стадию.

При автоматической инициализации можно сэкономить еще больше, если мы предварительно проделаем некоторые дополнительные вычисления и будем хранить некоторые файлы. Например, если мы вычисляем и храним список всех значений $2t(p)a^{-1} \bmod p$ для всех простых чисел $p$, по которым мы

просеиваем, то вычисление $r(p), s(p)$ из (6.3) можно осуществить, затратив одно умножение, а не два. А именно, умножить $-b + t(p)$ на хранимое значение $a^{-1} \bmod p$ и взять результат по модулю $p$. Это даст $r(p)$. Вычитая хранимое значение $2t(p)a^{-1} \bmod p$ и добавляя, в случае необходимости, $p$, получаем $s(p)$.

Можно даже избавиться от одного оставшегося умножения, упорядочив различные решения $b$ с помощью кода Грея (см. упр. 6.7). Дело в том, что китайская теорема об остатках (см. разд. 2.1.3) дает различные решения $b$ в виде $B_1 \pm B_2 \pm \ldots \pm B_k$. (Если $a = p_1 p_2 \ldots p_k$, то $B_i$ удовлетворяет сравнениям $B_i^2 \equiv n \pmod{p_i}$ и $B_i \equiv 0 \pmod{a/p_i}$.) Если мы упорядочим $2^{k-1}$ чисел $B_1 \pm B_2 \pm \ldots \pm B_k$ с помощью кода Грея и предварительно вычислим списки $2B_i a^{-1} \bmod p$ для всех $p$, по которым мы просеиваем, то мы сможем перейти от начальных координат просеивания для одного многочлена к координатам следующего, проделав лишь несколько малоразрядных операций сложения и вычитания для каждого $p$. Можно ограничиться хранением лишь наиболее часто используемых файлов со значениями $2B_i a^{-1} \bmod a$, если приоритетом является меньший объем памяти. Например, храня этот файл лишь для значений $i = k$, которые задействованы на каждом втором шаге в коде Грея, мы имеем крайне дешевую инициализацию в течение одной половины времени и выполняемую с помощью одного умножения по модулю $p$ для каждого $p$ (и нескольких сложений и вычитаний) в течение оставшегося времени.

Идея автоматической инициализации кратко описана в статье [Pomerance et al. 1988] и более подробно — в работах [Alford and Pomerance 1995] и [Peralta 1993]. В работе [Contini 1997] при помощи определенных экспериментов показано, что автоматическая инициализация дает примерно двукратное ускорение по сравнению со стандартными реализациями QS, использующими многие полиномы.

### 6.1.7. Специальное квадратичное решето Жанга

Быстрым квадратичное решето делает то, что мы имеем полиномиальную прогрессию, состоящую из малых квадратичных вычетов. То, что это квадратичные вычеты, делает их полезными для получения сравнимых между собой квадратов, которые могут разложить $n$. То, что это полиномиальная прогрессия (т. е. последовательные значения полинома), позволяет легко находить гладкие значения, а именно посредством решета. И, конечно, то, что они малы, делает вероятность их гладкости большей, чем у случайных вычетов по модулю $n$. Одним из возможных путей улучшения этого метода является нахождение полиномиальной прогрессии из еще меньших квадратичных вычетов. Не так давно М. Жанг обнаружил такой путь, но лишь для специальных значений $n$ (см. [Zhang 1998]). Мы будем называть его метод специальным квадратичным решетом или SQS (special quadratic sieve).

Пусть число $n$, которое мы пытаемся разложить (нечетное, составное и не являющееся степенью), может быть представлено в виде

$$n = m^3 + a_2 m^2 + a_1 m + a_0, \tag{6.4}$$

где $m, a_2, a_1, a_0$ — целые числа, $m \approx n^{1/3}$. На самом деле каждое число $n$ может быть представлено в таком виде. Просто берем $m = \lfloor n^{1/3} \rfloor$, полагаем $a_1 = a_2 = 0$ и $a_0 = n - m^3$. Однако далее мы увидим, что представление (6.4) будет полезно, лишь когда все $a_i$ малы по абсолютной величине, и, значит, мы рассматриваем лишь специальные значения $n$.

Пусть $b_0, b_1, b_2$ — целые переменные, и пусть

$$x = b_2 m^2 + b_1 m + b_0,$$

где $m$ такое же, как в формуле (6.4). Поскольку

$$\begin{aligned} m^3 &\equiv -a_2 m^2 - a_1 m - a_0 \pmod{n}, \\ m^4 &\equiv (a_2^2 - a_1) m^2 + (a_1 a_2 - a_0) m + a_0 a_2 \pmod{n}, \end{aligned}$$

мы имеем

$$x^2 \equiv c_2 m^2 + c_1 m + c_0 \pmod{n}, \tag{6.5}$$

где

$$\begin{aligned} c_2 &= (a_2^2 - a_1) b_2^2 - 2 a_2 b_1 b_2 + b_1^2 + 2 b_0 b_2, \\ c_1 &= (a_1 a_2 - a_0) b_2^2 - 2 a_1 b_1 b_2 + 2 b_0 b_1, \\ c_0 &= a_0 a_2 b_2^2 - 2 a_0 b_1 b_2 + b_0^2. \end{aligned}$$

Поскольку $b_0, b_1, b_2$ — свободные переменные, их, вероятно, можно выбрать малыми целыми числами так, чтобы $c_2 = 0$. Так оно и есть. Пусть

$$b_2 = 2, \quad b_1 = 2b, \quad b_0 = a_1 - a_2^2 + 2 a_2 b - b^2,$$

где $b$ — произвольное целое число. При таком выборе $b_0, b_1, b_2$ мы имеем

$$x(b)^2 \equiv y(b) \pmod{n}, \tag{6.6}$$

где

$$\begin{aligned} x(b) &= 2m^2 + 2bm + a_1 - a_2^2 + 2 a_2 b - b^2, \\ y(b) &= \left(4 a_1 a_2 - 4 a_0 - \left(4 a_1 + 4 a_2^2\right) b + 8 a_2 b^2 - 4 b^3\right) m \\ &\quad + 4 a_0 a_2 - 8 a_0 b + \left(a_1 - a_2^2 + 2 a_2 b - b^2\right)^2. \end{aligned}$$

План таков: заставить $b$ пробегать малые числа, использовать решето для поиска гладких значений $y(b)$ и затем использовать матрицу из векторов показателей для нахождения подмножества сравнений (6.6), по которому строятся два сравнимых квадрата по модулю $n$, которые затем могут быть использованы для разложения $n$. Если $a_0, a_1, a_2$ и $b$ порядка $O(n^\varepsilon)$, где $0 \leqslant \varepsilon < 1/3$ и $m = O\left(n^{1/3}\right)$, то $y(b) = O(n^{1/3+3\varepsilon})$. Анализ сложности из разд. 6.1.1 дает эвристическую оценку времени работы вида

$$L(n)^{\sqrt{2/3+6\varepsilon}+o(1)},$$

где $L(n)$ определяется равенством (6.1). Если $\varepsilon$ достаточно мало, то эта оценка улучшает эвристическую оценку сложности QS.

Также может оказаться полезным обобщение (6.4) вида

$$an = m^3 + a_2 m^2 + a_1 m + a_0.$$

Число $a$ не входит в выражения для $x(b), y(b)$, но влияет на величину числа $m$, которое будет порядка $(an)^{1/3}$.

Например, рассмотрим число $2^{601}-1$. Мы имеем два простых делителя — 3607 и 64863527, но число $n_0$, получающееся при делении на эти простые числа $2^{601}-1$, является составным, содержит 170 десятичных знаков, и нам не известны его делители. Мы имеем

$$2^2 \cdot 3607 \cdot 64863527 n_0 = 2^{603} - 2^2 = \left(2^{201}\right)^3 - 4,$$

так что мы можем взять $a_0 = -4$, $a_1 = a_2 = 0$, $m = 2^{201}$. Эти значения дают сравнение (6.6), в котором

$$x(b) = 2m^2 + 2bm - b^2, \quad y(b) = (16 - 4b^3)m + 32b + b^4, \quad m = 2^{201}.$$

По мере того как число $b$ растет по абсолютной величине, в $y(b)$ начинает преобладать член $-4b^3m$. Есть основания ожидать, что $b$ будет расти до $2^{40}$, и в таком случае величина $|y(b)|$ будет около $2^{323}$. Это не в лучшую сторону отличается от соответствующего показателя квадратичного решета с многими полиномами, в котором величина просеиваемых чисел, среди которых ищутся гладкие, была бы приблизительно равна $2^{20}\sqrt{n} \approx 2^{301}$. (Эта оценка предполагает, что длина интервала просеивания каждого полинома равна приблизительно $2^{20}$.)

Однако мы можем использовать многие полиномы и со специальным квадратичным решетом. Например, для нашего числа $n_0$ возьмем $b_0 = -2u^2$, $b_1 = 2uv$, $b_2 = v^2$. Тогда получается, что мы можем взять

$$x(u,v) = v^2m^2 + 2uvm - 2u^2, \quad y(u,v) = (4v^4 - 8u^3v)m + 16uv^3 + 4u^4$$

и заставить $u, v$ пробегать малые взаимно простые числа. (Важно брать $u, v$ взаимно простыми, поскольку в противном случае мы получили бы лишние соотношения.) Если разрешить значениям $u, v$ пробегать числа, по абсолютной величине не превосходящие $2^{20}$, то мы получим примерно то же количество пар, что и число возможностей для $b$, но теперь величина $|y(u,v)|$ порядка $2^{283}$, что означает экономию по сравнению с обычным квадратичным решетом. (Имеет место небольшая дополнительная экономия, так как на самом деле мы можем рассматривать пару $\frac{n-1}{2}x(u,v), \frac{1}{4}y(u,v)$.)

Возможно, не совсем ясно, почему введение переменных $u, v$ можно рассматривать как «многие полиномы». Суть в том, что мы можем фиксировать одну из этих переменных и просеивать по другой. Каждое фиксированное значение одной переменной дает новый полином относительно другой переменной.

Длина решета $2^{40}$, которая предполагалась в проведенном анализе, возможно, довольно мала для чисел такой величины как $n_0$. Бóльшая длина решета приведет к тому, что SQS будет выглядеть хуже в сравнении с обычным QS.

Не ясно, станет ли специальное квадратичное решето в том виде, как оно описано выше, полезным алгоритмом разложения (на момент написания этой книги оно фактически не было испытано на серьезных задачах). Если число $n$ не слишком велико, рост коэффициента при $m$ в формуле для $y(b)$ или $y(u,v)$ будет доминирующим и приведет к худшим показателям по сравнению с обыч-

ным квадратичным решетом. Если число $n$ несколько больше, так что специальное квадратичное решето начинает выглядеть более привлекательно, как в приведенном примере, то здесь, оказывается, существует другой алгоритм, который может вступить в игру и снова превзойти специальное квадратичное решето. Это — решето числового поля, его мы обсудим в следующем разделе.

## 6.2. Решето числового поля

В главе 5 мы сталкивались с некоторыми из новаторских идей Дж. Полларда. В 1988 г. (см. [Lenstra and Lenstra 1993]) Поллард предложил метод разложения на множители, который прекрасно подходил для чисел, близких к высокой степени (например, чисел Ферма). Вскоре этот метод был обобщен, и стало возможным его использование для составных чисел общего вида. В настоящее время решето числового поля (NFS — number field sieve) является асимптотически наилучшим эвристическим алгоритмом разложения из тех, которые известны нам для составных чисел, относящихся к «худшему случаю».

### 6.2.1. Базовый алгоритм NFS: стратегия

Метод квадратичного решета для разложения на множители работает быстро потому, что он производит малые квадратичные вычеты по модулю числа, которое мы раскладываем, и потому, что мы можем использовать решето для быстрого нахождения среди них гладких квадратичных вычетов. Метод QS был бы еще быстрее, если бы квадратичные вычеты, которые он производит, удалось сделать меньше, поскольку тогда вероятность их гладкости была бы больше, и, значит, нам не пришлось бы просеивать большое их количество. Интересная мысль на этот счет заключается в том, что им вовсе не обязательно быть квадратичными вычетами; достаточно того, что они малы! Линейная алгебра дает нам способ перемножать элементы подмножеств гладких чисел и тем самым получать квадраты. В квадратичном решете нам приходилось беспокоиться лишь об одной части нашего сравнения, поскольку другая его часть уже была квадратом. В решете числового поля мы используем метод линейной алгебры в обеих частях ключевого сравнения.

Однако наши сравнения не будут начинаться с двух целых чисел, сравнимых по модулю $n$. Они будут начинаться с пар $\theta, \phi(\theta)$, где $\theta$ лежит в специальном кольце алгебраических чисел, а $\phi$ является гомоморфизмом из этого кольца в $\mathbf{Z}_n$. (Эти понятия будут подробно описаны, но чуть позже.) Пусть мы имеем $k$ пар $\theta_1, \phi(\theta_1), \ldots, \theta_k, \phi(\theta_k)$, таких что произведение $\theta_1 \ldots \theta_k$ является квадратом в нашем числовом кольце, например, $\gamma^2$, и существует целый квадрат, например, $v^2$, такой что $\phi(\theta_1) \ldots \phi(\theta_k) \equiv v^2 \pmod{n}$. Тогда если $\phi(\gamma) \equiv u \pmod{n}$ для целого числа $u$, то мы имеем

$$u^2 \equiv \phi(\gamma)^2 \equiv \phi(\gamma^2) \equiv \phi(\theta_1 \ldots \theta_k) \equiv \phi(\theta_1) \ldots \phi(\theta_k) \equiv v^2 \pmod{n}.$$

Т. е., опуская все внутренние выражения, мы имеем сравнение $u^2 \equiv v^2 \pmod{n}$,

и потому мы могли бы попытаться разложить число $n$ на множители с помощью НОД $(u - v, n)$.

Вышеперечисленные идеи составляют стратегию NFS. Сейчас мы обсудим базовый подготовительный материал и введем понятия числового кольца и гомоморфизма $\phi$. Пусть мы пытаемся разложить число $n$, которое является нечетным, составным, и не является степенью. Пусть

$$f(x) = x^d + c_{d-1}x^{d-1} + \ldots + c_0$$

есть неприводимый многочлен из $\mathbf{Z}[x]$, и пусть $\alpha$ — комплексное число, являющееся корнем многочлена $f$. Нам не нужно численно аппроксимировать $\alpha$; мы просто используем символ «$\alpha$» для обозначения одного из корней многочлена $f$. Нашим числовым кольцом будет $\mathbf{Z}[\alpha]$. С вычислительной точки зрения его можно представлять себе как множество упорядоченных $d$-наборов целых чисел $(a_0, a_1, \ldots, a_{d-1})$, причем мы «изображаем» такой $d$-набор в виде элемента $a_0 + a_1\alpha + \ldots a_{d-1}\alpha^{d-1}$. Мы складываем два таких выражения покоординатно, а умножаем с помощью обычного правила умножения многочленов, только потом приводим к $d$-набору с помощью тождества $f(\alpha) = 0$. Другой эквивалентный способ представить себе числовое кольцо $\mathbf{Z}[\alpha]$ состоит в реализации его в виде $\mathbf{Z}[x]/(f(x))$, т. е. с использованием полиномиальной арифметики по модулю $f(x)$.

Связь с числом $n$, которое мы раскладываем, осуществляется посредством целого числа $m$, такого что

$$f(m) \equiv 0 \pmod{n}.$$

Нам необходимо знать, чему равно число $m$. Заметим, что существует очень простой метод обеспечения подходящего выбора $f(x)$ и $m$. Выберите степень $d$ нашего многочлена. (Ниже мы приведем эвристическое рассуждение относительно того, как выбрать $d$ таким образом, чтобы минимизировать время работы по разложению $n$. Эксперимент показывает, что для чисел, содержащих около 130-и цифр, $d = 5$ является подходящим выбором.) Положим $m = \lfloor n^{1/d} \rfloor$ и запишем число $n$ в системе с основанием $m$, так что

$$n = m^d + c_{d-1}m^{d-1} + \ldots + c_0,$$

где каждое $c_j \in [0, m-1]$. (Из упр. 6.8 мы получаем, что если $1.5(d/\ln 2)^d < n$, то $n < 2m^d$, так что коэффициент при $m^d$ в самом деле равен 1, как мы и указали выше.) Так что многочлен $f(x)$ получается прямо из записи числа $n$ в системе с основанием $m$: мы имеем $f(x) = x^d + c_{d-1}x^{d-1} + \ldots + c_0$, и старший коэффициент этого многочлена равен 1. Но он может не быть неприводимым. Оказывается, для нас это очень удачная ситуация, поскольку если у нас есть нетривиальное разложение $f(x) = g(x)h(x)$ в $\mathbf{Z}[x]$, то разложение целого числа $n = g(m)h(m)$ также нетривиально (см. [Brillhart et al. 1981] и упр. 6.9). Поскольку разложение многочлена — относительно легкая операция (см. [Lenstra et al. 1982], [Cohen 2000, p. 139]), то нам следует разложить $f$ на неприводимые многочлены в $\mathbf{Z}[x]$. Если это разложение нетривиально, то у нас есть нетривиальное разложение числа $n$. Если $f$ неприводим, то мы можем продолжать, следуя методу NFS.

Гомоморфизм $\phi$ из $\mathbf{Z}[\alpha]$ в $\mathbf{Z}_n$ определяется тем, что $\phi(\alpha)$ есть класс вычетов числа $m \pmod n$. Иными словами, $\phi$ сначала переводит $a_0+a_1\alpha+\ldots+a_{d-1}\alpha^{d-1}$ в целое число $a_0+a_1m+\ldots+a_{d-1}m^{d-1}$, а затем берет это число по модулю $n$. Будет полезно представлять себе $\phi$ в таком «двухступенчатом» виде, поскольку мы также будем работать с целым числом $a_0+a_1m+\ldots+a_{d-1}m^{d-1}$ до того, как оно будет браться по модулю.

Элементы $\theta$ из кольца $\mathbf{Z}[\alpha]$, которые мы будем рассматривать, все имеют вид $a-b\alpha$, где $a,b \in \mathbf{Z}$, причем НОД$(a,b)=1$. Таким образом, мы ищем множество $\mathcal{S}$, составленное из пар взаимно простых целых чисел $(a,b)$ таких, что

$$\prod_{(a,b)\in\mathcal{S}} (a-b\alpha) = \gamma^2 \text{ для некоторого } \gamma \in \mathbf{Z}[\alpha],$$

$$\prod_{(a,b)\in\mathcal{S}} (a-bm) = v^2 \text{ для некоторого } v \in \mathbf{Z}.$$

Тогда если $u$ — целое число, такое что $\phi(\gamma) \equiv u \pmod n$, то, как и выше, $u^2 \equiv v^2 \pmod n$, и мы можем попытаться разложить $n$ с помощью НОД $(u-v,n)$. (Пары $(a,b)$ из множества $\mathcal{S}$ предполагаются взаимно простыми в целях избежания тривиальных избыточных соотношений.)

### 6.2.2. Базовый метод NFS: векторы показателей

Как же мы планируем найти множество $\mathcal{S}$ пар $(a,b)$? Метод схож с тем, что мы используем в квадратичном решете. Там мы имеем одну переменную, которая пробегает интервал. Мы используем решето для обнаружения гладких значений заданного полинома, ассоциируем векторы показателей с этими гладкими значениями и используем линейную алгебру для нахождения подмножества значений с произведением, равным квадрату. В методе NFS у нас две переменных — $a,b$. Как и в специальном квадратичном решете (см. разд. 6.1.7) мы можем зафиксировать одну переменную и просеивать по другой, затем перейти к следующему значению первой переменной и опять просеивать по другой и т. д.

А что же нам просеивать? Для начала ответим на более простой вопрос. Не будем обращать внимания на задачу о получении произведения чисел $a-b\alpha$, равного квадрату в $\mathbf{Z}[\alpha]$, а сосредоточимся лишь на втором свойстве, которым должно обладать $\mathcal{S}$, а именно, что произведение чисел $a-bm$ должно являться квадратом в $\mathbf{Z}$. Здесь $m$ — фиксированное целое число, которое мы вычисляем в самом начале. Скажем, мы заставляем $a,b$ пробегать пары целых чисел, таких, что $0<|a|,b\leqslant M$, где $M$ есть некоторая большая граница (настолько большая, что число пар $a,b$ будет достаточным для успешной работы). Тогда у нас есть просто однородный многочлен $G(a,b)=a-bm$ 1-й степени, который мы просеиваем на предмет гладких значений, скажем, $B$-гладких. Мы отбрасываем любую найденную пару $(a,b)$, такую, что НОД$(a,b)>1$. Как только мы найдем более, чем $\pi(B)+1$ таких пар, на векторах показателей, отвечающих гладким значениям многочлена $G(a,b)$, можно использовать линейную

алгебру по модулю 2 для нахождения подмножества значений, произведение которых является квадратом.

Все бы хорошо, но мы проигнорировали наиболее трудную часть нашей задачи: сделать так, чтобы наше множество пар $(a,b)$ *одновременно* обладало тем дополнительным свойством, что произведение чисел $a-b\alpha$ является квадратом в $\mathbf{Z}[\alpha]$.

Пусть корнями многочлена $f(x)$ в комплексных числах являются $\alpha_1,\ldots,\alpha_d$, где $\alpha=\alpha_1$. Норма элемента $\beta=s_0+s_1\alpha+\ldots+s_{d-1}\alpha^{d-1}$ в поле алгебраических чисел $\mathbf{Q}[\alpha]$ (где коэффициенты $s_0,s_1,\ldots,s_{d-1}$ — произвольные рациональные числа) есть произведение комплексных чисел $s_0+s_1\alpha_j+\ldots+s_{d-1}\alpha_j^{d-1}$ для $j=1,2,\ldots,d$. Данное комплексное число, обозначаемое $N(\beta)$, на самом деле является рациональным, поскольку равно симметрическому выражению от корней $\alpha_1,\ldots,\alpha_d$, а элементарные симметрические многочлены от этих корней равны $\pm c_j$ при $j=0,1,\ldots,d-1$, причем эти коэффициенты — целые числа. В частном случае, если все рациональные числа $s_j$ являются целыми, то $N(\beta)$ также целое. (В дальнейшем мы будем обращаться к понятию «след числа $\beta$». Это есть сумма сопряженных элементов $s_0+s_1\alpha_j+\ldots+s_{d-1}\alpha_j^{d-1}$ для $j=1,2,\ldots,d$.)

Также довольно легко показать, что функция нормы $N$ является мультипликативной, т. е. $N(\beta\beta')=N(\beta)N(\beta')$. Отсюда получается важное следствие: если $\beta=\gamma^2$ для некоторого $\gamma\in\mathbf{Z}[\alpha]$, то $N(\beta)$ есть квадрат целого числа, а именно числа $N(\gamma)$.

Таким образом, *необходимым* условием того, что произведение чисел $a-b\alpha$ для $(a,b)$ из $\mathcal{S}$ равно квадрату в $\mathbf{Z}[\alpha]$ является то, что соответствующее произведение целых чисел $N(a-b\alpha)$ равно квадрату в $\mathbf{Z}$. На время оставим в стороне вопрос о *достаточности* этого условия и посмотрим, как сделать так, чтобы произведение чисел $N(a-b\alpha)$ было квадратом.

Сначала заметим, что

$$\begin{aligned} N(a-b\alpha) &= (a-b\alpha_1)\ldots(a-b\alpha_d)\\ &= b^d(a/b-\alpha_1)\ldots(a/b-\alpha_d)\\ &= b^d f(a/b), \end{aligned}$$

поскольку $f(x)=(x-\alpha_1)\ldots(x-\alpha_d)$. Пусть $F(x,y)$ есть однородная форма многочлена $f$, а именно,

$$F(x,y)=x^d+c_{d-1}x^{d-1}y+\ldots+c_0y^d=y^df(x/y).$$

Тогда $N(a-b\alpha)=F(a,b)$. Значит, число $N(a-b\alpha)$ можно довольно ясно представить себе как многочлен от двух переменных $a,b$.

Таким образом, мы можем сделать так, чтобы произведение чисел $N(a-b\alpha)$ для $(a,b)\in\mathcal{S}$ было квадратом, если заставим $a,b$ пробегать целые числа так, что $|a|,|b|\leqslant M$, применим решето для обнаружения $B$-гладких значений многочлена $F(a,b)$, составим соответствующие векторы показателей и применим матричные методы, чтобы найти подмножество $\mathcal{S}$. А если мы хотим, чтобы $\mathcal{S}$ обладало также и первым свойством (произведение чисел $a-bm$ является квадратом в $\mathbf{Z}$), то мы модифицируем процедуру и просеиваем на предмет гладких

значений произведение $F(a,b)G(a,b)$, которое также является многочленом от переменных $a, b$. Для найденных гладких значений мы строим векторы показателей, разбитые на *два поля* координат. Первое поле соответствует разложению на простые сомножители числа $F(a,b)$, а второе — разложению числа $G(a,b)$. Эти удлиненные векторы показателей затем собираются в матрицу, и мы снова можем проделать линейную алгебру по модулю 2. До этого нам было нужно всего $\pi(B)+2$ вектора, чтобы гарантировать успех. Теперь для гарантии успеха нам необходимо $2\pi(B)+3$ вектора, поскольку каждый вектор будет иметь $2\pi(B)+2$ координат: первая половина отвечает разложению на простые сомножители числа $F(a,b)$, а вторая половина — разложению числа $G(a,b)$. Так что нам нужно лишь набрать в два раза больше векторов, и тогда мы сможем решить обе эти задачи одновременно.

Теперь вернемся к вопросу о достаточности. А именно если $N(\beta)$ является квадратом в $\mathbf{Z}$, и $\beta \in \mathbf{Z}[\alpha]$, то верно ли, что $\beta$ является квадратом в $\mathbf{Z}[\alpha]$? Ответом будет обескураживающее «нет». По-видимому, поучительно рассмотреть простой пример. Рассмотрим случай многочлена $f(x)=x^2+1$, обозначив корень символом «$i$» (как Вы уже, наверное, догадались). Тогда $N(a+bi)=a^2+b^2$. Если $a^2+b^2$ равно квадрату в $\mathbf{Z}$, то $a+bi$ не обязано быть квадратом в $\mathbf{Z}[i]$. Например, если $a$ — положительное целое число, не являющееся квадратом, то оно не является квадратом и в $\mathbf{Z}[i]$, хотя $N(a)=a^2$ является квадратом в $\mathbf{Z}$.

На самом деле, кольцо $\mathbf{Z}[i]$, известное под названием кольца гауссовых чисел, является хорошо изученным кольцом с множеством замечательных свойств, находящихся в полной аналогии с кольцом $\mathbf{Z}$. Гауссовы числа являются областью с однозначным разложением на множители, как и $\mathbf{Z}$. Каждое простое число из $\mathbf{Z}[i]$ «лежит поверх» обычного простого числа $p$ из $\mathbf{Z}$. Если это простое число $p$ равно $1 \pmod 4$, то оно может быть записано в виде $a^2+b^2$, и тогда $a+bi$ и $a-bi$ будут теми двумя различными простыми из $\mathbf{Z}[i]$, которые лежат поверх $p$. (Каждое простое число имеет четыре «ассоциированных», получающихся умножением на четыре единицы: $1, -1, i, -i$. Ассоциированные простые числа рассматриваются как одно простое число, поскольку порождаемые ими главные идеалы в точности совпадают.) Если обычное простое число $p$ равно $3 \pmod 4$, то оно остается простым и в $\mathbf{Z}[i]$. А поверх простого числа 2 лежит единственное простое число $1+i$ (и ассоциированные с ним). Дополнительные сведения об арифметике гауссовых чисел см. в книге [Niven et al. 1991].

Итак, мы можем увидеть, например, что $5i$ определенно не является квадратом в $\mathbf{Z}[i]$, поскольку это число имеет разложение на простые сомножители вида $(2+i)(1+2i)$, а $2+i$ и $1+2i$ — различные простые. (Напротив, $2i$ *является* квадратом, оно равно $(1+i)^2$.) Однако $N(5i)=25$, и, разумеется, 25 мы рассматриваем как квадрат в $\mathbf{Z}$. Проблема состоит в том, что функция нормы смешивает два различных простых числа $1+2i$ и $2+i$. Поэтому нам хотелось бы иметь некоторый способ, позволяющий отличать друг от друга различные простые.

Если бы в решете числового поля наше кольцо $\mathbf{Z}[\alpha]$ было областью с однозначным разложением на множители, то поставленная нами задача была

бы значительно проще: мы просто составляем векторы показателей на основе разложения на простые сомножители различных элементов $a - b\alpha$. Здесь есть проблема с единицами, и если бы мы решили встать на этот путь, мы бы также захотели найти систему «фундаментальных единиц» и для каждой из них иметь координату в наших векторах показателей. (В случае кольца $\mathbf{Z}[i]$ фундаментальная единица довольно тривиальна, она есть просто $i$, и мы можем взять в качестве различных простых те простые из каждого класса ассоциированных, которые лежат в первом квадранте, но не лежат на мнимой оси.)

Однако мы увидим, что решето числового поля может запросто работать даже тогда, когда кольцо $\mathbf{Z}[\alpha]$ отнюдь не является областью с однозначным разложением на множители, и даже тогда, когда мы понятия не имеем о его единицах.

Для каждого простого числа $p$ обозначим через $R(p)$ множество целых чисел $r \in [0, p-1]$, таких что $f(r) \equiv 0 \pmod{p}$. Например, если $f(x) = x^2 + 1$, то $R(2) = \{1\}$, $R(3) = \{\ \}$ и $R(5) = \{2, 3\}$. Тогда если $a, b$ — взаимно простые числа, то

$F(a,b) \equiv 0 \pmod{p}$ тогда и только тогда, когда $a \equiv br \pmod{p}$ для некоторого $r \in R(p)$.

Следовательно, если мы обнаружим, что $p|F(a,b)$, то мы будем обладать и остальной информацией, а именно числом $r \in R(p)$, таким что $a \equiv br \pmod{p}$. (На самом деле, множества $R(p)$ применяются в решете, которое мы используем для разложения чисел $F(a,b)$. Мы можем зафиксировать число $b$ и рассматривать $F(a,b)$ как многочлен от переменной $a$. Тогда при просеивании по простому числу $p$ мы просеиваем классы вычетов $a \equiv br \pmod{p}$ для нахождения значений, кратных $p$.) Мы будем отражать эту дополнительную информацию в наших векторах показателей. Координатное поле наших векторов показателей, отвечающее разложению числа $F(a,b)$, будет содержать по одной компоненте на каждую пару $p, r$, где $p$ — простое число $\leqslant B$, а $r \in R(p)$.

Давайте снова рассмотрим многочлен $f(x) = x^2 + 1$. Если $B = 5$, то векторы показателей для $B$-гладких элементов $\mathbf{Z}[i]$ (т. е. для тех элементов $\mathbf{Z}[i]$, чьи нормы есть $B$-гладкие целые числа) будут иметь три координаты, отвечающие трем парам: (2,1), (5,2) и (5,3). Тогда

$$\begin{aligned} F(3,1) &= 10 \text{ имеет вектор показателей } (1,0,1), \\ F(2,1) &= 5 \text{ имеет вектор показателей } (0,1,0), \\ F(1,1) &= 2 \text{ имеет вектор показателей } (1,0,0), \\ F(2,-1) &= 5 \text{ имеет вектор показателей } (0,0,1). \end{aligned}$$

Хотя $F(3,1)F(2,1)F(1,1) = 100$ есть квадрат, векторы показателей позволяют нам увидеть, что $(3+i)(2+i)(1+i)$ *не есть* квадрат: сумма трех соответствующих векторов по модулю 2 равна $(0,1,1)$, а это ненулевой вектор. А теперь рассмотрим $(3+i)(2-i)(1+i) = 8+6i$. Сумма трех соответствующих векторов показателей по модулю 2 равна $(0,0,0)$, и действительно, $8+6i$ есть квадрат в $\mathbf{Z}[i]$.

Этот метод содержит небольшие неточности. Например, хотя в вышеуказанной схеме число $i$ имеет своим вектором показателей нулевой вектор, оно

не является квадратом. Если бы это было единственной проблемой (мы имеем в виду вопрос о единицах), то мы могли бы довольно легко найти ее решение. Однако это не единственная проблема.

Обозначим через $\mathcal{I}$ кольцо целых алгебраических чисел в поле алгебраических чисел $\mathbf{Q}[\alpha]$. Другими словами, $\mathcal{I}$ есть множество элементов из $\mathbf{Q}[\alpha]$, являющихся корнями некоторого многочлена из $\mathbf{Z}[x]$ со старшим коэффициентом единица. Множество $\mathcal{I}$ замкнуто относительно умножения и сложения. Значит, это кольцо (см. [Marcus 1977]). В случае многочлена $f(x) = x^2 + 1$ целые алгебраические числа в $\mathbf{Q}[i]$ составляют в точности кольцо $\mathbf{Z}[i]$. Кольцо $\mathbf{Z}[\alpha]$ будет всегда подмножеством $\mathcal{I}$, но, вообще говоря, оно может быть собственным подмножеством. Например, рассмотрим случай, когда $f(x) = x^2 - 5$. Кольцо всех целых алгебраических чисел в $\mathbf{Q}\left[\sqrt{5}\right]$ есть $\mathbf{Z}\left[(1+\sqrt{5})/2\right]$, которое содержит $\mathbf{Z}\left[\sqrt{5}\right]$ в качестве собственного подмножества.

Резюмируем ситуацию с векторами показателей для чисел $a - b\alpha$. Мы говорим, что число $a - b\alpha$ является $B$-гладким, если его норма $N(a - b\alpha) = F(a, b)$ является $B$-гладкой. Для взаимно простых $a, b$ мы связываем с $B$-гладким числом $a - b\alpha$ вектор показателей $\vec{v}(a - b\alpha)$, имеющий компоненты $v_{p,r}(a - b\alpha)$ для каждой пары $(p, r)$, где $p$ — простое число, не превосходящее $B$ и $r \in R(p)$. (В дальнейшем мы будем использовать обозначение $\vec{v}(a - b\alpha)$ для удлиненного вектора, который содержит в себе тот вектор, что мы здесь рассматриваем.) Если $a \not\equiv br \pmod{p}$, то мы полагаем $v_{p,r}(a - b\alpha) = 0$. В противном случае $a \equiv br \pmod{p}$ и $v_{p,r}(a - b\alpha)$ определяется как показатель степени числа $p$ в разложении числа $F(a, b)$ на простые сомножители. Справедлив следующий важный результат.

**Лемма 6.2.1.** *Если $\mathcal{S}$ есть множество пар взаимно простых целых чисел $a, b$, такое, что каждое $a - b\alpha$ является $B$-гладким, и если $\prod_{(a,b)\in\mathcal{S}}(a - b\alpha)$ является квадратом некоторого элемента из $\mathcal{I}$, кольца целых алгебраических чисел в $\mathbf{Q}[\alpha]$, то*

$$\sum_{(a,b)\in\mathcal{S}} \vec{v}(a - b\alpha) \equiv \vec{0} \pmod{2}. \tag{6.7}$$

Доказательство. Начнем с краткого обсуждения того, что представляют собой числа $v_{p,r}(a - b\alpha)$. В теории алгебраических чисел хорошо известно, что кольцо $\mathcal{I}$ является областью Дедекинда (см. [Marcus 1977]). В частности, ненулевые идеалы в $\mathcal{I}$ можно однозначным образом разложить на простые идеалы. Мы также воспользуемся понятием нормы идеала: если $J$ есть ненулевой идеал в $\mathcal{I}$, то $N(J)$ есть число элементов (конечного) фактор-кольца $\mathcal{I}/J$. (Норма нулевого идеала по определению равна нулю.) Функция нормы мультипликативна на идеалах, т. е. $N(J_1J_2) = N(J_1)N(J_2)$ для любых идеалов $J_1, J_2$ в $\mathcal{I}$. Связь между нормой элемента из $\mathcal{I}$ и нормой главного идеала, порождаемого им, выглядит удивительно: если $\beta \in \mathcal{I}$, то $N((\beta)) = |N(\beta)|$.

Если $p$ — простое число, и $r \in R(p)$, пусть $P_1, \ldots, P_k$ являются простыми идеалами в $\mathcal{I}$, которые делят идеал $(p, \alpha - r)$. (Этот идеал неединичный, поскольку $N(\alpha - r) = f(r)$, а это целое число делится на $p$.) Существуют положительные целые числа $e_1, \ldots, e_k$, такие, что $N(P_j) = p^{e_j}$ для $j = 1, \ldots, k$.

Обычно ситуация такова, что $k = 1$, $e_1 = 1$, и $(p, \alpha - r) = P_1$. А именно, данный сценарий имеет место в тех случаях, когда $p$ не делит индекс кольца $\mathbf{Z}[\alpha]$ в $\mathcal{I}$ (см. [Marcus 1977]). Как бы то ни было, мы будем проводить доказательство в общем случае.

Заметим, что если $r' \in R(p)$, и $r' \neq r$, то простые идеалы, которые делят $(p, \alpha - r)$, отличны от простых идеалов, которые делят $(p, \alpha - r')$, т. е. идеалы $(p, \alpha - r')$ и $(p, \alpha - r)$ взаимно просты. Это замечание справедливо, поскольку целое число $r - r'$ взаимно просто с простым числом $p$. Кроме того, если $a, b$ — целые числа, то $a - b\alpha \in (p, \alpha - r)$ тогда и только тогда, когда $a \equiv br \pmod{p}$. Чтобы убедиться в этом, запишем $a - b\alpha = a - br - b(\alpha - r)$, так что $a - b\alpha \in (p, \alpha - r)$ равносильно тому, что $a - br \in (p, \alpha - r)$, и равносильно тому, что $a \equiv br \pmod{p}$. Нам понадобится еще одно свойство: если $a, b$ — взаимно простые целые числа, $a \equiv br \pmod{p}$, и $P$ — простой идеал в $\mathcal{I}$, который делит как $(p)$, так и $(a - b\alpha)$, то $P$ делит $(p, \alpha - r)$, т. е. $P$ является одним из $P_j$. Для доказательства заметим, что из предположения о том, что $a, b$ взаимно просты, и что $a \equiv br \pmod{p}$, следует, что $b \not\equiv 0 \pmod{p}$, так что существует целое число $c$, такое, что $cb \equiv 1 \pmod{p}$. Далее, поскольку $a - b\alpha = a - br - b(\alpha - r) \in P$, и $a - br \equiv 0 \pmod{p}$, мы получаем, что $b(\alpha - r) \in P$, так что $cb(\alpha - r) \in P$, и $\alpha - r \in P$. Таким образом, $P$ делит $(p, \alpha - r)$, что и требовалось.

Пусть $a, b$ — взаимно простые целые числа, и пусть произведение $P_1^{a_1} \dots P_k^{a_k}$ входит в разложение идеала $(a - b\alpha)$ на простые идеалы. Как мы только что видели, если какой-либо из этих показателей $a_j$ положителен, то условие $a \equiv br \pmod{p}$ является необходимым и достаточным для того, чтобы все показатели $a_j$ были положительны, и ни один из оставшихся простых идеальных делителей идеала $(p)$ не делил идеал $(a - b\alpha)$. Таким образом, «$p$-составляющая» нормы числа $a - b\alpha$ есть в точности норма идеала $P_1^{a_1} \dots P_k^{a_k}$, т. е.

$$p^{v_{p,r}(a-b\alpha)} = N(P_1^{a_1} \dots P_k^{a_k}) = p^{e_1 a_1 + \dots + e_k a_k}.$$

Пусть $v_P(a - b\alpha)$ обозначает показатель степени простого идеала $P$ в разложении $(a - b\alpha)$ на простые идеалы. Тогда из последних равенств имеем

$$v_{p,r}(a - b\alpha) = \sum_{j=1}^{k} e_j v_{P_j}(a - b\alpha).$$

Итак, если произведение $\prod_{(a,b)\in\mathcal{S}}(a - b\alpha)$ является квадратом в $\mathcal{I}$, то главный идеал, который оно порождает, есть квадрат некоторого идеала. Следовательно, для каждого простого идеала $P$ в $\mathcal{I}$ мы имеем четную сумму $\sum_{(a,b)\in\mathcal{S}} v_P(a - b\alpha)$. Применим этот принцип к простым идеалам $P_j$, делящим идеал $(p, \alpha - r)$. Мы получаем

$$\sum_{(a,b)\in\mathcal{S}} v_{p,r}(a - b\alpha) = \sum_{j=1}^{k} e_j \sum_{(a,b)\in\mathcal{S}} v_{P_j}(a - b\alpha).$$

Поскольку каждая внутренняя сумма в правой части этого равенства есть четное целое число, целое число в левой части этого равенства также должно быть четным. □

### 6.2.3. Базовый метод NFS: сложность

Мы пока еще не дали полного описания метода NFS, но, вероятно, имеет смысл осознать, почему излагаемая стратегия ведет к быстрому методу разложения на множители, и наряду с этим получить представление о порядке величины выбираемых параметров.

Как в методе QS, так и в методе NFS, мы имеем дело с потоком чисел, на котором мы можем использовать решето для обнаружения гладких значений. Когда мы имеем достаточное число гладких значений, мы можем использовать линейную алгебру на векторах показателей, отвечающих гладким значениям, для нахождения непустого подмножества этих векторов, сумма по которому есть нулевой вектор по модулю 2. Сформулируем общую задачу следующим образом. Мы имеем случайную последовательность положительных целых чисел, ограниченных величиной $X$. Каково ожидаемое число членов этой последовательности, дающее нам нетривиальную подпоследовательность с произведением, равным квадрату? Эвристический анализ из разд. 6.1.1 дает ответ на этот вопрос: это число оценивается сверху величиной $L(X)^{\sqrt{2}+o(1)}$, причем граница гладкости, позволяющая получить эту оценку, равна $L(X)^{1/\sqrt{2}}$. (Мы используем здесь обозначение (6.1).) Эта эвристическая верхняя граница на самом деле может быть строго доказана в виде двухсторонней оценки с помощью следующей теоремы.

**Теорема 6.2.2** (Померанс 1996). *Пусть $m_1, m_2, \ldots$ есть последовательность целых чисел из промежутка $[1, X]$, каждое из которых выбирается независимо и по закону равномерного распределения. Пусть $N$ есть наименьшее целое число, такое что непустая подпоследовательность последовательности $m_1, m_2, \ldots, m_N$ имеет произведение, равное квадрату. Тогда ожидаемое значение для $N$ равно $L(X)^{\sqrt{2}+o(1)}$. Та же самая оценка будет справедлива, если мы дополнительно потребуем, чтобы каждое $m_j$, использованное в нашем произведении, было $B$-гладким, где $B = L(X)^{1/\sqrt{2}}$.*

Таким образом, в определенном смысле от гладких чисел нам никуда не деться, и они не просто артефакт. Интересно, что точно такая же теорема существует для случайной величины $N'$, равной наименьшему целому числу, такому что $m_1, m_2, \ldots, m_{N'}$ «мультипликативно зависимы», т. е. существуют целые числа $a_1, a_2, \ldots, a_{N'}$, не все равные нулю, такие, что $\prod m_j^{a_j} = 1$. (Или, что равносильно, числа $\ln m_1, \ln m_2, \ldots, \ln m_{N'}$ линейно зависимы над $\mathbf{Q}$.)

При анализе QS граница $X$ равна $n^{1/2+o(1)}$, именно поэтому мы получаем для QS оценку сложности вида $L(n)^{1+o(1)}$. Эта оценка сложности не является теоремой, поскольку числа, которые мы проверяем с целью формирования квадратов, не являются случайными; мы лишь предполагаем их случайность для удобства анализа.

Данный подход, таким образом, кажется достаточно удобным для проведения анализа сложности. Просто находим границу $X$ для чисел, которые мы пытаемся скомбинировать для составления квадратов. Чем меньше граница $X$, тем меньше сложность алгоритма. В методе NFS целые числа, с

которыми мы работаем, являются значениями многочлена $F(x,y)G(x,y)$, где $F(x,y) = x^d + c_{d-1}x^{d-1}y + \ldots + c_0y^d$ и $G(x,y) = x - my$. Мы не будем учитывать тот факт, что целые числа вида $F(a,b)G(a,b)$ уже разложены в произведение двух чисел, и поэтому могут оказаться гладкими с большей вероятностью, чем случайные числа того же порядка; данное свойство оказывает незначительное влияние на асимптотику сложности.

Пусть целое число $m$ в методе NFS ограничено величиной $n^{1/d}$, коэффициенты $c_j$ многочлена $f(x)$ также ограничены величиной $n^{1/d}$, и пусть мы исследуем такие значения $a, b$, что $|a|, |b| \leqslant M$. Тогда оценка для чисел $|F(a,b)G(a,b)|$ имеет вид $2(d+1)n^{2/d}M^{d+1}$. Если мы обозначим данное число через $X$, то ввиду теоремы 6.2.2 мы можем ожидать, что придется проверить $L(X)^{\sqrt{2}+o(1)}$ пар $a, b$, чтобы найти столько, сколько нужно для завершения алгоритма. Следовательно, $M$ должно удовлетворять условию $M^2 = L(X)^{\sqrt{2}+o(1)}$. Подставляя его в равенство $X = 2(d+1)n^{2/d}M^{d+1}$ и логарифмируя обе части, получаем

$$\ln X \sim \ln(2(d+1)) + \frac{2}{d}\ln n + (d+1)\sqrt{\frac{1}{2}\ln X \ln\ln X}. \tag{6.8}$$

Ясно, что первый член в правой части мал по сравнению с третьим членом. Допустим сначала, что $d$ фиксировано, т. е. мы собираемся проанализировать сложность метода NFS при фиксированной степени многочлена $f(x)$ и в предположении, что $n \to \infty$. Тогда последний член в правой части (6.8) мал по сравнению с левой частью, так что (6.8) сводится к

$$\ln X \sim \frac{2}{d}\ln n.$$

Следовательно, оценка времени работы при фиксированном $d$ есть

$$L(X)^{\sqrt{2}+o(1)} = L(n)^{\sqrt{4/d}+o(1)},$$

а это говорит о том, что NFS не будет лучше QS, пока мы не возьмем $d = 5$ или больше.

Пусть теперь $d \to \infty$ при $n \to \infty$. Тогда мы можем заменить коэффициент $d+1$ в последнем члене соотношения (6.8) на $d$ и получить

$$\ln X \sim \frac{2}{d}\ln n + d\sqrt{\frac{1}{2}\ln X \ln\ln X}.$$

Допустив некоторую неточность, заменим знак «$\sim$» на «$=$» и попытаемся выбрать $d$ так, чтобы минимизировать $X$. (Оптимальное значение $X_0$ будет таково, что $\ln X_0 \sim \ln X$.) Вычислив производную по «переменной» $d$, получим

$$\frac{X'}{X} = \frac{-2}{d^2}\ln n + \sqrt{\frac{1}{2}\ln X \ln\ln X} + \frac{dX'(1+\ln\ln X)}{4X\sqrt{\frac{1}{2}\ln X \ln\ln X}}.$$

Подставляя $X' = 0$, получаем

$$d = (2\ln n)^{1/2}((1/2)\ln X \ln\ln X)^{-1/4},$$

так что

$$\ln X = 2(2\ln n)^{1/2}((1/2)\ln X \ln\ln X)^{1/4}.$$

Тогда

$$(\ln X)^{3/4} = 2(2\ln n)^{1/2}((1/2)\ln\ln X)^{1/4},$$

так что $\frac{3}{4}\ln\ln X \sim \frac{1}{2}\ln\ln n$. Подставляя, получаем

$$(\ln X)^{3/4} \sim 2(2\ln n)^{1/2}((1/3)\ln\ln n)^{1/4},$$

или

$$\ln X \sim \frac{4}{3^{1/3}}(\ln n)^{2/3}(\ln\ln n)^{1/3}.$$

Таким образом, время работы NFS есть

$$L(X)^{\sqrt{2}+o(1)} = \exp\left(\left((64/9)^{1/3}+o(1)\right)(\ln n)^{1/3}(\ln\ln n)^{2/3}\right).$$

Значения $d$, позволяющие получить данную эвристическую асимптотическую оценку сложности, удовлетворяют условию

$$d \sim \left(\frac{3\ln n}{\ln\ln n}\right)^{1/3}$$

Мы видим, что «на бесконечности» NFS (эвристически) значительно превосходит QS. Более низкая оценка сложности должна заставить нас сосредоточиться и решить оставшиеся технические проблемы, связанные с алгоритмом.

Если бы мы могли предоставить многочлен с меньшими коэффициентами, то оценка сложности была бы меньше. А именно, если многочлен $f(x)$ имеет коэффициенты, ограниченные величиной $n^{\epsilon/d}$, то вышеприведенный анализ дает сложность $L(n)^{\sqrt{(2+2\epsilon)/d}+o(1)}$ для фиксированного $d$; а для $d \to \infty$ при $n \to \infty$ она равна $\exp\left(\left((32(1+\varepsilon)/9)^{1/3}+o(1)\right)(\ln n)^{1/3}(\ln\ln n)^{2/3}\right)$. Случай $\varepsilon = o(1)$ соответствует «специальному» решету числового поля (см. разд. 6.2.7).

## 6.2.4. Базовый метод NFS: затруднения

После этого экскурса в теорию сложности мы возвращаемся к стратегии метода NFS. Мы займемся поиском некоторого легко проверяемого условия для того, чтобы произведение чисел $(a-b\alpha)$ по $(a,b) \in \mathcal{S}$ было квадратом в $\mathbf{Z}[\alpha]$. Лемма 6.2.1 делает шаг на пути к этому условию, но некоторые «затруднения» пока еще остаются. Пусть выполнено условие (6.7). Положим $\beta = \prod_{(a,b)\in\mathcal{S}}(a-b\alpha)$.

(1) Если кольцо $\mathbf{Z}[\alpha]$ совпадает с $\mathcal{I}$ (кольцом всех целых алгебраических чисел в $\mathbf{Q}(\alpha)$), то мы по крайней мере имеем, что идеал $(\beta)$ в $\mathcal{I}$ равен квадрату некоторого идеала $J$. Но равенства $\mathbf{Z}[\alpha] = \mathcal{I}$ может и не быть. Так что идеал $(\beta)$ в $\mathcal{I}$ может и не быть квадратом некоторого идеала в $\mathcal{I}$.

(2) Даже если $(\beta) = J^2$ для некоторого идеала $J$ в $\mathcal{I}$, то $J$ может и не быть главным идеалом.

(3) Даже если $(\beta) = (\gamma)^2$ для некоторого $\gamma \in \mathcal{I}$, то равенства $\beta = \gamma^2$ может и не быть.

(4) Даже если $\beta = \gamma^2$ для некоторого $\gamma \in \mathcal{I}$, может и не быть так, что $\gamma \in \mathbf{Z}[\alpha]$.

Хотя эти четыре затруднения выглядят неприступными, мы увидим, что можно использовать два простых приема, чтобы преодолеть все четыре. Начнем с последнего из этих затруднений. Здесь представляет интерес следующая лемма.

**Лемма 6.2.3.** *Пусть $f(x)$ есть неприводимый многочлен из $\mathbf{Z}[x]$ со старшим коэффициентом единица, обладающий корнем $\alpha$ в комплексных числах. Пусть $\mathcal{I}$ есть кольцо целых алгебраических чисел из $\mathbf{Q}(\alpha)$, и пусть $\beta \in \mathcal{I}$. Тогда $f'(\alpha)\beta \in \mathbf{Z}[\alpha]$.*

Доказательство. Наше доказательство следует рассуждению в книге [Weiss 1963, Sections 3–7]. Пусть $\beta_0, \beta_1, \ldots, \beta_{d-1}$ — коэффициенты многочлена $f(x)/(x-\alpha)$. Т. е. $f(x)/(x-\alpha) = \sum_{j=0}^{d-1} \beta_j x^j$. Из утверждения 3-7-12 книги [Weiss 1963] (этот результат приписывают Эйлеру) мы имеем, что $\beta_0/f'(\alpha), \ldots, \beta_{d-1}/f'(\alpha)$ есть базис для $\mathbf{Q}(\alpha)$ над $\mathbf{Q}$, каждое $\beta_j \in \mathbf{Z}[\alpha]$, и след элемента $\alpha^k \beta_j / f'(\alpha)$ равен 1, если $j = k$, и 0 — в противном случае. (Определение следа см. в разд. 6.2.2. Из этого определения нетрудно вывести, что операция взятия следа $\mathbf{Q}$-линейна, принимает значения в $\mathbf{Q}$, и на элементах кольца $\mathcal{I}$ принимает значения в $\mathbf{Z}$.) Пусть $\beta \in \mathcal{I}$. Существуют рациональные числа $s_0, \ldots, s_{d-1}$, такие, что $\beta = \sum_{j=0}^{d-1} s_j \beta_j / f'(\alpha)$. Тогда след элемента $\beta\alpha^k$ есть $s_k$ при $k = 0, \ldots, d-1$. Так что каждое $s_k \in \mathbf{Z}$. Таким образом, $f'(\alpha)\beta = \sum_{j=0}^{d-1} s_j \beta_j$ принадлежит $\mathbf{Z}[\alpha]$. □

Мы используем лемму 6.2.3 следующим образом. Вместо того чтобы требовать наличия множества $\mathcal{S}$ взаимно простых целых чисел с произведением $\prod_{(a,b)\in\mathcal{S}}(a - b\alpha)$, равным квадрату в $\mathbf{Z}[\alpha]$, как мы сначала хотели, ограничимся тем, что это произведение равно квадрату в $\mathcal{I}$, скажем $\gamma^2$. Тогда по лемме 6.2.3 $f'(\alpha)\gamma \in \mathbf{Z}[\alpha]$, так что $f'(\alpha)^2 \prod_{(a,b)\in\mathcal{S}}(a - b\alpha)$ равно квадрату в $\mathbf{Z}[\alpha]$.

Первые три затруднения отличаются друг от друга, но есть тема, которая их объединяет, а именно, тема хорошо изученных групп. Затруднение (1) связано с группой $\mathcal{I}/\mathbf{Z}[\alpha]$. Затруднение (2) — с группой классов кольца $\mathcal{I}$. А затруднение (3) — с группой единиц кольца $\mathcal{I}$. Читатель, которого это шокирует, может обратиться к какому-либо руководству по алгебраической теории чисел, содержащему полное описание этих групп, но как мы дальше увидим, очень простой прием позволит нам преодолеть эти три затруднения. Более того, чтобы понять, как следует реализовывать решето числового поля, нужно лишь понять этот простой прием. Наш гипотетический шокированный читатель может лишь смело пропустить следующие несколько абзацев!

Что касается затруднения (1), хотя в разложении (на простые идеалы в $\mathcal{I}$) идеала $\left(\prod_{(a,b)\in\mathcal{S}}(a - b\alpha)\right)$ не все показатели степеней могут быть четными, простые идеалы с нечетными показателями все лежат поверх простых чисел, делящих индекс кольца $\mathbf{Z}[\alpha]$ в $\mathcal{I}$, так что число этих исключительных простых идеалов ограничено логарифмом (по основанию 2) от этого индекса.

Затруднение (2) более естественно описывается группой классов идеалов по модулю подгруппы квадратов классов идеалов. Она является 2-группой, ранг которой есть 2-ранг группы классов идеалов, он ограничен логарифмом (по основанию 2) от порядка группы классов, т. е. логарифмом от числа классов.

Затруднение (3) вновь более естественно описывается группой единиц по модулю подгруппы квадратов единиц. Это снова 2-группа, и ее ранг $\leqslant d$, степени многочлена $f(x)$. (Здесь мы используем знаменитую теорему Дирихле о единицах.)

Подробный анализ этих затруднений можно найти в работе [Buhler et al. 1993]. Нас будет устраивать вывод о том, что хотя затруднения (1), (2) и (3) отличаются друг от друга, все они незначительны. Существует метод грубой силы для преодоления этих трех затруднений, но существует также красивый и простой обходной путь. Идея этого обходного пути принадлежит Адлеману и работает следующим образом. Представьте себе на минуту, что Вы по какой-то причине не можете отличать положительные числа от отрицательных, но можете распознавать разложения на простые сомножители. Тогда и 4, и $-4$ будут выглядеть для Вас как квадраты, поскольку в их разложении на простые сомножители 2 возводится в четную степень, и больше никаких простых чисел нет. Однако, $-4$ — это не квадрат. Не пользуясь тем, что это число отрицательно, мы все равно сможем сказать, что $-4$ не есть квадрат, если заметим, что оно не есть квадрат по модулю 7. Мы можем обнаружить это с помощью символа Лежандра $\left(\frac{-4}{7}\right) = -1$. В более общем случае утверждение таково: если $q$ — нечетное простое число, и если $\left(\frac{m}{q}\right) = -1$, то $m$ не есть квадрат. Идея Адлемана состоит в использовании обратного утверждения, хоть оно и не является верной теоремой! Трюк в том, чтобы мыслить вероятностно. Пусть для заданного целого числа $m$ мы случайным образом выбрали $k$ различных нечетных простых чисел $q$ в интервале $q < |m|$. Пусть, далее, для каждого из этих $k$ тестовых простых чисел $q$ мы имеем $\left(\frac{m}{q}\right) = 1$. Если $m$ не есть квадрат, то вероятность наступления указанного события будет (эвристически) примерно $2^{-k}$. Итак, если данное событие *действительно наступает*, и $k$ велико (скажем, $k > \lg|m|$), то разумно считать, что $m$ в действительности является квадратом.

Нам хотелось бы применить эту идею к алгебраическим числам $a - b\alpha$, и следующий результат позволит нам сделать это при помощи обычных символов Лежандра.

**Лемма 6.2.4.** *Пусть $f(x)$ — неприводимый многочлен из $\mathbf{Z}[x]$ со старшим коэффициентом единица, и пусть $\alpha$ — корень многочлена $f$ в поле комплексных чисел. Положим $q$ равным нечетному простому числу, и $s$ — равным целому числу, такому, что $f(s) \equiv 0 \pmod q$ и $f'(s) \not\equiv 0 \pmod q$. Пусть $\mathcal{S}$ есть множество пар взаимно простых целых чисел $(a,b)$, таких, что $q$ не делит ни одно из чисел $a-bs$ при $(a,b) \in \mathcal{S}$, и $f'(\alpha)^2 \prod_{(a,b)\in\mathcal{S}}(a-b\alpha)$ является квадратом в $\mathbf{Z}[\alpha]$. Тогда*

$$\prod_{(a,b)\in\mathcal{S}} \left(\frac{a-bs}{q}\right) = 1. \tag{6.9}$$

Доказательство. Рассмотрим гомоморфизм $\phi_q$ из $\mathbf{Z}[\alpha]$ в $\mathbf{Z}_q$, где $\phi_q(\alpha)$ есть класс вычетов $s \pmod q$. Мы имеем, что $f'(\alpha)^2 \prod_{(a,b)\in\mathcal{S}}(a-b\alpha) = \gamma^2$ для некоторого $\gamma \in \mathbf{Z}[\alpha]$. По нашему предположению, $\phi_q(\gamma^2) \equiv f'(s)^2 \prod_{(a,b)\in\mathcal{S}}(a-bs) \not\equiv 0 \pmod q$. Тогда $\left(\frac{\phi_q(\gamma^2)}{q}\right) = \left(\frac{\phi_q(\gamma)^2}{q}\right) = 1$ и $\left(\frac{f'(s)^2}{q}\right) = 1$, так что

$$\left(\frac{\prod_{(a,b)\in\mathcal{S}}(a-bs)}{q}\right) = 1,$$

откуда следует, что выполнено соотношение (6.9). □

Итак, мы вновь получили необходимое условие равенства квадрату, в то время как мы по-прежнему ищем достаточное условие. Но мы уже почти у цели. Как мы уже отметили, можно эвристически обосновать, что если $k$ достаточно велико, и $q_1, \ldots, q_k$ — нечетные простые, которые не делят ни одно из чисел $N(a-b\alpha)$ при $(a,b) \in \mathcal{S}$, и если мы имеем $s_j \in R(q_j)$ при $j = 1, \ldots, k$, где $f'(s_j) \not\equiv 0 \pmod{q_j}$, то из соотношений

$$\sum_{(a,b)\in\mathcal{S}} \vec{v}(a-b\alpha) \equiv \vec{0} \pmod 2$$

и

$$\prod_{(a,b)\in\mathcal{S}} \left(\frac{a-bs_j}{q_j}\right) = 1 \text{ при } j = 1, \ldots, k$$

вытекает, что

$$\prod_{(a,b)\in\mathcal{S}} (a-b\alpha) = \gamma^2 \text{ для некоторого } \gamma \in \mathcal{I}.$$

Но что означает «достаточно велико»? Снова можно сказать, что поскольку масштабы затруднений (1), (2), (3) малы, $k$ здесь вовсе не обязано быть очень большим. Мы будем выбирать многочлен $f(x)$ так, чтобы его степень $d$ удовлетворяла неравенству $d^{2d^2} < n$ (где $n$ — число, которое мы раскладываем), и так, чтобы все коэффициенты $c_j$ многочлена $f$ удовлетворяли неравенству $|c_j| < n^{1/d}$. При таких условиях можно показать, что сумма мер первых трех затруднений меньше, чем $\lg n$ (см. [Buhler et al. 1993], теорема 6.7). Есть гипотеза, что достаточно выбрать $k = \lfloor 3 \lg n \rfloor$ (причем $k$ простых чисел $q_j$ выбираются наименьшими из возможных). Вероятно, подойдет и несколько меньшее значение $k$, но этот аспект не является узким местом с точки зрения времени работы алгоритма.

Мы используем пары $q_j, s_j$ для добавления к нашим векторам показателей $k$ дополнительных координат. Если $\left(\frac{a-bs_j}{q_j}\right) = 1$, то координата, соответствующая $q_j, s_j$ в исходном векторе показателей для $a-b\alpha$, равна 0. Если наш символ Лежандра есть $-1$, координата равна 1. (Это позволяет отобразить мультипликативную группу $\{1,-1\}$ порядка 2 в аддитивную группу $\mathbf{Z}_2$ порядка 2.) Данные удлиненные векторы показателей теперь оказываются не только необходимыми, но и достаточными (с практической точки зрения) для построения квадратов.

### 6.2.5. Базовый метод NFS: квадратные корни

Предположим теперь, что мы преодолели все затруднения предыдущего раздела и имеем множество $\mathcal{S}$ пар взаимно простых целых чисел, таких, что $f'(\alpha)^2 \prod_{(a,b)\in\mathcal{S}}(a-b\alpha) = \gamma^2$ для некоторого $\gamma \in \mathbf{Z}[\alpha]$, и $\prod_{(a,b)\in\mathcal{S}}(a-bm) = v^2$ для $v \in \mathbf{Z}$. Тогда мы почти у цели, так как если $u$ — целое число, такое, что $\phi(\gamma) \equiv u \pmod{n}$, то $u^2 \equiv (f'(m)v)^2 \pmod{n}$, и можно попытаться разложить $n$ с помощью НОД$(u - f'(m)v, n)$.

Однако остается одна проблема. Методы предыдущих разделов позволяют нам найти множество $\mathcal{S}$ с указанными свойствами, но они ничего не говорят нам о том, как быть с нахождением квадратных корней $\gamma$ и $v$. Иными словами, мы имеем квадраты, один из которых принадлежит $\mathbf{Z}[\alpha]$, а другой — $\mathbf{Z}$, и хотим найти соответствующие им квадратные корни.

Задача для $v$ проста и решается так же, как в методе QS. Исходя из наших векторов показателей, мы можем легко получить разложение числа $v^2$ на простые сомножители, а из него уже совсем легко получить разложение числа $v$. На самом деле нам не обязательно знать само число $v$, нам достаточно знать лишь его вычет по модулю $n$. Для каждой степени простого, делящей $v$, вычислим ее вычет по модулю $n$ с помощью быстрого алгоритма возведения в степень по модулю, например, 2.1.5. Затем перемножим эти вычеты в $\mathbf{Z}_n$, в итоге получим $v \pmod{n}$.

Более сложна и более интересна задача вычисления $\gamma$. Если $\gamma$ записано в виде $a_0 + a_1\alpha + \ldots + a_{d-1}\alpha^{d-1}$, то нужное нам значение $u$ есть $a_0 + a_1 m + \ldots + a_{d-1}m^{d-1}$. Поскольку нас вновь интересует лишь вычет $u \pmod{n}$, получается, что нас интересуют лишь вычеты $a_j \pmod{n}$. Это нам на руку, поскольку целые числа $a_0, \ldots, a_{d-1}$ могут быть весьма велики, и число цифр в них может быть порядка корня квадратного из числа шагов во всем остальном алгоритме! Не хотелось бы выполнять много арифметических действий над такими огромными числами. Даже если бы мы вычисляли лишь целое алгебраическое число $\gamma^2$ и не беспокоились бы о нахождении квадратного корня $\gamma$, нам пришлось бы использовать методы быстрого умножения гл. 9 для того, чтобы время работы не превысило оценку из разд. 6.2.3. И мы еще даже не начали говорить о том, как вычислять наш квадратный корень.

Если мы рассматриваем частный случай, когда $\mathbf{Z}[\alpha] = \mathcal{I}$, и данное кольцо является областью с однозначным разложением на множители, то мы можем использовать метод, аналогичный описанному выше для вычисления $v \pmod{n}$. Но в общем случае наше кольцо может сильно отличаться от области с однозначным разложением.

Один из методов, предложенный в работе [Buhler et al. 1993], начинает свою работу с нахождения простого числа $p$, такого, что $f(x)$ неприводим по модулю $p$. Затем мы находим $\gamma \pmod{p}$ (т. е. координаты $\gamma$ по модулю $p$). Для этого мы проводим вычисления в конечном поле $\mathbf{Z}_p[x]/(f(x))$ (см. разд. 2.2.2). Вычисление квадратного корня может проводиться по схеме алгоритма 2.3.8 (см. упр. 2.16). Это можно считать началом, поскольку мы фактически можем сравнительно легко найти вычеты $a_0 \pmod{p}, \ldots, a_{d-1} \pmod{p}$. Почему бы не

сделать этого для других простых $p$, а затем не склеить результат с помощью китайской теоремы об остатках? С этим общим подходом связана очевидная проблема. Для каждого простого числа $p$, с которым мы работаем, существует два квадратных корня, и мы не знаем, как выбирать знаки при склеивании. Мы могли бы проверять все возможные варианты, но если мы используем $k$ простых, лишь 2 из $2^k$ вариантов нам подойдут. Мы можем выбрать любое решение для одного из простых чисел $p$ и свести задачу к выбору из $2^{k-1}$ вариантов для остальных простых чисел, но это не совсем удобно, если $k$ велико.

Существует по меньшей мере два возможных способа решения проблемы выбора знаков. Метод, предложенный в работе [Buhler et al. 1993], не использует китайскую теорему об остатках с различными простыми числами, а использует поднятие решения по Гензелю для получения решений по модулю все более и более высоких степеней одного и того же простого числа $p$ (см. алгоритм 2.3.11). Когда степень $p$ превысит границу для коэффициентов $a_j$, это будет означать, что мы их нашли. Это проще, чем использование методов разложения многочленов на множители из работы [Lenstra 1983], но на последней стадии решения по Гензелю, когда мы работаем с наибольшими степенями простых чисел, мы производим действия с огромными целыми числами, и чтобы держать оценку сложности под контролем, нам необходимо использовать быстрые подпрограммы из гл. 9.

Другая стратегия, предложенная в работе [Couveignes 1993], использует китайскую теорему об остатках, но работает лишь в случае нечетного $d$. В этом случае норма числа $-1$ равна $-1$, так что мы можем с самого начала определиться, что ищем вариант значения $\gamma$ с положительной нормой. Поскольку по векторам показателей можно узнать разложение $N(\gamma)$ на простые сомножители, мы можем вычислить $N(\gamma) \pmod p$, где $p$, как и раньше, простое число, по модулю которого $f(x)$ неприводим. Вычислив $\gamma_p$, удовлетворяющее сравнению $\gamma_p^2 \equiv \gamma^2 \pmod p$, мы выбираем $\gamma_p$ или $-\gamma_p$ в зависимости от того, какое из них имеет норму, сравнимую с $N(\gamma) \pmod p$. Это обеспечивает корректный выбор знаков для каждого использованного $p$. Данную идею, по-видимому, нельзя обобщить на случай четных степеней $d$.

Оказывается, существует один эвристический подход к нахождению квадратных корней, который весьма успешно работает на практике и уменьшает вклад этой стадии алгоритма в общее время работы. Этот метод использует некоторые из идей, перечисленных выше, а также некоторые другие идеи. Подробности см. в работах [Montgomery 1994], [Nguyen 1998].

### 6.2.6. Базовый метод NFS: итоговый алгоритм

Сейчас мы подведем итог предыдущим разделам, дав обозримое описание метода NFS. Поскольку данный алгоритм довольно изощренный, в следующем листинге мы решили использовать побольше словесных описаний.

**Алгоритм 6.2.5** (решето числового поля). Нам задано нечетное составное число $n$, не являющееся степенью. Этот алгоритм пытается найти нетривиальное разложение числа $n$.

1. [Начальная установка]
   $d = \lfloor (3 \ln n / \ln\ln n)^{1/3} \rfloor$; // Это $d$ таково, что $d^{2d^2} < n$.
   $B = \lfloor \exp((8/9)^{1/3} (\ln n)^{1/3} (\ln\ln n)^{2/3}) \rfloor$;
   // Заметим, что $d$, $B$ можно при желании настроить
   // по своему усмотрению.
   $m = \lfloor n^{1/d} \rfloor$;
   Записать $n$ по основанию $m$: $n = m^d + c_{d-1} m^{d-1} + \ldots + c_0$;
   $f(x) = x^d + c_{d-1} x^{d-1} + \ldots + c_0$; // Установить многочлен $f$.
   Попытаться разложить $f(x)$ на неприводимые многочлены в $\mathbf{Z}[x]$ с использованием алгоритма разложения авторов работы [Lenstra et al. 1982] или, например, варианта из книги [Cohen 2000, p. 139];
   Если $f(x)$ допускает нетривиальное разложение $g(x)h(x)$, возвратить (также нетривиальное) разложение $n = g(m)h(m)$;
   $F(x,y) = x^d + c_{d-1} x^{d-1} y + \ldots + c_0 y^d$; // Установить многочлен $F$.
   $G(x,y) = x - my$;
   for(простое $p \leqslant B$) найти множество
   $$R(p) = \{r \in [0, p-1] : f(r) \equiv 0 \pmod p\};$$
   $k = \lfloor 3 \lg n \rfloor$;
   Вычислить $k$ первых простых чисел $q_1, \ldots, q_k > B$, таких, что $R(q_j)$ содержит некоторый элемент $s_j$, такой, что $f'(s_j) \not\equiv 0 \pmod{q_j}$, и сохранить при этом $k$ пар $(q_j, s_j)$;
   $B' = \sum_{p \leqslant B} \#R(p)$;
   $V = 1 + \pi(B) + B' + k$;
   $M = B$;
2. [Решето]
   Использовать решето для нахождения множества $\mathcal{S}'$ пар взаимно простых целых чисел $(a,b)$, таких, что $0 < |a|, b \leqslant M$, и произведение $F(a,b)G(a,b)$ является $B$-гладким, пока мы не получим $\#\mathcal{S}' > V$, а если это не удается, увеличить $M$ и попробовать еще раз, или перейти к шагу [Начальная установка] и там увеличить $B$;
3. [Матрица]
   // Построим двоичную матрицу размером $V \times V$,
   // где на каждую пару $(a,b)$ приходится одна строка.
   for($(a,b) \in \mathcal{S}'$) {
   // Вычислим $\vec{v}(a - b\alpha)$, двоичный вектор показателей для числа $a - b\alpha$, имеющий $V$ битов (координат), следующим образом:
   Установить первый бит вектора $\vec{v}$ в 1, если $G(a,b) < 0$, иначе установить этот бит в 0;
   // Следующие $\pi(B)$ битов зависят от простых чисел $p \leqslant B$: определим $p^\gamma$ как степень числа $p$ в разложении $|G(a,b)|$ на простые сомножители.
   Установить бит для $p$ в 1, если $\gamma$ нечетно, иначе установить этот бит в 0;

// Следующие $B'$ битов будут соответствовать парам $p, r$, где $p$ — простое, не превосходящее $B$, и $r \in R(p)$. Мы будем использовать обозначение $v_{p,r}(a - b\alpha)$, введенное перед леммой 6.2.1.

Установить бит для пары $p, r$ в 1, если $v_{p,r}(a - b\alpha)$ нечетно, иначе установить его в 0;

// Наконец, последние $k$ битов соответствуют парам $q_j, s_j$.

Установить бит для пары $q_j, s_j$ в 1, если $\left(\frac{a-bs_j}{q_j}\right)$ есть $-1$, иначе установить его в 0;

Добавить вектор показателей $\vec{v}(a - b\alpha)$ в качестве очередной строки нашей матрицы;

}

4. [Линейная алгебра]

При помощи какого-либо из методов линейной алгебры (см. разд. 6.1.3) найти непустое подмножество $\mathcal{S}$ множества $\mathcal{S}'$, такое, что $\sum_{(a,b)\in\mathcal{S}} \vec{v}(a - b\alpha)$ есть нулевой вектор $(\mathrm{mod}\ 2)$;

5. [Квадратные корни]

Использовать известное разложение квадрата $\prod_{(a,b)\in\mathcal{S}}(a - bm)$ на простые сомножители для нахождения вычета $v \bmod n$, такого, что $\prod_{(a,b)\in\mathcal{S}}(a - bm) \equiv v^2 \pmod{n}$;

Используя метод наподобие тех, которые представлены в разд. 6.2.5, найти квадратный корень $\gamma$, принадлежащий $\mathbf{Z}[\alpha]$, из числа $f'(\alpha)^2 \prod_{(a,b)\in\mathcal{S}}(a - b\alpha)$, и путем простой подстановки $\alpha \rightarrow m$ вычислить $u = \phi(\gamma) \pmod{n}$;

6. [Разложение]

return НОД$(u - f'(m)v, n)$;

Если алгоритм 6.2.5 выдает тривиальный делитель числа $n$, то у нас есть возможность найти новые линейные зависимости в нашей матрице и попытаться еще раз. Если новых линейных зависимостей нет, у нас есть возможность просеивать дальше, чтобы найти новые строки для нашей матрицы и тем самым новые линейные зависимости.

## 6.2.7. NFS: дальнейшие соображения

Как и в случае базового метода квадратичного решета, существуют разнообразные «приспособления», которыми можно дополнить решето числового поля, чтобы сделать этот метод разложения еще лучше. В этом разделе мы кратко обсудим некоторые из этих улучшений.

### Свободные соотношения

Пусть $p$ есть простое число из «фактор-базы», т. е. $p \leqslant B$. Наши векторы показателей имеют координату, соответствующую $p$, как возможному простому делителю числа $a - bm$, а также $\#R(p)$ координат, соответствующих целым числам $r \in R(p)$. (Вспомним, что $R(p)$ есть множество вычетов $r \pmod{p}$, таких, что $f(r) \equiv 0 \pmod{p}$.) В среднем $\#R(p)$ есть 1, но его нижняя граница — это 0

(в случае, когда $f(x)$ не имеет корней $(\mathrm{mod}\ p)$), а верхняя — это $d$, степень многочлена $f(x)$ (в случае, когда $f(x)$ разлагается на $d$ различных сомножителей $(\mathrm{mod}\ p)$). В последнем случае мы имеем, что произведение простых идеалов $(p, \alpha - r)$ в полном кольце целых алгебраических чисел из $\mathbf{Q}[\alpha]$ есть $(p)$.

Пусть $p$ есть простое число, $p \leqslant B$, и $R(p)$ состоит из $d$ элементов. Поместим в нашу матрицу дополнительный вектор-строку $\vec{v}(p)$, который содержит 1 на тех местах, которые соответствуют числу $p$ и каждой паре $p, r$, где $r \in R(p)$. Кроме того, в последней группе из $k$ координат, отвечающих квадратичным характерам по модулю $q_j$, где $j = 1, \ldots, k$, запишем 0 в координату для номера $j$ вектора $\vec{v}(p)$, если $\left(\frac{p}{q_j}\right) = 1$, и запишем 1 в координату для номера $j$, если $\left(\frac{p}{q_j}\right) = -1$. Такой вектор $\vec{v}(p)$ называется свободным соотношением, поскольку он найден на этапе предварительных вычислений, а не на стадии просеивания. Если сейчас мы найдем подмножество строк, которые в сумме дают нулевой вектор по модулю 2, то этому подмножеству будут соответствовать множество $\mathcal{S}$ взаимно простых пар $a, b$ и множество $\mathcal{F}$ свободных соотношений. Пусть $w$ есть произведение простых чисел $p$, отвечающих свободным соотношениям в $\mathcal{F}$. Тогда должно быть

$$wf'(\alpha)^2 \prod_{(a,b)\in\mathcal{S}} (a - b\alpha) = \gamma^2 \text{ для некоторого } \gamma \in \mathbf{Z}[\alpha],$$

$$wf'(m)^2 \prod_{(a,b)\in\mathcal{S}} (a - bm) = v^2 \text{ для некоторого } v \in \mathbf{Z}.$$

Тогда если $\phi(\gamma) = u$, то мы имеем $u^2 \equiv v^2 \ (\mathrm{mod}\ n)$, как и раньше.

Преимущество использования свободных соотношений в том, что чем больше их у нас есть, тем меньше соотношений нужно будет искать на более медленной стадии решета. Кроме того, векторы $\vec{v}(p)$ более разреженные по сравнению с типичным вектором показателей $\vec{v}(a, b)$, так что включение в матрицу свободных соотношений позволяет матричной стадии выполняться быстрее.

Так каково же ожидаемое количество свободных соотношений, которые мы найдем? Свободное соотношение соответствует простому числу $p$, которое полностью разлагается в поле алгебраических чисел $\mathbf{Q}(\alpha)$. Пусть $g$ есть порядок поля разложения многочлена $f(x)$, т. е. замыкания Галуа поля $\mathbf{Q}(\alpha)$ в поле комплексных чисел. Из теоремы Чеботарева о плотности следует, что количество простых чисел $p$ до некоторой границы $X$, которые полностью разлагаются в $\mathbf{Q}(\alpha)$, асимптотически равно $\frac{1}{g}\pi(X)$ при $X \to \infty$. Т. е. в среднем одно из каждых $g$ простых чисел соответствует свободному соотношению. Предполагая, что граница $B$ нашей фактор-базы достаточна для того, чтобы можно было воспользоваться указанной асимптотикой (это *еще одно* эвристическое, но разумное, предположение), мы, таким образом, можем ожидать порядка $\frac{1}{g}\pi(B)$ свободных соотношений. Далее, порядок $g$ поля разложения имеет нижнюю границу, равную $d$, степени многочлена $f(x)$, а верхнюю — $d!$. Понятно, что чем меньше $g$, тем больше будет свободных соотношений. К несчастью, в общей ситуации $g = d!$. Иными словами, для большинства неприводимых многочленов $f(x)$ из $\mathbf{Z}[x]$ степени $d$ порядок поля разложения многочлена $f(x)$ равен $d!$. Так,

например, если $d = 5$, мы можем ожидать лишь примерно $\frac{1}{120}\pi(B)$ свободных соотношений, если мы выберем наш многочлен $f(x)$ в соответствии со схемой, реализованной на шаге [Начальная установка] алгоритма 6.2.5. Поскольку наши векторы имеют порядка $2\pi(B)$ координат, то при использовании свободных соотношений время просеивания сократилось бы меньше, чем на полпроцента. Однако эта небольшая помощь достается нам, по-существу, даром.

Свободные соотношения могут помочь значительно больше в случае специальных многочленов $f(x)$ с небольшими полями разложения. Например, при разложении 9-го числа Ферма $F_9$ был использован многочлен $f(x) = x^5 + 8$. Порядок поля разложения здесь равен 20, так что свободные соотношения позволили уменьшить время просеивания примерно на 2.5%.

**Частичные соотношения**

Как и в методе квадратичного решета, просеивание в решете числового поля обнаруживает не только пары $a, b$, для которых каждое из чисел $N(a - b\alpha) = F(a, b) = b^d f(a/b)$ и $a - bm$ являются $B$-гладкими, но и пары $a, b$, для которых одно или оба этих числа равны $B$-гладким числам, умноженным на некоторое большее простое число. Если мы допускаем соотношения, которые включают такие большие простые числа, не более одного на каждое из $N(a - b\alpha)$ и $a - bm$, то мы имеем структуру данных, напоминающую квадратичное решето с двойной вариацией больших простых чисел (см. разд. 6.1.4). Также было предложено использовать данные, в которых $N(a-b\alpha)$ имеет два больших простых делителя, а $a - bm$ является $B$-гладким, и наоборот. А некоторые даже рассматривают использование данных, в которых оба указанных числа имеют до двух больших простых делителей. Интересно было бы выяснить, не будет ли в этом случае более легким и эффективным решением простое увеличение границы $B$.

**Многочлены с неединичным старшим коэффициентом**

В алгоритме 6.2.5 сказано, что многочлен $f(x)$ выбирается на этапе [Начальная установка] специальным образом, в результате чего старший коэффициент $f$ оказывается равным 1. При изложении предыдущих разделов мы действительно опирались на этот факт. В этом случае ($\alpha$ здесь есть корень $f(x)$) кольцо $\mathbf{Z}[\alpha]$ есть подкольцо кольца целых алгебраических чисел в $\mathbf{Q}(\alpha)$. В действительности у нас есть больше свободы в выборе $f(x)$, чем утверждалось ранее. Необходимо лишь, чтобы $f(x) \in \mathbf{Z}[x]$ был неприводимым. Не обязательно выбирать $f$ специальным образом, как это сделано на этапе [Начальная установка], равно как не обязательно, чтобы $f$ имел старший коэффициент 1. Простые, делящие старший коэффициент многочлена $f(x)$, будут считаться подозрительными в наших векторах показателей. Но мы уже сталкивались с похожей ситуацией, поскольку те простые, которые делят дискриминант многочлена $f(x)$, при рассмотрении случая единичного старшего коэффициента также были подозрительными и частично обусловили необходимость квадратичных характеров на шаге [Матрица] в алгоритме 6.2.5 (они обсуждались в разд. 6.2.4). Нам будет достаточно знать, что многочлены с неединичным старшим коэффициентом не привносят никаких новых существенных трудностей.

Но зачем нам потребовались эти многочлены? Как мы выяснили в разд. 6.2.3, ключом к более быстрому алгоритму является уменьшение размера чисел, которые мы просеиваем в надежде отыскать среди них гладкие. Размер этих чисел в NFS напрямую зависит от размера числа $m$ и коэффициентов многочлена $f(x)$ при заданной степени $d$. Выбирая многочлен с единичным старшим коэффициентом, мы можем ограничить число $m$ и его коэффициенты величиной $n^{1/d}$. Если же мы допускаем многочлены с неединичным коэффициентом, мы можем выбрать $m$ вида $\lceil n^{1/(d+1)} \rceil$. Записав $n$ в системе счисления с основанием $m$, будем иметь $n = c_d m^d + c_{d-1} m^{d-1} + \ldots + c_0$. Эта запись показывает, что мы можем использовать многочлен $f(x) = c_d x^d + c_{d-1} x^{d-1} + \ldots + c_0$. Его коэффициенты $c_i$ ограничены величиной $n^{1/(d+1)}$, так что и $m$, и эти коэффициенты делятся на величину порядка $n^{1/(d^2+d)}$.

Когда величина раскладываемых чисел стремится к бесконечности, это уменьшение коэффициентов не очень существенно: эвристическая сложность NFS остается прежней. (Асимптотическое ускорение составляет величину $\ln^{1/6} n$.) Мы, однако, не раскладываем бесконечно большие числа, а для тех чисел, которые мы раскладываем, эта экономия существенна.

Предположим, что многочлен $f(x) = c_d x^d + c_{d-1} x^{d-1} + \ldots + c_0$ неприводим в $\mathbf{Z}[x]$, и $\alpha \in \mathbf{C}$ — один из его корней. Тогда $c_d \alpha$ — целое алгебраическое число. Оно является корнем $F(x) = x^d + c_{d-1} x^{d-1} + c_d c_{d-2} x^{d-2} + \ldots + c_d^{d-1} c_0$, что очевидно, поскольку $F(c_d x) = c_d^{d-1} f(x)$. Мы заключаем, что если $\mathcal{S}$ есть множество пар $a, b$ взаимно простых целых чисел, если

$$\prod_{(a,b)\in\mathcal{S}} (a - b\alpha)$$

есть квадрат в $\mathbf{Q}(\alpha)$, и если $\mathcal{S}$ состоит из *четного числа* пар, то

$$F'(c_d\alpha)^2 \prod_{(a,b)\in\mathcal{S}} (ac_d - bc_d\alpha)$$

есть квадрат в $\mathbf{Z}[c_d\alpha]$, скажем, $\gamma^2$. Нахождение целочисленных коэффициентов (по модулю $n$) числа $\gamma$ относительно базиса $1, c_d\alpha, \ldots, (c_d\alpha)^{d-1}$ позволяет, как и раньше, получить два сравнимых квадрата по модулю $n$ и тем самым дает возможность разложить число $n$. (Заметим, что если $F(x,y) = y^d f(x/y)$ есть однородная форма многочлена $f(x)$, то $F(c_d x, c_d) = c_d F(c_d x)$, так что $F_x(c_d\alpha, c_d) = c_d F'(c_d\alpha)$. Поэтому в предыдущих рассуждениях мы можем использовать $F_x(c_d\alpha, c_d)$ вместо $F'(c_d\alpha)$, если захотим.) Итак, использование многочленов с неединичным старшим коэффициентом не представляет большой сложности. Чтобы обеспечить четное число элементов в множестве $\mathcal{S}$, можно дополнить все наши векторы показателей одной координатой, которую сделаем всегда равной 1.

В приведенных выше рассуждениях предполагается, что коэффициент $c_d$ взаимно прост с $n$. Это условие проверить нетрудно. И поскольку $c_d$ меньше $n$ во всех рассматриваемых случаях, нетривиальное значение НОД привело бы к нетривиальному разложению $n$. За дальнейшими подробностями относительно использования многочленов с неединичным старшим коэффициентом, а также

относительно использования однородных многочленов, мы отсылаем читателя к работе [Buhler et al. 1993, Section 12].

В вопросе выбора полиномов имеются определенные достижения, сыгравшие важную роль в нахождении рекордного разложения знаменитого 155-значного числа RSA в конце 1999 г. Оказывается, удачный многочлен может настолько изменить ситуацию, что имеет смысл потратить значительные ресурсы на поиск подходящих многочленов. Подробности о последних стратегиях см. в работах [Murphy 1998, 1999].

### Полиномиальные пары

В методе NFS, описанном в предыдущих разделах, фактически присутствуют *два* многочлена, хотя мы и делаем акцент лишь на одном многочлене $f(x)$, для которого имеем целое число $m$ с условием $f(m) \equiv 0 \pmod{n}$. Более точно, мы рассматриваем однородную форму многочлена $f$, а именно $F(x,y) = y^d f(x/y)$, где $d$ есть степень $f(x)$. Второй многочлен более тривиален: $g(x) = x - m$. Его однородная форма $G(x,y) = yg(x/y) = x - my$. Числа, которые мы просеиваем в поиске гладких значений, есть значения многочлена $F(x,y)G(x,y)$ в прямоугольнике в окрестности начала координат.

Однако степень многочлена $g(x)$ не обязана быть равной 1. Пусть у нас есть два различных неприводимых (не обязательно с единичным старшим коэффициентом) многочлена $f(x), g(x) \in \mathbf{Z}[x]$ и целое число $m$, такое, что $f(m) \equiv g(m) \equiv 0 \pmod{n}$. Пусть $\alpha$ есть корень $f(x)$ в $\mathbf{C}$, а $\beta$ есть корень $g(x)$ в $\mathbf{C}$. Предполагая, что старший коэффициент $c$ многочлена $f(x)$ и старший коэффициент $C$ многочлена $g(x)$ взаимно просты с $n$, мы имеем гомоморфизмы $\phi : \mathbf{Z}[c\alpha] \to \mathbf{Z}_n$ и $\psi : \mathbf{Z}[C\beta] \to \mathbf{Z}_n$, где $\phi(c\alpha) \equiv cm \pmod{n}$ и $\psi(C\beta) \equiv Cm \pmod{n}$.

Допустим также, что мы имеем множество $\mathcal{S}$, состоящее из четного числа пар $a, b$ взаимно простых целых чисел, и такие элементы $\gamma \in \mathbf{Z}[\alpha]$ и $\delta \in \mathbf{Z}[\beta]$, что

$$F_x(c\alpha, c)^2 \prod_{(a,b)\in\mathcal{S}} (ac - bc\alpha) = \gamma^2, \qquad G_x(C\beta, C)^2 \prod_{(a,b)\in\mathcal{S}} (aC - bC\beta) = \delta^2.$$

Если $\mathcal{S}$ состоит из $2k$ элементов, и $\phi(\gamma) \equiv v \pmod{n}$, $\psi(\delta) \equiv w \pmod{n}$, то

$$\left(C^k G_x(Cm, C)v\right)^2 \equiv \left(c^k F_x(cm, c)w\right)^2 \pmod{n},$$

так что мы можем попытаться разложить $n$, вычислив НОД$(C^k G_x(Cm, C)u - c^k F_x(cm, c)v, n)$.

Может возникнуть вопрос, почему выгодно использовать два многочлена степени, большей 1. В ответе есть определенная тонкость. Хотя среди свойств чисел, просеиваемых на предмет гладких значений, для нас важнейшим является их величина, существует и свойство, стоящее на втором месте, которое также имеет определенное значение. Если нам дано, что число в окрестности $x$ является произведением двух чисел в окрестности $x^{1/2}$, то оно с большей вероятностью является гладким, чем случайное число в окрестности $x$, которое не обязательно является таким произведением. Если мы интересуемся

$y$-гладкостью, и $u = \ln x / \ln y$, то этот эффект второго порядка можно измерить величиной $2^u$. Это означает, что число в окрестности $x$, заданное в виде произведения двух случайных чисел в окрестности $x^{1/2}$, является $y$-гладким с вероятностью примерно в $2^u$ раз большей, чем случайное число в окрестности $x$. Если в решете числового поля у нас имеются два многочлена одинаковой степени с коэффициентами одного порядка, то и соответствующие им однородные формы имеют значения одинакового порядка. Именно произведение этих однородных форм мы просеиваем на предмет гладких значений, так что такая $2^u$-философия кажется уместной.

Однако в «обыкновенном» методе NFS, в том виде как он описан в алгоритме 6.2.5, мы точно так же исследуем на гладкость произведение двух чисел: одно есть значение однородной формы $F(a,b)$, а другое — значение линейной формы $a-bm$. Они не являются величинами одного порядка. В действительности, если использовать предложенные параметры, $F(a,b)$ имеет порядок произведения в степени $3/4$, а число $a-bm$ — порядок произведения в степени $1/4$. Такие числа также имеют повышенную вероятность $y$-гладкости, а именно в $\left(4/3^{3/4}\right)^u$ раз бóльшую.

Итак, используя два многочлена одинаковой степени $d \approx \frac{1}{2}(3\ln n/\ln\ln n)^{1/3}$ с коэффициентами, ограниченными величиной порядка $n^{1/2d}$, мы увеличиваем вероятность гладкости по сравнению с теми параметрами, которые выбраны в алгоритме 6.2.5, примерно в $\left(3^{3/4}/2\right)^u$ раза. У нас $u$ — величина порядка $2(3\ln n/\ln\ln n)^{1/3}$, так что использование этих двух многочленов степени $d$ позволяет сэкономить множитель порядка $(1.46)^{(\ln n/\ln\ln n)^{1/3}}$. Хотя это и не меняет нашу базовую оценку сложности, данное ускорение представляет собой серьезную экономию.

Однако при использовании парных многочленов встает проблема их нахождения. Пока никто не предложил удовлетворительного метода нахождения таких многочленов, если не считать полного перебора, возможно, дополненного быстрыми методами, основанными на решетках. Возьмем, к примеру, случай $d = 3$. Нам не известен хороший метод, позволяющий по заданному большому числу $n$ найти два различных неприводимых многочлена третьей степени $f(x), g(x)$ с коэффициентами, ограниченными, скажем, величиной $n^{1/6}$, и такое целое число $m$, возможно, очень большое, что $f(m) \equiv g(m) \equiv 0 \pmod n$. Мощностные соображения подсказывают, что такие многочлены должны существовать даже при меньших коэффициентах, скажем, ограниченных величиной порядка $n^{1/8}$.

### Специальное решето числового поля (SNFS)

Мощностные соображения показывают, что для большинства чисел $n$ при нахождении многочленов мы не можем предложить ничего существенно лучшего, чем прямолинейная стратегия алгоритма 6.2.5. Тем не менее, имеется большая совокупность чисел, для которых намного лучшие многочлены действительно существуют, и если бы мы могли найти такие многочлены, сложность метода NFS была бы существенно снижена. Термин специальное решето числового

поля (SNFS) относится к вариантам NFS, в которых нам удается отыскать чрезвычайно хорошие многочлены.

Метод SNFS использовался, главным образом, для разложения большой совокупности чисел Каннингема (они представляют собой числа вида $b^k \pm 1$ при $b = 2, 3, 5, 6, 7, 10, 11, 12$, см. [Brillhart et al. 1988]). Мы уже упоминали разложение 9-го числа Ферма $F_9 = 2^{512} + 1$ Ленстрой и др. (см. [Lenstra et al. 1993a]). Они использовали многочлен $f(x) = x^5 + 8$ и целое число $m = 2^{103}$, так что $f(m) = 8F_9 \equiv 0 \pmod{F_9}$. Хотя нам уже был известен делитель 2424833 числа $F_9$ (найденный Вестерном в 1903 г.), он был проигнорирован. Т. е. была использована удобная природа самого числа $F_9$; число $F_9/2424833$ уже не такое удобное!

Многочлен делает необычайно хорошим то, что он имеет очень малые коэффициенты. Если нам дано число $n = b^k \pm 1$, то мы можем построить многочлен следующим образом. Скажем, мы хотим, чтобы степень $f(x)$ была равной 5. Запишем $k = 5l + r$, где $r$ есть остаток от деления $k$ на 5. Тогда $b^{5-r}n = b^{5(l+1)} \pm b^{5-r}$. Следовательно, мы можем использовать многочлен $f(x) = x^5 \pm b^{5-r}$ и выбрать $m = b^{l+1}$. При больших $k$ коэффициенты многочлена $f(x)$ очень малы по сравнению с $n$.

Небольшим преимуществом многочлена вида $x^d + c$ является то, что порядок соответствующей группы Галуа является делителем числа $d\varphi(d)$, в отличие от обычного для многочленов степени $d$ значения $d!$. Вспомним, что выгода от использования свободных соотношений обратно пропорциональна порядку группы Галуа. Таким образом, свободные соотношения более выгодны в случае специальных многочленов вида $x^d + c$, нежели в общем случае.

Иногда при выборе специальных многочленов приходится демонстрировать чудеса изобретательности. Возьмем, к примеру, число $10^{193} - 1$, разложенное в 1996 г. М. Элькенбрахтом-Хюйцингом и П. Монтгомери. Они могли бы использовать многочлен $x^5 - 100$ и $m = 10^{39}$, как предлагалось в приведенных выше рассуждениях, или, возможно, $10x^6 - 1$ и $m = 10^{32}$. Как бы то ни было, разложение все равно казалось бы невыполнимой задачей. Число $10^{193} - 1$ уже было частично разложено. У него существует очевидный делитель 9, но нам также были известны делители

$$773,\ 39373,\ 561470969,\ 639701219449517,\ 4274417556076113498947,$$

$$26409540111952717487908689681403.$$

После деления $10^{193} - 1$ на эти известные числа получившееся число $n$ вновь оказалось составным и содержало 108 цифр. Можно было бы подумать об использовании квадратичного решета или общего метода NFS для разложения $n$, но было бы непростительной ошибкой не воспользоваться простым происхождением числа $n$. А именно, мы знаем, что 10 имеет малый мультипликативный порядок по модулю $n$. Это приводит нас к сравнению $\left(10^{64}\right)^3 \equiv 10^{-1} \pmod{n}$ и к сравнению $\left(6 \cdot 10^{64}\right)^3 \equiv 6^3 \cdot 10^{-1} \equiv 108 \cdot 5^{-1} \pmod{n}$. Следовательно, для многочлена $f(x) = 5x^3 - 108$ и $m = 6 \cdot 10^{64}$ мы имеем $f(m) \equiv 0 \pmod{n}$. Однако $m$ слишком велико для того, чтобы выгодно использовать линейный многочлен

$x - m$. Вместо этого Элькенбрахт-Хюйцинг и Монтгомери искали квадратичный многочлен $g(x)$ со сравнительно малыми коэффициентами и такой, что $g(m) \equiv 0 \pmod{n}$. Это было сделано с помощью рассмотрения решетки целочисленных троек $(A, B, C)$, таких, что $Am^2 + Bm + C \equiv 0 \pmod{n}$. Задача состоит в нахождении короткого вектора в этой решетке. Используя методы нахождения таких коротких векторов, они вскоре обнаружили вариант выбора $A, B, C$, в котором каждое из чисел содержит не более 36 цифр. Затем они использовали $f(x)$ вместе с $g(x) = Ax^2 + Bx + C$ для получения полного разложения $n$ и нашли, что $n$ есть произведение двух простых, наименьшее из которых равно

$$447798287131284928051408304965265782892174953181087929.$$

### Многие многочлены

Нетрудно догадаться о возможности использования в методе NFS многих многочленов. Например, выберем степень $d$, положим $m = \lceil n^{1/(d+1)} \rceil$, запишем $n$ в системе с основанием $m$, получив $n = c_d n^d + \ldots + c_0$, положим $f(x) = c_d x^d + \ldots + c_0$ и $f_j(x) = f(x) + jx - mj$ для различных малых целых чисел $j$. Или можно рассмотреть семейство $f_{j,k}(x) = f(x) + kx^2 - (mk - j)x - mj$ для различных малых целых чисел $k, j$. Значение каждого из этих многочленов, вычисленное в точке $m$, равно $n$.

Можно использовать такое семейство для поиска наиболее подходящего многочлена, например такого, который склонен иметь кратные корни для большого количества малых простых чисел. Такой многочлен может обладать однородной формой, значения которой являются гладкими чаще, чем в случае многочлена, для которого малые простые не имеют такой тенденции.

Но могут ли все эти многочлены использоваться одновременно? Этому существует очевидная помеха. Каждый раз, когда вводится новый многочлен, фактор-база должна быть расширена, для того чтобы учесть варианты расщепления простых для этого многочлена. Таким образом, каждый используемый многочлен должен иметь свое собственное поле координат в векторах показателей; тем самым введение большего числа многочленов удлиняет векторы.

В работе [Coppersmith 1993] найден способ (теоретически) преодолеть эту трудность. Автор использует большую фактор-базу для линейной формы $a - bm$ и малые фактор-базы для различных используемых многочленов. Более точно, если простые числа до $B$ используются для линейной формы, и используются $k$ многочленов, то мы используем лишь простые до $B/k$ для каждого из этих многочленов. Далее, мы рассматриваем лишь пары $a, b$, для которых $a - bm$ является $B$-гладким, а однородная форма одного из многочленов является $(B/k)$-гладкой. После того как будет найдено $B$ соотношений, мы (вероятнее всего) будем иметь более чем достаточное количество информации для составления сравнимых квадратов.

Копперсмит предложил сначала просеивать линейную форму $a - bm$ на предмет $B$-гладких чисел, а затем по отдельности проверять однородную форму каждого используемого многочлена, с тем чтобы выяснить, является ли ее

значение в точке $a, b$ $B/k$-гладким. Данная проверка может быть быстро осуществлена с использованием метода эллиптических кривых (см. разд. 7.4). Метод эллиптических кривых (ECM) в качестве теста на гладкость на практике не является столь же эффективным, как просеивание. Однако если бы мы захотели использовать ECM в методе QS или NFS вместо просеивания, итоговая эвристическая сложность осталась бы неизменной, с разницей лишь в выражении $o(1)$. В своем варианте NFS Копперсмит не смог эффективно использовать просеивание для проверки своих однородных многочленов на гладкость, поскольку пары $a, b$, которые он проверял, расположены нерегулярно, это те пары, в которых число $a - bm$ прошло тест на гладкость. (В действительности мы могли бы просеивать по переменной $j$ в семействе $f_j(x)$, предложенном выше, но здесь мы не получили бы достаточно длинный массив для того, чтобы сделать решето экономичным.) Так или иначе, использование ECM в качестве теста на гладкость позволяет использовать те же оценки сложности, которые использовались бы в случае просеивания.

Предполагая, что всего порядка $B^2$ пар $a, b$ подставляется в линейную форму $a - bm$, получим в результате $B^2k$ пар, составленных из линейной формы и норменной формы многочлена, проверяемых на одновременную гладкость (первая проверяется на $B$-гладкость, вторая — на $B/k$-гладкость). Если параметры подобраны так, что после первого решета остается не более $B^2/k$ пар $a, b$, то общие затраты времени не намного превосходят $B^2$. Эта экономия ведет к понижению сложности NFS. Копперсмит приводит эвристическое рассуждение, показывающее, что при оптимальном выборе параметров время работы по разложению числа $n$ составляет $\exp\left((c + o(1))(\ln n)^{1/3}(\ln \ln n)^{2/3}\right)$, где

$$c = \frac{1}{3}\left(92 + 26\sqrt{13}\right)^{1/3} \approx 1.9019.$$

Это значение можно сравнить со значением $c = (64/9)^{1/3} \approx 1.9230$ для метода NFS в том виде, как он описан в алгоритме 6.2.5. Как упоминалось выше, к меньшему значению $c$ в методе Копперсмита добавляется более «весомое» $o(1)$. Этот побочный фактор, по всей видимости, смещает точку перевала, начиная с которой вариант Копперсмита становится наилучшим, в область порядка тысячи цифр. К тому времени как мы достигнем этой точки, метод NFS будет, вероятно, вытеснен много лучшими методами. Тем не менее, вариант метода NFS, принадлежащий Копперсмиту, в настоящий момент является асимптотически самым быстрым из известных эвристических методов разложения на множители. Могут существовать и некоторые практические предпосылки для использования многих многочленов. Обсуждение их см. в работе [Elkenbracht-Huizing 1997].

## 6.3. Строгое разложение на множители

Ни один из методов разложения, обсуждавшихся до сих пор в этой главе, не является строго обоснованным. С другой стороны, субэкспоненциальный

метод ECM, обсуждаемый в следующей главе, близок к строгому обоснованию. Приняв разумную гипотезу о распределении гладких чисел в коротких интервалах, Ленстра в работе [Lenstra 1987] показал, что ожидаемое число арифметических операций с целыми числами порядка $n$, необходимых для нахождения наименьшего делителя $p$ составного числа $n$ методом ECM, равно $\exp((2+o(1))\sqrt{\ln p \ln\ln p})$, где член «$o(1)$» стремится к 0 при $p \to \infty$. Таким образом, в ECM есть лишь один эвристический «пробел». Напротив, QS и NFS имеют несколько эвристических «пробелов» в своем обосновании. Интересно выяснить, каков самый быстрый алгоритм разложения из тех, которые мы можем строго обосновать. Это не обязательно будет иметь практическую ценность, но мы должны сделать это хотя бы из уважения к предмету!

Первый вопрос, к которому можно обратиться, — является алгоритм разложения детерминированным или вероятностным. Поскольку случайность — очень сильный фактор, мы можем ожидать меньшую рекордную сложность скорее от вероятностных алгоритмов разложения, чем от детерминированных. Так оно и есть. Самым быстрым детерминированным алгоритмом разложения, который строго обоснован, является метод Полларда—Штрассена. Он использует методы быстрого вычисления значений многочлена, обсуждаемые в разд. 5.5, и время его работы по разложению числа $n$ оказывается равным $O\left(n^{1/4+o(1)}\right)$.

В предположении расширенной гипотезы Римана (ERH) (см. гипотезу 1.4.2) алгоритм Шенкса детерминированно раскладывает $n$ на множители с оценкой времени работы $O(n^{1/5+o(1)})$. Этот метод обсуждается в разд. 5.6.4.

Это все, что можно сказать о строгих, детерминированных методах. А как насчет вероятностных методов? Первым субэкспоненциальным вероятностным алгоритмом разложения на множители с полностью строгим обоснованием был «метод случайных квадратов» Диксона (см. [Dixon 1981]). В его алгоритме выбираются случайные целые числа $r$ из промежутка $[1, n]$ и среди них ищутся те, для которых вычет $r^2 \bmod n$ является гладким. Если их найдено достаточное количество, то можно составить сравнимые между собой квадраты, как это делается в методе QS, и, пользуясь этим, попытаться разложить число $n$. Случайность используемых чисел $r$ позволяет точно сказать, как часто вычеты $r^2 \bmod n$ являются гладкими, и с какой вероятностью составленные сравнимые квадраты ведут к нетривиальному разложению числа $n$. Диксон показал, что ожидаемое время работы его алгоритма разложения $n$ ограничено величиной $\exp\left((c+o(1))\sqrt{\ln n \ln\ln n}\right)$, где $c = \sqrt{8}$. Последние улучшения Померанса и затем Валле снизили значение $c$ до $\sqrt{4/3}$.

В настоящее время наилучшей оценкой времени работы строго обоснованного вероятностного алгоритма разложения на множители является величина $\exp((1+o(1))\sqrt{\ln n \ln\ln n})$. Ее достиг «метод соотношений группы классов» из работы [Lenstra and Pomerance 1992]. Ранее этой временной границы достиг А. Ленстра с очень похожим алгоритмом, но его обоснование требовало использования ERH. Интересно, что эта граница совпадает с эвристической границей, достигнутой методом QS. Вновь все зло — в символе «$o(1)$», и практическое

сравнение этих методов оказывается не в пользу метода соотношений группы классов.

Интересно, что как улучшенные версии метода случайных квадратов, так и метод соотношений группы классов используют ECM в качестве подпрограммы для быстрого распознавания гладких чисел. Может вызвать удивление, как алгоритм, еще не имеющий строгого обоснования, может использоваться в качестве подпрограммы в строгом алгоритме. Ответ состоит в том, что нет необходимости показывать, что подпрограмма работает всегда, нужно лишь показать, что частота срабатывания достаточна для того, чтобы подпрограмма была полезной. Можно строго доказать, что ECM распознает большинство $y$-гладких чисел, меньших $x$, за $y^{o(1)} \ln x$ арифметических операций с целыми числами порядка $x$. Могут существовать некие исключительные числа, которые не поддаются методу ECM, но можно строго доказать, что их число невелико.

В связи с вопросом о тестах на гладкость можно упомянуть, что вероятностный алгоритм, анонсированный в статье [Lenstra et al. 1993b], распознает все $y$-гладкие числа $n$ за $y^{o(1)} \ln n$ арифметических операций. Это означает, что его эффективность аналогична эффективности ECM, однако в отличие от ECM эта оценка сложности строго обоснована; и строго доказано, что исключительных чисел нет.

## 6.4. Метод индекс-исчисления для дискретных логарифмов

В главе 5 мы описали некоторые общие алгоритмы вычисления дискретных логарифмов, работающие практически в любой циклической группе, для которой мы можем представлять групповые элементы с помощью компьютера и выполнять групповую операцию. Число шагов в этих экспоненциальных алгоритмах есть величина порядка корня квадратного из числа элементов группы. В группах, имеющих определенную специфику, у нас есть дополнительная информация, и ее можно выгодно использовать для вычислений, связанных с задачей DL. В этой главе мы видели, что вездесущие гладкие числа всюду выступают в качестве вспомогательного средства для разложения на множители. В некоторых группах можно придать смысл высказыванию о том, что некоторый элемент группы является гладким. В ситуации, когда это можно сделать, часто оказывается возможным решить задачу DL посредством субэкспоненциального алгоритма. Основная идея заключена в методе индекс-исчисления.

Сначала мы опишем метод индекс-исчисления для мультипликативной группы конечного поля $\mathbf{F}_p$, где $p$ — простое. Далее мы увидим, как можно использовать этот метод для всех конечных полей.

Тот факт, что для решения задач DL в мультипликативной группе конечного поля существуют субэкспоненциальные методы, подтолкнул криптографов к использованию других групп, наиболее популярными из которых являются группы эллиптических кривых, см. гл. 7.

### 6.4.1. Дискретные логарифмы в простых конечных полях

Рассмотрим мультипликативную группу $\mathbf{F}_p^*$, где $p$ — большое простое число. Эта группа — циклическая, каждый из порождающих ее элементов известен под названием «первообразного корня» (опр. 2.2.6). Пусть $g$ — первообразный корень, и $t$ — некоторый элемент группы. Задача DL для группы $\mathbf{F}_p^*$ состоит, при заданных $p, g, t$, в нахождении целого числа $l$, такого, что $g^l = t$. В действительности $l$ определяется этим равенством неоднозначно, подходящие целые числа $l$ образуют класс вычетов по модулю $p-1$. Мы пишем $l \equiv \log_g t \pmod{p-1}$.

То, что метод индекс-исчисления работает в $\mathbf{F}_p^*$, связано с тем, что нам не обязательно рассматривать $g$ и $t$ как абстрактные групповые элементы; можно считать их целыми числами, и мы можем рассматривать уравнение $g^l = t$ как сравнение $g^l \equiv t \pmod{p}$. Метод индекс-исчисления состоит из двух основных стадий. Первая стадия заключается в сборе соотношений. Это сравнения $g^r \equiv p_1^{r_1} \dots p_k^{r_k} \pmod{p}$, где $p_1, \dots, p_k$ — малые простые числа. Такие сравнения приводят к сравнениям для дискретных логарифмов:

$$r \equiv r_1 \log_g p_1 + \dots + r_k \log_g p_k \pmod{p-1}.$$

Если имеется достаточное количество таких соотношений, то тогда станет возможным использование линейной алгебры для решения этой системы относительно $\log_g p_i$. После этих предварительных вычислений, которые составляют основу метода, окончательное вычисление дискретного логарифма элемента $t$ довольно просто. Если у нас есть соотношение вида $g^R t \equiv p_1^{\tau_1} \dots p_k^{\tau_k} \pmod{p}$, то мы имеем, что

$$\log_g t \equiv -R + \tau_1 \log_g p_1 + \dots + \log_g p_k \pmod{p-1}.$$

Соотношения каждого сорта находятся посредством случайного выбора чисел $r, R$. Выбор числа $r$ приводит к некоторому вычету $g^r \bmod p$, который либо раскладывается полностью на малые простые сомножители $p_1, \dots, p_k$, либо не раскладывается. Аналогично, выбор числа $R$ приводит к вычету $g^R t \bmod p$. Выбирая вычеты, ближайшие к 0, и допуская в разложении на простые сомножители сомножитель $(-1)$, можно извлечь небольшую выгоду. Заметим, что нам не обязательно искать дискретный логарифм числа $(-1)$; про него нам известно, что он равен $(p-1)/2$. Подведем итог описанию метода индекс-исчисления для $\mathbf{F}_p^*$ следующим псевдокодом.

**Алгоритм 6.4.1** (метод индекс-исчисления для $\mathbf{F}_p^*$). **Нам заданы простое число $p$, первообразный корень $g$ и ненулевой вычет $t \pmod{p}$. Этот вероятностный алгоритм пытается найти $\log_g t$.**

1. [Установить границу гладкости]
   Выбрать границу гладкости $B$; // Разумные варианты выбора $B$ // см. в тексте.
   Найти простые числа $p_1, \dots, p_k$ в промежутке $[1, B]$;
2. [Поиск общих соотношений]

Выбирать случайные числа $r$ из промежутка $[1, p-2]$, пока не будет найдено $B$ чисел, таких, что $g^r \bmod p$ является $B$-гладким;

// Немного выгоднее использовать вычет числа $g^r \bmod p$, ближайший к 0.

3. [Линейная алгебра]

   С помощью какого-либо из методов линейной алгебры из найденных соотношений найти $\log_g p_1, \ldots, \log_g p_k$;

4. [Поиск специального соотношения]

   Выбирать случайные числа $R$ из промежутка $[1, p-2]$ и находить ближайший к 0 вычет числа $g^R t \pmod p$, пока не будет найдено такое число, вычет которого является $B$-гладким;

   Использовать найденное специальное соотношение вместе со значениями $\log_g p_1, \ldots, \log_g p_k$, найденными на шаге [Линейная алгебра], для нахождения $\log_g t$;

В связи с этим кратким описанием возникает несколько вопросов:

(1) Как определить, является ли число $B$-гладким?
(2) Как производить операции линейной алгебры по модулю составного числа $p-1$?
(3) Является ли число соотношений $B$ тем числом, которое дает разумные шансы на успех на шаге [Линейная алгебра]?
(4) Как выбрать подходящее $B$?
(5) Какова сложность этого метода, и действительно ли он субэкспоненциальный?

Что касается вопроса (1), существует несколько возможностей; в их число входят метод пробных делений, ро-метод Полларда (алгоритм 5.2.1) и метод эллиптических кривых (алгоритм 7.4.2). От выбранного метода зависит итоговая сложность, но при любом выборе метод индекс-исчисления остается субэкспоненциальным.

Существует определенная тонкость при осуществлении операций матричной алгебры над кольцом $\mathbf{Z}_n$ при составном $n$. На шаге [Линейная алгебра] нам требуется проделать это при $n = p-1$, которое является составным для всех простых чисел $p > 3$. Как и при решении полиномиальных сравнений, одна из идей состоит в сведении этой задачи к вычислениям по простым модулям. Матричная алгебра над $\mathbf{Z}_q$ при простом $q$ есть не что иное как матричная алгебра над конечным полем, и здесь работают как привычные методы Гаусса, так и различные быстрые методы. Как и в случае полиномиальных сравнений, также можно задействовать методы гензелева типа для операций матричной алгебры по модулю степеней простых и методы, основанные на китайской теореме об остатках, для склеивания степеней различных простых. Кроме того, не обязательно тратить много сил на разложение на множители. Если некоторый большой делитель числа $p-1$ на самом деле является составным и с трудом поддается дальнейшему разложению, то можно осуществлять операции матричной алгебры по модулю этого делителя так, как если бы он был простым. Если нам потребуется обратить ненулевой вычет, то, скорее всего, нам это удастся, а если нет, то разложение получится тривиально. Так что либо

мы успешно выполним операции матричной алгебры, что является первоочередной задачей, либо получим разложение модуля, и значит, сможем заново начать выполнение операций матричной алгебры, используя полученные уточнения в составе делителей.

Что касается вопроса (3), весьма вероятно, что при несколько большем, чем $\pi(B)$, числе соотношений вида $g^r \equiv p_1^{r_1} \dots p_k^{r_k} \pmod{p}$, где через $p_1, \dots, p_k$ обозначены все простые из промежутка $[1, B]$, найденное семейство векторов показателей $(r_1, \dots, r_k)$ порождает модуль $\mathbf{Z}_{p-1}^k$. Так что получение $B$ таких векторов есть небольшая перестраховка. Кроме того, даже не обязательно, чтобы эти векторы порождали весь модуль; обязательно лишь, чтобы вектор, отвечающий соотношению, найденному на шаге [Поиск специального соотношения], находился в порожденном ими подмодуле. Эта идея, таким образом, делает необязательным нахождение каждого отдельного значения $\log_g p_i$ на шаге [Линейная алгебра]. Иными словами, можно выполнять операции линейной алгебры лишь после того, как будет найдено специальное соотношение.

На два последних заданных вопроса можно дать общий ответ. Как и при анализе некоторых приведенных методов разложения на множители мы найдем, что асимптотически наилучший выбор $B$ имеет вид $L(p)^c$, где $L(p)$ определено соотношением (6.1). Если используется быстрый тест на гладкость, скажем, метод эллиптических кривых, мы берем $c = 1/\sqrt{2}$ и в итоге получаем сложность $L(p)^{\sqrt{2}+o(1)}$. Если же используется медленный тест на гладкость, скажем, метод пробных делений, то нам следует выбрать меньшее значение $c$, а именно $c = 1/2$, которое даст итоговую сложность вида $L(p)^{2+o(1)}$. Если используется тест на гладкость промежуточной сложности, то это приведет нас к промежуточному значению $c$ и к промежуточной сложности алгоритма.

В области конечных параметров этот асимптотический анализ следует воспринимать лишь как общее руководство, и оптимальный выбор параметров следует осуществлять разработчику на основании некоторого числа тестовых запусков. Подробности о методе индекс-исчисления для простых конечных полей см. в работе [Pomerance 1987b].

### 6.4.2. Вычисление дискретных логарифмов посредством гладких полиномов и гладких целых алгебраических чисел

Успешная работа и даже само существование метода индекс-исчисления для $\mathbf{F}_p$ обусловлены тем, что мы можем представлять себе $\mathbf{F}_p$ как $\mathbf{Z}_p$ и тем самым работать с элементами группы как с целыми числами. Но то, что $\mathbf{F}_{p^d}$ изоморфно $\mathbf{Z}_{p^d}$ при $d > 1$, уже *неверно*, и потому не существует удобного способа представления элементов конечных полей, не являющихся простыми, в виде целых чисел. Как мы убедились в разд. 2.2.2, мы можем реализовать $\mathbf{F}_{p^d}$ как поле частных $\mathbf{Z}_p[x]/(f(x))$, где $f(x)$ есть неприводимый многочлен из $\mathbf{Z}_p[x]$ степени $d$. Следовательно, мы можем поставить в соответствие каждому элементу $\mathbf{F}_{p^d}^*$ ненулевой многочлен из $\mathbf{Z}_p[x]$ степени, меньшей, чем $d$.

Кольцо многочленов $\mathbf{Z}_p[x]$ во многом похоже на кольцо целых чисел $\mathbf{Z}$. Оба являются областями с однозначным разложением на простые сомножители, где «простыми» в $\mathbf{Z}_p[x]$ являются неприводимые многочлены положительной степени с единичным старшим коэффициентом. Оба имеют лишь конечное число обратимых элементов (вычеты $1, 2, \ldots, p-1$ по модулю $p$ — в первом случае, и целые числа $\pm 1$ — в последнем), и наконец, оба кольца обладают понятием величины элемента. В самом деле, хоть $\mathbf{Z}_p[x]$ и не является упорядоченным кольцом, мы, тем не менее, имеем аналог понятия величины, которым является степень многочлена. Тем самым мы имеем понятие «гладкости» многочлена: будем говорить, что многочлен является $b$-гладким, если каждый из его неприводимых сомножителей имеет степень не выше $b$. Существует даже теорема, аналогичная (1.44): доля $b$-гладких многочленов из $\mathbf{Z}_p[x]$, степени, меньшей $d$, есть величина порядка $u^{-u}$, где $u = d/b$, для широкого диапазона переменных $p, d, b$.

Далее ясно, что все это не имеет большого смысла при малых $d$. Например, при $d = 2$ все полиномы являются 1-гладкими, и примерно $1/p$ от числа всех полиномов являются 0-гладкими. Однако при больших $d$ метод индекс-исчисления действительно работает и вычисляет дискретные логарифмы в $\mathbf{F}_{p^d}^*$, предоставляя нам метод, который является субэкспоненциальным (см. [Lovorn Bender and Pomerance 1998]).

А как быть в случаях, когда $d > 1$ и $d$ невелико? Имеется альтернативное представление поля $\mathbf{F}_{p^d}$, которое оказывается полезным в таких случаях. Пусть $K$ — алгебраическое числовое поле степени $d$ над полем рациональных чисел. Пусть $O_K$ обозначает кольцо целых алгебраических чисел в $K$. Если $p$ — простое число, которое является инертным в $K$, т. е. идеал $(p)$ в $O_K$ является простым идеалом, то структура частного $O_K/(p)$ изоморфна $\mathbf{F}_{p^d}$. Так что мы можем представлять себе элементы конечного поля как целые алгебраические числа. И как мы убедились на примере алгоритма NFS для разложения на множители, имеет смысл говорить о гладкости целых алгебраических чисел: а именно, целое алгебраическое число является $y$-гладким, если все простые делители его нормы в рациональных числах не превосходят $y$.

Проиллюстрируем здесь случай $d = 2$, когда $p$ — простое, сравнимое с 3 (mod 4). Берем $K = \mathbf{Q}[i]$, поле гауссовых рациональных чисел, т. е. множество $\{a + bi : a, b \in \mathbf{Q}\}$. Тогда $O_K$ есть $\mathbf{Z}[i] = \{a + bi : a, b \in \mathbf{Z}\}$, кольцо гауссовых целых чисел. Получаем, что $\mathbf{Z}[i]/(p)$ изоморфно конечному полю $\mathbf{F}_{p^2}$. Итак, метод индекс-исчисления по-прежнему работает, но сейчас вместо обыкновенных целых чисел мы имеем дело с гауссовыми целыми числами $a+bi$.

В случае $d = 2$ метод индекс-исчисления, использующий мнимое квадратичное поле, может быть полностью обоснован (см. [Lovorn 1992]). Предполагается, что использование других полей также возможно, но обоснование метода индекс-исчисления для таких случаев остается эвристическим.

Существуют эвристические методы, аналогичные алгоритму разложения NFS, для нахождения дискретных логарифмов в любом конечном поле $\mathbf{F}_{p^d}$, включая случай $d = 1$. Для широкого спектра случаев сложность эвристически

снижена до функций вида $\exp\left(c\left(\log p^d\right)^{1/3}\left(\log\log p^d\right)^{2/3}\right)$ (см. работы [Gordon 1993], [Schirokauer et al. 1996] и [Adleman 1994]). Эти методы можно назвать грандиозными обобщениями метода индекс-исчисления, а работают они благодаря такому представлению элементов группы, которое допускает понятие гладкости. Именно по этой причине криптографы склонны отказываться от мультипликативных групп конечных полей в пользу групп эллиптических кривых. Для групп эллиптических кривых у нас нет удобного понятия гладкости, и метод индекс-исчисления будет бесполезным. Для таких групп все лучшие универсальные методы решения задачи DL работают экспоненциальное время.

## 6.5. Упражнения

**6.1.** Вам дано составное число $n$, не являющееся степенью, и нетривиальное разложение $n = ab$. Опишите эффективный алгоритм нахождения нетривиального разложения числа $n$ на взаимно простые сомножители, т. е. нахождения таких взаимно простых чисел $A$, $B$, каждое из которых больше 1, что $n = AB$.

**6.2.** Покажите, что если $n$ — нечетное, составное, и не является степенью, то по меньшей мере половина пар $x, y$, таких, что $0 \leqslant x, y < n$, и $x^2 \equiv y^2 \pmod{n}$, обладает свойством $1 < \text{НОД}(x - y, n) < n$.

**6.3.** Иногда при использовании метода QS число $n$, которое требуется разложить, заменяют на $kn$ при малом целом $k$. Хотя использование множителя увеличивает порядок вычетов, просеиваемых на предмет гладкости, это можно в значительной мере компенсировать. Может случиться так, что под действием $k$ множество просеивающих простых чисел сместится в сторону меньших значений. Исследуйте варианты выбора множителя при использовании QS для разложения числа

$$n = 1883199855619205203.$$

В частности, сравните время, необходимое для разложения данного числа $n$, с временем, необходимым для разложения $3n$. (Это означает, что алгоритму задано число $3n$, и он в итоге должен выдать разложение $3n = ab$, где $3 < a < b$.) Далее исследуйте варианты выбора множителя при использовании QS для разложения числа

$$n = 215659417219997979398437139 63.$$

(Если Вас интересует фактическое построение программы, обсуждение деталей реализации см. в упр. 6.14.)

**6.4.** Существуют многочисленные методы разложения на множители, которые эксплуатируют идею «малых квадратов» в том виде, в котором она сформулирована в начале этой главы. В то время как QS и NFS являются мощными воплощениями этой идеи, существуют и другие, не настолько мощные, но все же интересные методы, которые используют вспомогательные разложения малых вычетов с получаемой в итоге линейной комбинацией, как и обсуждавшийся метод QS. Одним из ранних методов такого типа был метод непрерывных

дробей Бриллхарта—Моррисона (краткое резюме см. в книге [Cohen 2000]), который включал в себя использование разложения в непрерывную дробь числа $\sqrt{n}$ (или числа $\sqrt{kn}$ при малом целом $k$) с целью получения большого числа сравнений вида $Q \equiv x^2 \pmod{n}$, где $Q \neq x^2, |Q| = O(\sqrt{n})$. Далее пытаются разложить числа $Q$ для построения экземпляров сравнения $u^2 \equiv v^2 \pmod{n}$. Одним из первых триумфов этого метода было сокрушение числа $F_7$ Бриллхартом и Моррисоном в 1974 г. (см. табл. 1.3). В отношении величины порождаемых квадратичных вычетов $Q$ данный метод несколько лучше QS. Однако к последовательности чисел $Q$ не удается применить решето, и те, кто использует метод непрерывных дробей, вынуждены тратить на каждое значение $Q$ немалое количество времени, хотя большая часть из этих $Q$ в конце концов будет отвергнута по причине недостаточной гладкости.

Мы не будем далее углубляться в метод непрерывных дробей. Вместо этого мы приведем здесь разнообразные вопросы и задачи, предназначенные для пояснения — посредством практики, алгебры, и даже посредством некоторых занимательных(!) моментов — вопросов создания и использования метода «малых квадратов» по модулю заданного $n$, подлежащего разложению. Далее мы сосредоточимся на специальных числах, таких, как числа Ферма $n = F_k = 2^{2^k} + 1$ или числа Мерсенна $n = M_q = 2^q - 1$, поскольку работать с ними во многих отношениях проще по причине их специального вида. Но как и в случае могучего метода NFS, эти понятия большей частью могут быть распространены на составные числа $n$ более общего вида.

(1) Используйте явно выписанные сравнения

$$\begin{aligned} 258883717^2 \bmod M_{29} &= -2 \cdot 3 \cdot 5 \cdot 29^2, \\ 301036180^2 \bmod M_{29} &= -3 \cdot 5 \cdot 11 \cdot 79, \\ 126641959^2 \bmod M_{29} &= 2 \cdot 3^2 \cdot 11 \cdot 79 \end{aligned}$$

для составления соответствующего нетривиального сравнения вида $u^2 \equiv v^2$ и таким образом найдите делитель числа $M_{29}$.

(2) Оказывается, что $\sqrt{2}$ существует по модулю каждого из специальных чисел $n = F_k, k \geqslant 2$, и $n = M_q, q \geqslant 3$. И, что замечательно, можно явно выписать эти корни независимо от того, является $n$ составным, или нет. А именно, покажите, что

$$2^{3 \cdot 2^{k-2}} - 2^{2^{k-2}}, \qquad 2^{(q+1)/2}$$

являются квадратными корнями из 2 в случае чисел Ферма и Мерсенна, соответственно. Кроме того, выпишите явно примитивный корень 4-й степени из $(-1)$ в случае чисел Ферма и зависящий от $(q \bmod 4)$ корень 4-й степени из 2 в случае чисел Мерсенна. Кстати говоря, данные наблюдения имеют реальное практическое применение: теперь можно отбрасывать любую степень числа 2 в возводимом в квадрат вычете, поскольку теперь у нас есть явная формула для $\sqrt{2^k}$. Подобным образом в случае чисел Ферма в возводимых в квадрат вычетах могут быть отброшены степени $(-1)$.

(3) Используя идеи предыдущего пункта, с помощью только карандаша и бумаги докажите сравнение

$$2(2^6-8)^2 \equiv (2^6+1)^2 \pmod{M_{11}}$$

и получите отсюда разложение числа $M_{11}$.

(4) По счастливому стечению обстоятельств для определенного числа $\omega$, являющегося примитивным корнем 4-й степени из 2 по модулю $M_{43}$, мы имеем

$$\left(2704\omega^2-3\right)^2 \bmod M_{43} = 2^3\cdot 3^4\cdot 43^2\cdot 2699^2.$$

Воспользуйтесь этим обстоятельством для нахождения делителя числа $M_{43}$.

(5) Для числа $\omega$, которое является примитивным корнем 4-й степени из $(-1)$ по модулю $F_k, k\geqslant 2$, и для заданных целых чисел $a,b,c,d$ положим

$$x = a+b\omega+c\omega^2+d\omega^3.$$

Интересно, что определенные комбинации чисел $a,b,c,d$ автоматически дают малые квадраты — их можно назвать «символическими квадратами» — для любых указанных $F_k$. Покажите, что если принять ограничение

$$ad+bc=0,$$

то $x^2 \bmod F_k$ можно записать в виде многочлена от переменной $\omega$, степени, меньшей, чем 3. Так что, например,

$$\left(-6+12\omega+4\omega^2+8\omega^3\right)^2 \equiv 4(8\omega^2-52\omega-43),$$

причем коэффициенты этого сравнения остаются неизменными для всех указанных чисел Ферма (но $\omega$, конечно, зависит от числа Ферма). Используя эти идеи, укажите нижнюю границу при заданной постоянной $K$ числа возможных «символических квадратов», таких, что

$$|x^2 \bmod F_k| < K\sqrt{F_k}.$$

Далее укажите аналогичную оценку для малых квадратов по модулю чисел Мерсенна $M_q$.

(6) Развивая предыдущий пункт, исследуйте данный тип методов разложения в применении к нечетным составным числам более общего вида $N=\omega^4+1$, при этом используя квадрат фиксированной кубической формы, например,

$$x = -16+8\omega+2\omega^2+\omega^3,$$

по следующей схеме. Покажите, что $(-1)$ всегда является квадратом по модулю $N$, и

$$x^2 \equiv 236-260\omega-\omega^2 \pmod{N}.$$

Исходя из этого, найдите собственный делитель числа

$$N = 16452725990417,$$

отыскав определенный квадрат, который нетривиальным образом сравним с $x^2$. Конечно, разложение данного конкретного $N$ легко провести с помощью других методов, но этот пример показывает, что определенные формы вида $\omega^4+1$ допускают прямое использование такого формализма малых квадратов. Исследуйте, далее, способы такого подбора коэффициентов многочлена $x$, чтобы подобный формализм стал возможен для большого количества других чисел вида $N=\omega^4+1$.

Родственные идеи, касающиеся построения малых квадратов для разложения чисел, являющихся значениями определенных кубических форм, появились в работе [Zhang 1998].

**6.5.** Предположим, что Вы обладаете устройством, таким, что если Вы задаете ему натуральное число $n$ и целое число $a$ из промежутка $[1,n]$, то оно говорит Вам одно из решений сравнения $x^2\equiv a \pmod n$, если такое существует, или же говорит, что решений нет, если это так. Если данное сравнение имеет несколько решений, то устройство выбирает одно из них некоторым неизвестным Вам способом. Пусть данное устройство требует полиномиальное время для своей работы. Т. е. время, которое оно тратит на предоставление ответа, ограничено константой, умноженной на фиксированную степень логарифма $n$. Покажите, как, вооружившись такого рода устройством, можно выполнять разложение на множители посредством вероятностного алгоритма, ожидаемое время работы которого будет полиномиальным. Обратно, покажите, что если Вы можете раскладывать на множители за полиномиальное время, то Вы можете сконструировать такое устройство.

**6.6.** Предположим, что у Вас есть магический алгоритм, который при заданном числе $N$, подлежащем разложению, может автоматически (и при этом быстро, скажем, за полиномиальное время на один экземпляр) находить целые числа $x$, удовлетворяющие условиям

$$\sqrt{N}<x<N-\sqrt{N},\quad x^2 \bmod N<N^{\alpha}$$

при некотором фиксированном $\alpha$. (Заметим, что метод непрерывных дробей и квадратичное решето, в сущности, способны решать эту задачу при $\alpha\approx 1/2$.) Допустим далее, что каждое такое сравнение с «малыми квадратами» требует $O(\ln^{\beta} N)$ операций для того, чтобы выписать его явно. Укажите (эвристически) сложность, которая получается для разложения числа этим магическим алгоритмом.

**6.7.** Код Грея представляет собой последовательность $k$-битовых двоичных строк, упорядоченных таким образом, что при переходе от одной строки к другой ровно один бит изменяется на противоположный. Покажите, что такой код — либо для варианта QS с автоматической инициализацией, либо для каких-то других приложений — можно легко сгенерировать с помощью функции, которая использует операции исключающего или «$\wedge$» и сдвига «>>» следующим изящным образом:

$$g(n)=n\wedge(n>>1).$$

Легко видеть, что этот чрезвычайно простой генератор предоставляет нам, например, 3-битовый счетчик Грея, который выдает следующую цепочку:

$$(g(0),\ldots,g(7)) = (000,001,011,010,110,111,101,100),$$

в которой, очевидно, ровно один бит изменяется на каждой итерации.

**6.8.** Покажите, что если $n \geqslant 64$, и $m = \lfloor n^{1/3} \rfloor$, то $n < 2m^3$. Докажите более общее утверждение: если $d$ — положительное целое число, $n > 1.5(d/\ln 2)^d$, и $m = \lfloor n^{1/d} \rfloor$, то $n < 2m^d$.

**6.9.** Следующий результат, сводящий разложение целых чисел на множители к разложению многочленов на множители, доказан в работе [Brillhart et al. 1981].

**Теорема.** *Пусть $n$ — натуральное число, $m$ — целое число, такое что $m \geqslant 2$; запишем $n$ в системе с основанием $m$ как $n = f(m)$, где $f(x) = c_d x^d + c_{d-1}x^{d-1} + \ldots + c_0$, так что $c_i$ являются неотрицательными целыми числами, меньшими $m$. Допустим, что $f(x)$ приводим в $\mathbf{Z}[x]$, причем $f(x) = g(x)h(x)$, где ни $g(x)$, ни $h(x)$ не являются постоянными, равными $\pm 1$. Тогда $n = g(m)h(m)$ есть нетривиальное разложение числа $n$. В частности, если $n$ — простое, то $f(x)$ неприводим.*

Упражнение состоит в том, чтобы доказать эту теорему в случае $m \geqslant 3$, используя следующие указания:

(1) Докажите неравенство

$$\left|\frac{f(z)}{z^{d-1}}\right| \geqslant \operatorname{Re}(c_d z) + c_{d-1} - \sum_{j=2}^{d} \frac{c_{d-j}}{|z|^{j-1}}$$

и используйте его для доказательства того, что $f(z) \neq 0$ при $\operatorname{Re} z \geqslant m-1$. (Воспользуйтесь тем, что каждое $c_j$ удовлетворяет неравенствам $0 \leqslant c_j \leqslant m-1$, и $c_d \geqslant 1$.)

(2) Используя разложение многочлена в произведение по корням, покажите, что $|g(m)| > |c| \geqslant 1$, где $c$ — старший коэффициент многочлена $g(x)$, и, аналогично, что $|h(m)| > 1$. Таким образом, разложение $n = g(m)h(m)$ нетривиально.

**6.10.** Данное упражнение состоит в том, чтобы доказать теорему упр. 6.9 в оставшемся случае $m = 2$. Указание: с помощью несколько более точного неравенства, чем неравенство п. (1) упр. 6.9 (используя то, что $\operatorname{Re}(c_{d-2}/z) \geqslant 0$ при $\operatorname{Re}(z) > 0$), покажите, что каждый корень $\rho$ многочлена $f$ таков, что $\operatorname{Re}(\rho) < 1.49$. Затем положите $G(x) = g(x + 1.49)$ и покажите, что все коэффициенты рационального многочлена $G(x)$ имеют одинаковый знак. Выведите отсюда, что $1 \leqslant |g(1)| = |G(-0.49)| < |G(0.51)| = |g(2)|$, и, аналогично, что $|h(2)| > 1$, так что разложение $n = g(2)h(2)$ нетривиально.

**6.11.** Используйте метод упр. 6.9 для разложения числа $n = 187$ при помощи основания $m = 10$. Проделайте то же самое для $n = 4189, m = 29$.

**6.12.** Обобщите построение многочленов $x(u,v), y(u,v)$ из разд. 6.1.7 на произвольные числа $n$, удовлетворяющие условию (6.4).

**6.13.** Приведите эвристическое рассуждение, дающее для специального решета числового поля (SNFS) оценку сложности

$$\exp\left((c+o(1))(\ln n)^{1/3}(\ln\ln n)^{2/3}\right)$$

операций при $c=(32/9)^{1/3}$.

**6.14.** Здесь мы приводим некоторые практические примеры, связанные с QS, которые могут служить как образец при создании по-настоящему мощных его реализаций. В частности, читатель, желающий реализовать QS, может использовать нижеследующие примеры для отладки программы. Кстати говоря, каждый из приведенных ниже примеров, за исключением последнего, можно проделать в типичной программе для символьных вычислений, поддерживающей операции высокой точности. Так что это упражнение показывает, что числа размером 30 и более знаков можно обрабатывать даже без помощи быстрых откомпилированных программных реализаций.

(1) Для примера возьмем в алгоритме 6.1.1 очень малое число $n=10807$, и поскольку данное $n$ гораздо меньше чисел, составляющих типичный интервал практического применения QS, сделаем так, что в начале алгоритма граница гладкости $B=200$. Далее вам следует найти $k=21$ подходящих простых чисел. Вы получите двоичную матрицу размера $21\times 21$ и сможете применить к ней метод Гаусса. Кстати, для такой матричной алгебры существуют специальные пакеты, например, в языке системы *Mathematica* линейная система с матрицей $m$ может быть решена в нашем случае с помощью единственного оператора

```
r = NullSpace[Transpose[m], Modulus->2];
```

(хотя, как нам указал Лихтблау, можно оптимизировать общее число операций путем вмешательства на более низком уровне, используя, скажем, битовые операции, а не редукцию по модулю 2). С помощью этой команды находим, что существует строка матрицы решения $r$, которая имеет лишь 3 единицы; это приводит к соотношению

$$3^4\cdot 11^4\cdot 13^4\equiv 106^2\cdot 128^2\cdot 158^2 \pmod n$$

и тем самым к разложению числа $n$.

(2) Далее, в применении к несколько бóльшему составному числу, а именно $n=7001\cdot 70001$, попробуйте оставить значение переменной $B$ в алгоритме 6.1.1 *как есть*; в таком случае Вы должны получить $B=2305$, $k=164$. Получившаяся матрица размера $164\times 164$ не слишком велика в наш компьютерный век, так что разложение числа $n$ должно получиться с помощью того же подхода, что и в предыдущем пункте.

(3) А теперь попытайтесь разложить число Мерсенна $n=2^{67}-1$, используя при этом $B=80000$, что приводит к $k=3962$. Этот пример не только положит начало серьезному тестированию вашей реализации QS, он проде-

монстрирует, как посредством QS 12-значные делители можно находить в течение нескольких секунд или минут (в зависимости от эффективности решета и матричного пакета). Это пока что несколько медленнее прямого просеивания или, скажем, ро-методов Полларда, но, разумеется, QS можно перенести в область *много* больших чисел из-за его благоприятного асимптотического поведения.

(4) Попытайтесь разложить репьюнит (число из повторяющихся единиц)

$$n = \frac{10^{29} - 1}{9} = 11111111111111111111111111111,$$

используя жестко заданный параметр $B = 40000$, при котором матрицы будут примерно размером $2000 \times 2000$.

(5) Если Вас не удовлетворили приведенные упражнения, реализуйте быстрый откомпилированный вариант алгоритма 6.1.1 и попытайтесь разложить, скажем, 100-значное составное число.

**6.15.** Здесь в духе упр. 6.14 мы проработаем следующие явные примеры, относящиеся к алгоритму NFS 6.2.5. Вновь задача состоит в предоставлении читателю определенного руководства и средств для отладки алгоритма. Мы обнаружим, что одно из затруднений — нахождение квадратного корня в числовом поле — преодолевается различными способами в зависимости от масштаба задачи.

(1) Начните с несложного числа $n = 10403$ и сделайте вывод, что многочлен $f$ *приводим*, так что разложение получается уже на стадии [Начальная установка], и никакого просеивания не требуется.

(2) Используйте алгоритм 6.2.5, оставив параметры инициализации в листинге псевдокода *как есть*, для разложения $n = F_5 = 2^{32} + 1$. (Разумеется, данное составное число больше подходит для SNFS, но упражнение состоит в том, чтобы проверить работу общего метода NFS!) После инициализации мы получим $d = 2$, $B = 265$, $m = 65536$, $k = 96$, так что матричная размерность $V = 204$. Количество матричных операций точно такое же, как в упражнении 6.14, и в итоге Вы получите подходящее множество $\mathcal{S}$ пар $(a, b)$. Здесь для нашего малого составного $n$ (и соответствующих ему малых параметров) Вы можете на стадии [Квадратные корни] просто вычислить произведение $\prod_{(a,b)\in\mathcal{S}}(a - b\alpha)$ и получить гауссово число, поскольку допустимо присваивание $\alpha = i$. Заметьте, что случаи, когда $d = 2$, чрезвычайно удобны в том отношении, что вычисление квадратного корня в числовом поле становится тривиальным. Оказывается, квадратный корень из гауссова числа $c + di$ может быть получен путем решения простой системы соотношений. Так что для такого малого порядка, как $d = 2$, предпоследнюю стадию [Квадратные корни] алгоритма 6.2.5 можно упростить почти до предела.

(3) В качестве переключения на «вторую скорость» в том, что касается трудностей с квадратным корнем, снова возьмите то же самое составное число $n = F_5$, но задайте параметры $d = 4$, $B = 600$, которые обеспечат успешную работу NFS. Здесь на стадии [Квадратные корни] Вы можете снова

просто вычислить произведение членов $(a - b\alpha)$, где теперь $\alpha = \sqrt{i}$, и далее можете извлечь квадратный корень из получившегося элемента

$$s_0 + s_1\alpha + s_2\alpha^2 + s_3\alpha^3$$

в числовом поле. Для этого существуют несложные численные методы, например, сработает простой вариант обратной свертки из упр. 6.18; или можно использовать схему Вандермонда, обсуждаемую далее в настоящем упражнении.

(4) Далее, положите $n = 76409$ и на этот раз задайте параметры $d = 2$, $B = 96$, чтобы получился полином $f(x) = x^2 + 233x$. Теперь перед завершением алгоритма Вы можете вновь вычислить произведение членов $(a - b\alpha)$, а затем использовать несложные арифметические операции для нахождения квадратного корня в числовом поле, что позволяет завершить разложение.

(5) Точно таким же способом, как в предыдущем пункте, разложите репьюнит $n = 11111111111$, проинициализировав параметры так: $d = 2$, $B = 620$.

(6) Далее, для $n = F_6 = 2^{64} + 1$ положите $d = 4$, $B = 2000$, и на этот раз положим для удобства $k = 80$. Используйте любой из указанных методов для нахождения квадратного корня в числовом поле при $\alpha = \sqrt{i}$.

(7) Теперь мы готовы испытать «третью скорость» в том, что касается трудности с квадратным корнем. Разложите репьюнит $n = (10^{17} - 1)/9 = 1111111111111111$, положив при этом $d = 3$, $B = 2221$. На этот раз квадратные корни нужно извлекать в числовом поле, содержащем кубический корень из 1. В данной ситуации можно также обсудить матричный метод Вандермонда для извлечения корня. Сформируем элемент $\gamma^2$, который представляет собой форму $f'(\alpha)^2 \prod_{(a,b)\in S}(a - b\alpha)$, путем простого перемножения соответствующих членов по модулю $f(\alpha)$. (Подобная процедура, в принципе, работает всегда, хотя для достаточно большого $n$ коэффициенты получающегося $\gamma^2$ становятся громоздкими.) Матричный подход Вандермонда будет выглядеть так. Запишем величину, из которой необходимо извлечь квадратный корень, в виде

$$\gamma^2 = s_0 + s_1\alpha + \ldots + s_{d-1}\alpha^{d-1}.$$

Далее используйте $d$ корней многочлена $f$, скажем, $\alpha_1, \ldots, \alpha_d$ (вычисленных с достаточной степенью точности), для построения матрицы возрастающих степеней корней

$$H = \begin{pmatrix} 1 & \alpha_1 & {\alpha_1}^2 & {\alpha_1}^{d-1} \\ 1 & \alpha_2 & {\alpha_2}^2 & {\alpha_2}^{d-1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha_d & \alpha_d^2 & {\alpha_d}^{d-1} \end{pmatrix}.$$

Затем с достаточной степенью точности извлеките квадратные корни из *вещественных чисел*, а именно, вычислите вектор

$$\beta = \sqrt{Hs},$$

где $s = (s_0, \ldots, s_{d-1})$ — вектор коэффициентов элемента $\gamma^2$, а квадратный корень из произведения матрица-вектор берется покомпонентно. Идея состоит в вычислении произведений матрица-вектор

$$H^{-1} \begin{pmatrix} \pm\beta_0 \\ \pm\beta_1 \\ \vdots \\ \pm\beta_{d-1} \end{pmatrix},$$

где каждая из возможностей $\pm$ выбирается по одному разу, пока у вектора, получающегося в результате этого умножения на $H^{-1}$, не будут целыми все компоненты. Такой вектор будет являться квадратным корнем в числовом поле. Чтобы помочь Вам в разного рода реализациях, приведем в явном виде небольшой пример использования данного метода извлечения корня. Возьмем полином $f(x) = x^3 + 5x + 6$ и извлечем корень из величины $\gamma^2 = 117 - 366x + 46x^2$ по модулю $f(x)$ (мы будем пользоваться тем, что эта величина заведомо является квадратом). Построим матрицу Вандермонда, используя нули многочлена $f$, а именно $(\alpha_1, \alpha_2, \alpha_3) = \left(-1, \left(1 - i\sqrt{23}\right)/2, \left(1 + i\sqrt{23}\right)/2\right)$, в виде приближенной числовой матрицы, первая строка которой есть $(1, -1, 1)$, а элементы остальных строк — комплексные числа. Здесь нужна достаточная точность; для настоящего примера можно взять, скажем, 12 десятичных знаков после запятой. Здесь мы (покомпонентно) извлекаем квадратный корень и перебираем 8 возможных комбинаций $(\pm)$:

$$\gamma = H^{-1} \begin{pmatrix} \pm r_1 \\ \pm r_2 \\ \pm r_3 \end{pmatrix}, \qquad \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \sqrt{H \begin{pmatrix} 177 \\ -366 \\ 46 \end{pmatrix}}.$$

Достаточно очевидно, что одна из этих восьми комбинаций дает вектор

$$\gamma = \begin{pmatrix} 15 \\ -9 \\ -1 \end{pmatrix},$$

показывающий, что

$$\left(15 - 9x - x^2\right)^2 \bmod f(x) = 117 - 366x + 46x^2,$$

что и требовалось.

(8) Как и в случае упр. 6.14, с помощью программ для символьных вычислений нам удастся дойти лишь до этого места, и мы будем вынуждены перейти к быстрым откомпилированным программам, чтобы работать с большими составными числами. Тем не менее, с числами, имеющими порядка 30 десятичных цифр, в самом деле можно работать в режиме интерпретатора. Возьмите репьюнит $n = (10^{29} - 1)/9$, положите $d = 4$, $B = 30000$, и в этот раз пусть $k = 100$; Вы найдете разложение, которое успешно получается без помощи быстрых программ. В рассматриваемом

случае Вы можете использовать любой из приведенных методов для работы с числовыми полями степени 4; эти методы, как и прежде, используют грубое перемножение для величины $\gamma^2$ (хотя при заданных параметрах нам, возможно, уже нужна точность 3000 знаков, и наши хитроумные средства, обсуждаемые в тексте и в упр. 6.18, становятся на стадии извлечения корня чересчур трудоемкими для символьных вычислений).

Сформулированные выше конкретные задачи имеют целью сделать первый шаг на пути оттачивания серьезной реализации NFS. Однако существуют и другие возможности, которые можно реализовать даже для таких сравнительно небольших составных чисел. Например, свободные соотношения и другие оптимизации из разд. 6.2.7 могут помочь даже в сформулированных выше задачах, не говоря уже о том, что их следует применять для больших составных чисел.

**6.16.** Здесь мы решаем конкретную простую задачу DL, чтобы проиллюстрировать метод индекс-исчисления (алгоритм 6.4.1). Возьмем простое число $p = 2^{13} - 1$, первообразный корень $g = 17$, и пусть мы хотим решить сравнение $g^l \equiv 5 \pmod p$. Обратим внимание на следующие сравнения, которые можно мгновенно получить на компьютере:

$$\begin{aligned} g^{3513} &\equiv 2^3 \cdot 3 \cdot 5^2 \pmod p, \\ g^{993} &\equiv 2^4 \cdot 3 \cdot 5^2 \pmod p, \\ g^{1311} &\equiv 2^2 \cdot 3 \cdot 5 \pmod p. \end{aligned}$$

(В принципе, можно получать сравнения, установив границу гладкости простых делителей получаемого вычета и подставляя случайные степени первообразного корня $g$.) А теперь решите задачу DL, отыскав при помощи линейной алгебры три целых числа $a, b, c$, таких, что

$$g^{3513a+993b+1311c} \equiv 5 \pmod p.$$

## 6.6. Проблемы для исследования

**6.17.** Исследуйте предлагаемый ниже подход к построению субэкспоненциального алгоритма разложения на множители. Сначала обратите внимание на интересное алгебраическое тождество (см. [Crandall 1996a]):

$$\begin{aligned} F(x) &= \left((x^2 - 85)^2 - 4176\right)^2 - 2880^2 \\ &= (x-13)(x-11)(x-7)(x-1)(x+1)(x+7)(x+11)(x+13), \end{aligned}$$

так что $F$ фактически имеет 8 простейших алгебраических сомножителей в $\mathbf{Z}[x]$. Другим тождеством подобного типа является

$$\begin{aligned} G(x) &= ((x^2 - 377)^2 - 73504)^2 - 50400^2 \\ &= (x-27)(x-23)(x-15)(x-5)(x+5)(x+15)(x+23)(x+27), \end{aligned}$$

разумеется, есть и другие. Ввиду этого, для разложения числа $N = pq$ (где, скажем, простое $p \approx q$) можно просто вычислять НОД$(F(x) \bmod N, N)$ для

случайных $x \pmod N$, так что $N$ должно разложиться примерно за $\sqrt{N}/(2\cdot 8)$ вычислений значения многочлена $F$. (Дополнительный множитель 2 возникает из-за того, что в качестве делителя мы можем получить как $p$, так и $q$.) Поскольку $F$ вычисляется с использованием трех возведений в квадрат по модулю $N$, и нам потребуется одно умножение для аккумулирования нового значения произведения $F$, мы должны получить выигрыш в числе операций в $8/4 = 2$ раза по сравнению с наивным накоплением произведения. На самом деле выигрыш будет еще больше, если учесть простоту операции возведения в квадрат по модулю по сравнению с умножением по модулю. А что если нам удалось бы открыть подходящее множество $\{a_j\}$ фиксированных целых чисел и определить

$$H(x) = (\ldots((((x^2 - a_1)^2 - a_2)^2 - a_3)^2 - a_4)^2 - \ldots)^2 - a_k^2$$

так, чтобы в итоге $k$ возведений в квадрат (предполагаем, что $a_k^2$ вычислено заранее) порождали $2^k$ алгебраических множителей? Может ли эта идея с последовательными возведениями в квадрат напрямую привести к субэкспоненциальной (если не полиномиальной) сложности разложения на множители? Или же существуют непреодолимые преграды на пути к столь блестящему достижению? Другой вопрос: замечая, что приведенные выше многочлены $(F, G)$ имеют непересекающиеся множества корней, т. е. произведение $F(x)G(x)$ имеет 16 различных корней, можно ли как-то использовать два тождества для улучшения результата? Еще одно наблюдение: поскольку все корни произведения $F(x)G(x)$ нечетные, можно просто увеличить/уменьшить значение $x$, подставив $x \pm 1$, и получить новую группу множителей. Существует ли способ использовать это явление для улучшения результата?

Кстати говоря, существуют и другие тождества, которые требуют для получения желаемого произведения членов меньше операций, чем можно было бы ожидать. Например, мы имеем еще одно общее тождество, которое выглядит так:

$$\frac{(n+8)!}{n!} = \left(204 + 270n + 111n^2 + 18n^3 + n^4\right)^2 - 16(9 + 2n)^2;$$

оно позволяет вычислить произведение 8 последовательных целых чисел посредством 5 умножений (не считая умножений на константы). Так что даже если схема с чистыми возведениями в квадрат в начале этого упражнения не допускает обобщений, существуют, возможно, другие способы двигаться дальше.

Существуют и теоретические изыскания на эту тему. Например, Дильхер (см. [Dilcher 1999]) рассуждает о трудностях построения более длинных схем возведения в квадрат подобного типа. Недавно Д. Саймс открыл тождество для $k = 4$ с коэффициентами $(a_1, a_2, a_3, a_4)$, которые определяются конструкцией

$$(((x^2 - 67405)^2 - 3525798096)^2 - 533470702551552000)^2 - 469208209191321600^2;$$

с помощью программы для символьных вычислений читатель может убедиться, что данная конструкция в самом деле является произведением 16-и одно-

членов! П. Кармоди недавно сообщил, что многие из таких случаев с четырьмя возведениями в квадрат легко генерируются с помощью, скажем, GP/Pari скрипта.

**6.18.** Существуют ли неизвестные нам пока способы извлечения квадратных корней в числовых полях (эта операция необходима для успешной работы метода NFS)? В разд. 6.2.5 мы обсудили некоторые современные подходы, а в упр. 6.15 увидели, что существуют некоторые элементарные приемы. Здесь мы перечислим некоторые дальнейшие идеи и направления.

(1) Метод поднятия решений Гензеля, упомянутый в разд. 6.2.5, является своего рода $p$-адическим вариантом метода Ньютона. Существуют ли другие варианты метода Ньютона? Заметим, как и в упр. 9.14, что можно, в принципе, извлекать квадратные корни без вычисления обратных элементов, по крайней мере, в вещественном поле. Кроме того, существует метод Ньютона для решения *системы* нелинейных уравнений. Но набор таких уравнений есть то, что мы получаем при записи соотношений, означающих, что один многочлен есть квадрат другого (существует дополнительная сложность, связанная с рассуждениями по модулю $f$, но эти рассуждения, вероятно, можно провести в терминах матрицы Ньютона—Якоби).

(2) В числовых полях, связанных с многочленами простого вида $f(x) = x^d + 1$, оказывается возможным извлечение квадратных корней посредством «негациклической обратной свертки» (методы, относящиеся к последующему изложению, см. в разд. 9.5.3). Пусть величина, про которую известно, что для нее существует квадратный корень, записана в виде

$$\gamma^2 = \sum_{j=0}^{d-1} z_j \alpha^j,$$

где $\alpha$ — корень степени $d$ из $(-1)$ (т. е. корень многочлена $f$). Далее, в терминологии обработки сигналов мы говорим, что для некоторого сигнала $\gamma$ длины $d$, подлежащего определению, выполнено соотношение

$$z = \gamma \times_- \gamma,$$

где $\times_-$ означает негациклическую свертку, а $z$ есть сигнал, состоящий из коэффициентов $z_j$. Но нам известно, как вычислить негациклическую свертку с помощью методов быстрого преобразования. Записав

$$\Gamma_k = \sum_{j=0}^{d-1} \gamma_j \alpha^j \alpha^{-2kj},$$

можно установить тождество со сверткой с весами:

$$z_n = \alpha^{-n} \frac{1}{d} \sum_{k=0}^{d-1} \Gamma_k^2 \alpha^{+2nk}.$$

Теперь идея обратной свертки формулируется просто: получив сигнал $z$, из которого нужно извлечь квадратный корень, примените преобразование к последнему уравнению для получения коэффициентов $\Gamma_k^2$, затем зафиксируйте один из $2^{d-1}$ различных вариантов выбора знаков для соответствующих величин $\pm\sqrt{\Gamma_k^2}$, $k \in [1, d-1]$, затем найдите $\gamma_j$ посредством другого преобразования. Эта процедура обратной негациклической свертки в итоге дает правильный квадратный корень $\gamma$ из величины $\gamma^2$. Вопрос для исследования таков: поскольку нам известно, что задача для числовых полей, связанных с многочленом $f(x) = x^d + 1$, легко решается многими другими способами, можно ли обобщить данный подход с обратной сверткой? Как насчет $f(x) = x^d + c$ или даже более общего $f$? Также интересен вопрос о том, должны ли приведенные выше преобразования быть вещественными (они в действительности производятся за счет вычислений с высокой точностью), или же можно ввести абсолютно точные целочисленные теоретико-числовые преобразования.

(3) Для каждой из этих разнообразных идей первоочередным вопросом является следующий: как избежать стремительного роста коэффициентов? Поэтому необходимо иметь в виду, что процедура извлечения квадратного корня, будучи корректной с вычислительной точки зрения, также должна держать под контролем коэффициенты. Одно из общих предложений состоит в комбинировании какого бы то ни было метода извлечения корня с китайской теоремой об остатках, т. е. с работой по модулю многих простых чисел одновременно. На этом пути также возможен машинный параллелизм. Как мы упомянули в тексте главы, идеи Кувейня и Монтгомери превратили узкое место, связанное с извлечением корня, во вполне эффективную фазу современного NFS. Тем не менее, было бы неплохо иметь простую, ясную и высокоэффективную схему, которая обобщается на случаи, не зависящие от четности степени $d$, а также некоторым образом контролирует коэффициенты, не прибегая к реконструкции, основанной на китайской теореме об остатках.

Глава 7

# АРИФМЕТИКА ЭЛЛИПТИЧЕСКИХ КРИВЫХ

История того, что носит название эллиптических кривых, насчитывает более ста лет. Первоначально разработанный для целей классического анализа, аппарат эллиптических кривых нашел свое применение в абстрактной и вычислительной теории чисел, и теперь по праву занимает в этих областях фундаментальное положение. Как и сами простые числа, эллиптические кривые имеют замечательные аспекты — элегантность, сложность, мощь. Эллиптические кривые — это не только знаменитые алгебраические объекты, но и большое подспорье в вопросах, касающихся простых чисел и разложения на множители. Приложения эллиптических кривых простираются даже за пределы этих областей. Например, как мы увидим в разд. 8.1.3, возрастает их популярность в современной криптографии.

В дальнейшем наше основное внимание будет сосредоточено на эллиптических кривых над полями $\mathbf{F}_p$, где $p > 3$ — нечетное простое. Многие имеют представление о современном большом направлении исследований — даже, скорее, об индустрии — связанной с полями $\mathbf{F}_{p^k}$, где $k > 1$, или с (преобладающими в современных приложениях) полями $\mathbf{F}_{2^k}$. Но поскольку темой настоящей книги являются простые числа, мы решили ограничиться обсуждением первых полей, имеющих фундаментальное значение. За дополнительной информацией об остальных полях заинтересованный читатель может обратиться к литературе, например, к книге [Seroussi et al. 1999] и разнообразным журнальным статьям, которые там указаны.

## 7.1. Основы теории эллиптических кривых

Рассмотрим общее уравнение, представляющее собой многочлен 3-й степени от двух переменных с коэффициентами в поле $F$, приравненный к 0:

$$ax^3 + bx^2y + cxy^2 + dy^3 + ex^2 + fxy + gy^2 + hx + iy + j = 0. \qquad (7.1)$$

Чтобы этот многочлен на самом деле имел степень 3, будем предполагать, что по крайней мере одно из чисел $a, b, c, d$ отлично от нуля. Будем предполагать также, что этот многочлен абсолютно неприводим, т. е. неприводим в $\overline{F}[x, y]$, где $\overline{F}$ обозначает алгебраическое замыкание поля $F$. Можно рассматривать пары $(x, y) \in F \times F$, которые удовлетворяют (7.1). Они называются аффинными решениями этого уравнения. Или же можно рассматривать проективные решения. Для этого начнем с троек $(x, y, z) \in F \times F \times F$ ($x, y, z$ не все равны

нулю), которые удовлетворяют уравнению

$$ax^3 + bx^2y + cxy^2 + dy^3 + ex^2z + fxyz + gy^2z + hxz^2 + iyz^2 + jz^3 = 0. \quad (7.2)$$

Заметим, что $(x, y, z)$ является решением тогда и только тогда, когда $(tx, ty, tz)$ является решением, где $t \in F$, $t \neq 0$. Таким образом, в проективном случае более уместно говорить о решении, как об $[x, y, z]$, где эта запись означает, что мы отождествляем любые два решения $(x, y, z), (x', y', z')$ уравнения (7.2) тогда и только тогда, когда существует ненулевое $t \in F$, такое что $x' = tx, y' = ty, z' = tz$.

Проективные решения уравнения (7.2) есть почти то же самое, что аффинные решения уравнения (7.1). В частности, решение $(x, y)$ уравнения (7.1) можно отождествить с решением $[x, y, 1]$ уравнения (7.2), и любое решение $[x, y, z]$ уравнения (7.2) с $z \neq 0$ можно отождествить с решением $(x/z, y/z)$ уравнения (7.1). Решения $[x, y, z]$ с $z = 0$ не соответствуют никакому аффинному решению и называются «бесконечно удаленными точками» этого уравнения.

Уравнения (7.1) и (7.2) громоздки. Полезно рассмотреть замену переменных, которая переводит решения с координатами в поле $F$ в другие решения, а обратное преобразование — наоборот. Например, рассмотрим уравнение Ферма для показателя 3, а именно

$$x^3 + y^3 = z^3.$$

Будем рассматривать решения в поле $F$ характеристики, отличной от 2 и 3. Полагая $X = 12z$, $Y = 36(x - y)$, $Z = x + y$, получим равносильное уравнение

$$Y^2Z = X^3 - 432Z^3.$$

Обратной заменой переменных будет $x = \frac{1}{72}Y + \frac{1}{2}Z$, $y = -\frac{1}{72}Y + \frac{1}{2}Z$, $z = \frac{1}{12}X$.

Говорят, что проективная кривая (7.2) «неособая» (или «гладкая») над полем $F$, если и над алгебраическим замыканием поля $F$ на этой кривой не существует точки $[x, y, z]$, в которой все три частные производные обращаются в 0. Оказывается, что если характеристика поля $F$ отлична от 2 и 3, любое неособое проективное уравнение (7.2) с хотя бы одним решением в $F \times F \times F$ (отличным от решения, у которого все координаты равны 0) можно привести заменой переменных к стандартному виду

$$y^2z = x^3 + axz^2 + bz^3, \quad a, b \in F, \quad (7.3)$$

при этом заданное решение первоначального уравнения переходит в $[0, 1, 0]$. Далее, ясно, что кривая, заданная уравнением (7.3), имеет только одну бесконечно удаленную точку — $[0, 1, 0]$. Аффинная форма этого уравнения —

$$y^2 = x^3 + ax + b. \quad (7.4)$$

Такая форма записи уравнения кубической кривой называется формой Вейерштрасса. Иногда бывает удобно заменить $x$ на $(x+\text{константа})$, и таким образом получить другую форму Вейерштрасса:

$$y^2 = x^3 + Cx^2 + Ax + B, \quad A, B, C \in F. \quad (7.5)$$

Если мы имеем кривую, заданную в виде (7.4), и характеристика поля $F$ отлична от 2 и 3, то такая кривая неособа в том и только в том случае, когда

$4a^3+27b^2$ не равно 0 (см. упр. 7.3). Если же кривая задана в виде (7.5), условие ее неособости усложняется. Оно выглядит так: $4A^3+27B^2-18ABC-A^2C^2+4BC^3\neq 0$.

Как при работе с аффинной формой (7.4), так и с формой (7.5), мы будем пользоваться обозначением $O$ для единственной бесконечно удаленной точки $[0,1,0]$, которая возникает при записи этой кривой в проективной форме.

Сейчас мы дадим основное определение этой главы.

**Определение 7.1.1.** Неособая кубическая кривая (7.2) с коэффициентами в поле $F$, на которой существует хотя бы одна точка с координатами в поле $F$ (из которых не все равны нулю), называется эллиптической кривой над полем $F$. Если характеристика поля $F$ отлична от 2 и 3, то уравнения (7.4) и (7.5) также определяют эллиптические кривые над полем $F$ при условии, что $4a^3+27b^2\neq 0$ в случае уравнения (7.4) и $4A^3+27B^2-18ABC-A^2C^2+4BC^3\neq 0$ в случае уравнения (7.5). В двух названных случаях будем обозначать через $E(F)$ множество точек с координатами в поле $F$, удовлетворяющих этому уравнению, наряду с бесконечно удаленной точкой, обозначаемой через $O$. Так что в случае уравнения (7.4)

$$E(F)=\{(x,y)\in F\times F: y^2=x^3+ax+b\}\cup\{O\},$$

и аналогично для кривой, заданной уравнением (7.5).

Заметим, что наше внимание сосредоточено на полях характеристики, отличной от 2 и 3. Для полей типа $\mathbf{F}_{2^m}$ необходимо использовать видоизмененное уравнение (7.11) упражнения 7.1 (объяснение этого см., например, в книге [Koblitz 1994][1]).

Мы используем форму (7.5), поскольку иногда она полезна с вычислительной точки зрения, например, в криптографии и в задачах разложения на множители. Поскольку форма (7.4) есть частный случай формы (7.5) при $C=0$, достаточно выписать произвольную формулу для формы (7.5), предоставляя читателю возможность переписать ее для формы (7.4), когда квадратичный член отсутствует. Однако важно отметить, что уравнение (7.5) является переопределенным из-за дополнительного параметра. Одним словом, форма Вейерштрасса (7.4) обладает всей общностью для кривых над рассматриваемыми полями, но иногда наша параметризация (7.5) удобна с вычислительной точки зрения.

Следующие классы параметров имеют особую практическую ценность:

(1) $C=0$, что сразу приводит к форме Вейерштрасса $y^2=x^3+Ax+B$. Такая параметризация является стандартной для большей части теоретической работы с эллиптическими кривыми.

(2) $A=1$, $B=0$, так что кривые имеют вид $y^2=x^3+Cx^2+x$. Эта параметризация особенно важна для практической реализации разложения на множители (см. [Montgomery 1987], [Brent et al. 2000]) и является арифметически выгодной на практике.

[1] Имеется перевод: [Коблиц 2001]. — *Прим. ред.*

(3) $C = 0,\ A = 0$, так что кривая принимает вид $y^2 = x^3 + B$. Такая форма представляет ценность при нахождении специальных кривых, имеющих заданный порядок (т. е. заданное число элементов множества $E$, см. ниже), и также является арифметически выгодной с практической точки зрения.
(4) $C = 0,\ B = 0$, так что кривая есть $y^2 = x^3 + Ax$, с такими же преимуществами, как в пункте (3).

Невообразимая мощь эллиптических кривых становится реальной, когда мы определяем специальную групповую операцию, с которой $E(F)$ становится, фактически, абелевой группой.

**Определение 7.1.2.** Пусть $E(F)$ есть эллиптическая кривая, определяемая уравнением (7.5) над полем $F$ характеристики, отличной от 2 и 3. Обозначая две произвольные точки на кривой (не обязательно различные) через $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$, и через $O$ — бесконечно удаленную точку, определим коммутативную операцию $+$ с обратной операцией $-$ следующим образом:

(1) $-O = O$;
(2) $-P_1 = (x_1, -y_1)$;
(3) $O + P_1 = P_1$;
(4) если $P_2 = -P_1$, то $P_1 + P_2 = O$;
(5) если $P_2 \neq -P_1$, то $P_1 + P_2 = (x_3, y_3)$, где

$$\begin{aligned} x_3 &= m^2 - C - x_1 - x_2, \\ -y_3 &= m(x_3 - x_1) + y_1, \end{aligned}$$

и *угловой коэффициент* $m$ определяется равенствами

$$m = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & \text{если } x_2 \neq x_1, \\ \dfrac{3x_1^2 + 2Cx_1 + A}{2y_1}, & \text{если } x_2 = x_1. \end{cases}$$

Определенные таким образом операции сложения/вычитания допускают интересную геометрическую интерпретацию в случае, когда основное поле $F$ является полем действительных чисел. А именно, 3 точки на такой кривой лежат на одной прямой тогда и только тогда, когда их сумма равна 0. Эта интерпретация обобщена с учетом возможности двойного пересечения в точке касания (если только она не является точкой перегиба, в таком случае имеет место тройное пересечение). Наконец, такая геометрическая интерпретация позволяет считать, что вертикальные прямые пересекают нашу кривую в бесконечно удаленной точке. В случае же конечного поля, скажем $F = \mathbf{F}_p$, геометрическая интерпретация не так наглядна, если представлять себе $\mathbf{F}_p$ как целые числа по модулю $p$. В частности, операции деления для углового коэффициента $m$ суть вычисления обратных элементов $(\mathrm{mod}\ p)$.

Чудесным результатом теории эллиптических кривых является то, что операции на кривой в определении 7.1.2 задают группу. Более того, эта группа обладает специальными свойствами, зависящими от основного поля. Мы объединим эти результаты в следующую теорему.

**Теорема 7.1.3** (Касселс). *Эллиптическая кривая $E(F)$ с операциями определения* 7.1.2 *есть абелева группа. В случае конечного поля группа $E(\mathbf{F}_{p^k})$ либо циклическая, либо изоморфна произведению двух циклических групп:*

$$E \cong \mathbf{Z}_{d_1} \times \mathbf{Z}_{d_2},$$

*где $d_1|d_2$ и $d_1|p^k-1$.*

То, что $E$ есть абелева группа показать несложно, лишь доказательство ассоциативности немного утомительно (см. упр. 7.7). Результат о структуре $E(\mathbf{F}_{p^k})$ можно найти в работах [Cassels 1966], [Silverman 1986], [Cohen 2000].

Если поле $F$ конечно, $E(F)$ всегда конечная группа, а ее порядок, $\#E(F)$, т. е. число точек $(x, y)$ на аффинной кривой плюс 1 для бесконечно удаленной точки, есть число, которое приводит к завораживающим и глубоким результатам. И действительно, вопрос о порядке будет возникать в таких областях, как доказательство простоты, разложение на множители и криптография.

Определим эллиптическое умножение на целые числа естественным образом: для точки $P \in E$ и натурального числа $n$ будем обозначать $n$-е кратное этой точки через

$$[n]P = P + P + \ldots + P,$$

где справа стоит в точности $n$ экземпляров точки $P$. По определению $[0]P$ есть нуль аддитивной группы, т. е. бесконечно удаленная точка $O$. Далее, определяем $[-n]P$ как $-[n]P$. Из теории групп известно, что если $E(F)$ конечна, то

$$[\#E(F)]P = O;$$

этот факт имеет первостепенную важность в практических приложениях эллиптических кривых. Эти приложения порядка кривой обсуждаются более подробно в разд. 7.5. Как и в любой группе, мы можем рассматривать порядок любого элемента. В группе эллиптической кривой порядок точки $P$ есть наименьшее натуральное число $n$, такое, что $[n]P = O$, а если такого числа $n$ не существует, то говорят, что $P$ имеет бесконечный порядок. Если $E(F)$ конечна, то каждая точка группы $E(F)$ имеет конечный порядок, делящий $\#E(F)$.

Фундаментальное значение для применения эллиптических кривых при разложении на множители будет иметь тот факт, что если нужно разложить составное число $n$, то можно пытаться работать на эллиптической кривой над $\mathbf{Z}_n$, хотя $\mathbf{Z}_n$ и не является полем, и такое обращение с ним можно рассматривать как «незаконное». Когда возникает незаконная операция на кривой, она используется для нахождения делителя числа $n$. Эта идея о так называемых «псевдокривых» является отправной точкой метода эллиптических кривых (ECM) Х. Ленстры для разложения на множители, подробности которого обсуждаются в разд. 7.4. Перед тем как мы приступим к этому замечательному алгоритму, мы вначале обсудим «законную» арифметику эллиптических кривых над полем.

## 7.2. Арифметика эллиптических кривых

Вооруженные некоторыми основами теории эллиптических кривых, перейдем к разработке практических алгоритмов для их арифметики. Для простоты примем в качестве основного поля конечное поле $\mathbf{F}_p$ при простом $p > 3$, хотя, вообще говоря, структура алгоритмов останется такой же и для других полей. Мы начнем с простого метода нахождения в явном виде точек $(x, y)$ на заданной кривой, идея которого состоит в требовании, чтобы соответствующая кубическая форма от $x$ была квадратом по модулю $p$.

**Алгоритм 7.2.1** (нахождение точки на заданной эллиптической кривой). Предположим, что для простого $p > 3$ эллиптическая кривая $E(\mathbf{F}_p)$ задана кубическим уравнением $y^2 = x^3 + ax + b$. Этот алгоритм возвращает точку $(x, y)$ на кривой $E$.

1. [Цикл]
   Выбрать случайное $x \in [0, p-1]$;
   $t = (x(x^2 + a) + b) \bmod p$; // Аффинная кубическая форма от $x$.
   if$(\left(\frac{t}{p}\right) == -1)$ goto [Цикл]; // Посредством алгоритма 2.3.5.
   return $(x, \pm\sqrt{t} \bmod p)$; // Квадратный корень посредством алгоритма 2.3.8 или 2.3.9.

Можно возвращать любой из двух квадратных корней из найденного вычета, поскольку утверждение $(x, y) \in E(\mathbf{F}_p)$ влечет утверждение $(x, -y) \in E(\mathbf{F}_p)$. Хотя этот алгоритм и вероятностный, можно ожидать, что метод потребует всего несколько итераций do-цикла.

Здесь есть еще один важный момент: в определенных задачах $y$-координата не требуется, и мы всегда можем проверить, что некоторая точка $(x, ?)$ существует — т. е. $x$ есть допустимая $x$-координата — просто проверив, что символ Якоби $\left(\frac{t}{p}\right)$ не равен $-1$.

Указанные средства нахождения точки на заданной кривой полезны при доказательстве простоты и в криптографии. Но возникает другой интересный вопрос: как можно *одновременно* искать случайную кривую и точку на указанной кривой? Этот вопрос важен при разложении на множители. Мы отложим такой алгоритм до разд. 7.4, где затронуты «псевдокривые» с арифметикой по модулю составного $n$.

Но если задана точка $P$ или некоторая совокупность точек на кривой $E$, как нам складывать их попарно? И самое главное, как нам вычислять эллиптические кратные $[n]P$? Для осуществления таких операций есть несколько возможностей.

Возможность (1): аффинные координаты. Использовать фундаментальные групповые операции определения 7.1.2 непосредственно; такой подход в общем случае предполагает вычисление обратного элемента при выполнении операции на кривой.

Возможность (2): проективные координаты. Использовать групповые операции, но для проективных координат $[X, Y, Z]$, чтобы избежать вычисления

обратных элементов. Когда $Z \neq 0$, $[X, Y, Z]$ соответствует аффинной точке $(X/Z, Y/Z)$ на кривой. Точка $[0, 1, 0]$ есть $O$, бесконечно удаленная точка.

Возможность (3): модифицированные проективные координаты. Использовать тройки $\langle X, Y, Z\rangle$, которые, если $Z \neq 0$, соответствуют аффинным точкам $(X/Z^2, Y/Z^3)$ на кривой, и точку $\langle 0, 1, 0\rangle$, которая соответствует бесконечно удаленной точке $O$. Эта система также позволяет избежать вычисления обратных элементов и требует меньшего числа операций, чем проективные координаты.

Возможность (4): $X, Z$ координаты, иногда называемые координатами Монтгомери. Использовать координаты $[X : Z]$, которые представляют собой те же проективные координаты $[X, Y, Z]$, но с опущенным $Y$. Можно восстановить $x$-координату аффинной точки при $Z \neq 0$ по формуле $x = X/Z$. В общем случае существует два варианта выбора $y$, и это так и остается неоднозначностью. Обсуждаемая возможность хорошо подходит для эллиптического умножения и если $y$-координаты не требуются на каждом шаге, как это иногда случается при разложении на множители и в криптографии, или когда наша эллиптическая алгебра должна работать в высших областях, где сами координаты могут быть многочленами.

Какой из этих алгоритмических подходов является наилучшим, зависит от влияния различных факторов. Например, если при основном поле $\mathbf{F}_p$ у нас есть быстрое вычисление обратного $(\bmod p)$, можно выбрать указанную выше возможность (**1**). С другой стороны, если у нас уже программно реализована возможность (1), и мы желаем уменьшить затраты времени на (медленное) вычисление обратного, можно обратиться к возможностям (2) или (3), как мы далее увидим, при минимальных изменениях в работе алгоритмов. Если мы хотим построить программную реализацию «с нуля», можно указать на возможность (4), особенно в случае разложения на множители очень больших чисел с помощью ECM; в этом случае можно совсем обойтись без вычисления обратных $(\bmod n)$, где $n$ — составное.

Для полноты картины явный вид арифметики эллиптических кривых начнем изучать с возможности (1), хотя операции для этой возможности и легко вывести прямо из определения 7.1.2. Важно заметить, что здесь и в последующих алгоритмах приведены операции для основного поля $F$, хотя дальнейшая работа с «псевдокривыми», как например при разложении на множители составного $n$, и предусматривает использование кольца $\mathbf{Z}_n$ с операциями по модулю $n$, а не по модулю $p$; в то же время обобщение на поля $\mathbf{F}_{p^k}$ предполагает либо прямую полиномиальную, либо эквивалентную арифметику и т. д.

**Алгоритм 7.2.2** (эллиптическое сложение: аффинные координаты). Предполагаем, что эллиптическая кривая $E(F)$ (см. замечание перед этим алгоритмом) задана аффинным уравнением $Y^2 = X^3 + aX + b$, где $a, b \in F$ и характеристика поля $F$ отлична от 2 и 3. Представляем точки $P$ тройками $(x, y, z)$, где для аффинной точки $z = 1$ и $(x, y)$ лежит на аффинной кривой, а для бесконечно удаленной точки $O$ $z = 0$ (обе тройки $(0, 1, 0), (0, -1, 0)$ означают одну и ту же

точку). Настоящий алгоритм предоставляет функции для обращения знака, удвоения, сложения и вычитания точек.

1. [Функция эллиптическое обращение знака]
   $neg(P)$ return $(x, -y, z)$;
2. [Функция эллиптическое удвоение]
   $double(P)$ return $add(P, P)$;
3. [Функция эллиптическое сложение]
   $add(P_1, P_2)\{$
   if($z_1 == 0$) return $P_2$; // Точка $P_1 = O$.
   if($z_2 == 0$) return $P_1$; // Точка $P_2 = O$.
   if($x_1 == x_2$) {
   if($y_1 + y_2 == 0$) return $(0, 1, 0)$; // т. е. возвратить $O$.
   $m = (3x_1^2 + a)(2y_1)^{-1}$; // Вычисление обратного в поле $F$.
   } else {
   $m = (y_2 - y_1)(x_2 - x_1)^{-1}$; // Вычисление обратного в поле $F$.
   }
   $x_3 = m^2 - x_1 - x_2$;
   return $(x_3, m(x_1 - x_3) - y_1, 1)$;
   }
4. [Функция эллиптическое вычитание]
   $sub(P_1, P_2)$ return $add(P_1, neg(P_2))$;

В случае возможности (2), используя обычные проективные координаты, рассмотрим кривую $Y^2Z = X^3 + aXZ^2 + bZ^3$ и точки $P_i = [X_i, Y_i, Z_i]$, где $i = 1, 2$. Правило (5) определения 7.1.2 для суммы $P_1 + P_2$ при $P_1 \neq \pm P_2$ и $P_1, P_2$ не равны $O$ будет выглядеть так:

$$P_3 = P_1 + P_2 = [X_3, Y_3, Z_3],$$

где

$$\begin{aligned} X_3 &= \alpha\left(\gamma^2\zeta - \alpha^2\beta\right), \\ Y_3 &= \frac{1}{2}\left(\gamma\left(3\alpha^2\beta - \gamma^2\zeta\right) - \alpha^3\delta\right), \\ Z_3 &= \alpha^3\zeta, \end{aligned}$$

и

$$\alpha = X_2Z_1 - X_1Z_2, \quad \beta = X_2Z_1 + X_1Z_2,$$
$$\gamma = Y_2Z_1 - Y_1Z_2, \quad \delta = Y_2Z_1 + Y_1Z_2, \quad \zeta = Z_1Z_2.$$

Учитывая затраты на промежуточные вычисления $\alpha^2, \alpha^3, \alpha^2\beta, \gamma^2\zeta$, координаты точки $P_1 + P_2$ можно вычислить за 14 умножений в поле и 8 сложений в поле (умножение на $1/2$ можно в общем случае осуществить посредством сдвига или сложения и сдвига). В случае удвоения точки по правилу (5), если $[2]P \neq O$, проективные равенства для

$$[2]P = [2][X, Y, Z] = [X', Y', Z']$$

суть

$$\begin{aligned} X' &= \nu(\mu^2 - 2\lambda\nu), \\ Y' &= \mu\left(3\lambda\nu - \mu^2\right) - 2Y^2\nu^2, \\ Z' &= \nu^3, \end{aligned}$$

где

$$\lambda = 2XY, \quad \mu = 3X^2 + aZ^2, \quad \nu = 2YZ.$$

Итак, удвоение можно осуществить за 13 умножений в поле и 4 сложения в поле. Ни при сложении, ни при удвоении вычисления обратных величин к переменным не требуется.

Когда мы используем проективные координаты и начинаем с заданной аффинной точки $(u, v)$, мы с легкостью создаем проективные координаты, дописывая к этой точке 1, т. е. создавая проективную точку $[u, v, 1]$. Если мы желаем восстановить аффинную точку из точки $[X, Y, Z]$ в конце длинных вычислений, то если это не бесконечно удаленная точка, мы вычисляем $Z^{-1}$ в поле и получаем аффинную точку $(XZ^{-1}, YZ^{-1})$.

Мы увидим, что возможность (3) также позволяет обойтись без вычисления обратных величин в поле. Если сравнивать с возможностью (2), то сложение для возможности (3) обходится дороже, а удвоение — дешевле. Поскольку при типичном эллиптическом умножении $[n]P$ нам следует ожидать примерно вдвое больше удвоений, чем сложений, понятно, что возможность (3) можно предпочесть возможности (2). Вспоминая обозначения, под $\langle X, Y, Z\rangle$ будем понимать аффинную точку $(X/Z^2, Y/Z^3)$ на кривой $y^2 = x^3 + ax + b$, если $Z \neq 0$, а под $\langle 0, 1, 0\rangle$ будем понимать бесконечно удаленную точку. И снова, если мы начинаем с аффинной точки $(u, v)$ на нашей кривой и желаем перейти к модифицированным проективным координатам, мы просто дописываем к ней 1, создавая точку $\langle u, v, 1\rangle$. А если мы имеем модифицированную проективную точку $\langle X, Y, Z\rangle$, отличную от бесконечно удаленной, и желаем найти соответствующую ей аффинную точку, мы вычисляем $Z^{-1}, Z^{-2}, Z^{-3}$ и искомую аффинную точку $(XZ^{-2}, YZ^{-3})$. Следующий алгоритм осуществляет алгебру модифицированных проективных координат, т. е. возможность (3).

**Алгоритм 7.2.3** (эллиптическое сложение: модифицированные проективные координаты). Предполагаем, что эллиптическая кривая $E(F)$ над полем $F$ характеристики $\neq 2, 3$ (однако см. замечание перед алгоритмом 7.2.2) задана своим аффинным уравнением $y^2 = x^3 + ax + b$. Данный алгоритм работает с модифицированными проективными точками вида $P = \langle X, Y, Z\rangle$, причем обе точки $\langle 0, 1, 0\rangle, \langle 0, -1, 0\rangle$ означают бесконечно удаленную точку $P = O$, и предоставляет функции для обращения знака, удвоения, сложения и вычитания точек.

1. [Функция эллиптическое обращение знака]
   $neg(P)$ return $\langle X, -Y, Z\rangle$;
2. [Функция эллиптическое удвоение]
   $double(P)$ {
   if($Y == 0$ or $Z == 0$) return $\langle 0, 1, 0\rangle$;

```
        M = (3X^2 + aZ^4);  S = 4XY^2;
        X' = M^2 - 2S;  Y' = M(S - X') - 8Y^4;  Z' = 2YZ;
        return ⟨X', Y', Z'⟩;
    }
```

3. [Функция эллиптическое сложение]

```
    add(P1, P2) {
        if(Z1 == 0) return P2;                          // Точка P1 = O.
        if(Z2 == 0) return P1;                          // Точка P2 = O.
        U1 = X2 Z1^2;  U2 = X1 Z2^2;
        S1 = Y2 Z1^3;  S2 = Y1 Z2^3;
        W = U1 - U2;  R = S1 - S2;
        if(W == 0) {                                    // x-координаты совпадают.
            if(R == 0) return double(P1);
            return ⟨0, 1, 0⟩;
        }
        T = U1 + U2;  M = S1 + S2;
        X3 = R^2 - TW^2;
        Y3 = 1/2((TW^2 - 2X3)R - MW^3);
        Z3 = Z1 Z2 W;
        return ⟨X3, Y3, Z3⟩;
    }
```

4. [Функция эллиптическое вычитание]

```
    sub(P1, P2) {
        return add(P1, neg(P2));
    }
```

Следует подчеркнуть, что во всех наших алгоритмах эллиптического сложения при выполнении арифметики в $\mathbf{Z}_n$ редукция по модулю производится всякий раз, как только промежуточные числа становятся больше модуля. Алгоритм возможности (3) (модифицированные проективные координаты) очевидно требует больше умножений в поле, чем возможность (1) (аффинные координаты), но, как было сказано, его цель — обойтись без вычисления обратных элементов (см. упр. 7.9). Будем иметь в виду, что при программной реализации алгоритма 7.2.3 следует сохранить некоторые промежуточные вычисления для дальнейшего использования. Не на все такие вычисления явно указывает приведенный текст этого алгоритма. В частности, в функции эллиптическое сложение значение $W^2$, используемое для $X_3$, берется из памяти при вычислении $W^3$, нужного для $Y_3$, аналогично поступаем со значением $TW^2$. Если позаботиться о сохранении промежуточных вычислений, функция *double*() потребует 10 умножений в поле. (Однако при малых $a$ или для частного случая $a = -3$ в поле это значение 10 можно еще более уменьшить. См. упр. 7.10.) Общая функция сложения *add*(), в свою очередь, требует 16 умножений в поле, но важно, что эту оценку можно уточнить: при $Z_1 = 1$ требуется только 11 умножений. А это дополнительное условие обычно выполняется. На самом деле, оно всегда выполняется в определенных классах умножающих схем. (В

случае обычных проективных координат, обсуждавшихся перед алгоритмом 7.2.3, условие $Z_1 = 1$ уменьшает количество умножений с 14, необходимых для сложения в общем случае, также до 11.)

Обсудив возможности (1), (2), (3) для осуществления эллиптической арифметики, мы готовы обсудить эллиптическое умножение, т. е. задачу о вычислении $[n]P$, в которой целое число $n$ действует на точки $P \in E$. Можно, конечно, использовать для этой цели алгоритм 2.1.5. Но поскольку удвоение обходится настолько дешевле сложения двух неравных точек, и поскольку вычитание стоит столько же, сколько сложение, предпочтительный метод — это модифицированная двоичная схема, так называемая схема сложений—вычитаний. Для большинства чисел $n$ она имеет большее отношение удвоений к операциям сложения—вычитания, чем стандартные двоичные схемы, такие, как в алгоритме 2.1.5, и суммарное число обращений к эллиптической арифметике у нее ниже. Данный метод хорошо работает тогда, когда вычисление обратного в группе (или обращение знака) не представляет особого труда — в случае эллиптических кривых мы просто изменяем знак координаты $y$. (Заметим, что в дальнейшем мы продемонстрируем совершенно иной подход к схеме эллиптического умножения, см. алгоритм 7.2.7.)

**Алгоритм 7.2.4** (эллиптическое умножение: схема сложений—вычитаний). Этот алгоритм предполагает, что функции $double(), add(), sub()$ берутся либо из алгоритма 7.2.2, либо из 7.2.3, и выполняет эллиптическое умножение $[n]P$ для неотрицательного целого числа $n$ и точки $P \in E$. Предполагаем, что задано $B$-битовое двоичное представление числа $m = 3n$ в виде последовательности битов $(m_{B-1}, \ldots, m_0)$, а также соответствующее $B$-битовое представление $(n_j)$ числа $n$ (это представление дополнено нулями слева до $B$ битов), при этом полагаем $B = 0$ при $n = 0$.

1. [Инициализация]
   if($n == 0$) return $O$; // Бесконечно удаленная точка.
   $Q = P$;
2. [Сравнение битов чисел $3n, n$]
   for($B - 2 \geqslant j \geqslant 1$) {
   $Q = double(Q)$;
   if($(m_j, n_j) == (1, 0)$) $Q = add(Q, P)$;
   if($(m_j, n_j) == (0, 1)$) $Q = sub(Q, P)$;
   }
   return $Q$;

Задача о доказательстве правильности этого алгоритма встречается позднее в качестве упражнения 9.30. Увлекательная открытая область исследований связана с наилучшим способом построения схемы. В связи с этим см. упр. 9.77.

Перед тем как обсудить возможность (4) для осуществления эллиптической арифметики, рассмотрим необычайно полезную идею, влияние которой не ограничивается рамками возможности (4).

**Определение 7.2.5.** Если $E(F)$ есть эллиптическая кривая над полем $F$, заданная уравнением $y^2 = x^3 + Cx^2 + Ax + B$, и $g$ — ненулевой элемент поля $F$, то квадратичное искажение кривой $E$ посредством $g$ есть эллиптическая кривая над $F$, заданная уравнением $gy^2 = x^3 + Cx^2 + Ax + B$. В силу замены переменных $X = gx, Y = g^2y$, форма Вейерштрасса для этого искажения имеет вид $Y^2 = X^3 + gCX^2 + g^2AX + g^3B$.

Мы обнаружим, что при одних обстоятельствах будет удобно оставить кривую в виде $gy^2 = x^3 + Cx^2 + Ax + B$, при других же мы пожелаем воспользоваться равносильной формой Вейерштрасса.

Сразу замечаем, что если $g, h$ — ненулевые элементы поля $F$, то квадратичное искажение эллиптической кривой посредством $g$ дает группу, изоморфную квадратичному искажению этой кривой посредством $gh^2$. (В самом деле, просто положим новую переменную $Y$ равной $hy$. Чтобы понять, что эти группы изоморфны, достаточно просто взглянуть на формулы.) Таким образом, если $\mathbf{F}_q$ — конечное поле, в действительности существует только одно квадратичное искажение эллиптической кривой $E(\mathbf{F}_q)$, отличное от самой этой кривой. Это следует из того, что когда $g$ не является квадратом в поле $\mathbf{F}_q$, и $h$ пробегает ненулевые элементы поля $\mathbf{F}_q$, $gh^2$ пробегает все неквадраты. Это единственное нетривиальное квадратичное искажение кривой $E(\mathbf{F}_q)$ иногда обозначается через $E'(\mathbf{F}_q)$, в особенности когда нас не интересует, посредством какого неквадрата произведено это искажение.

А теперь — возможность (4), однородные координаты с опущенным «$Y$». Будем обсуждать ее с использованием искажения $gy^2 = x^3 + Cx^2 + Ax + B$ (см. определение 7.2.5). Начнем развивать эту идею в аффинных координатах. Пусть $P_1, P_2$ — аффинные точки на эллиптической кривой $E(F)$, и $P_1 \neq \pm P_2$. Используя определение 7.1.2 (обобщенное с учетом наличия «$g$»), можно записать выражения для $x_+, x_-$, т. е. $x$-координат точек $P_1 + P_2$ и $P_1 - P_2$, соответственно. Если перемножить эти выражения, можно увидеть, что $y$-координаты точек $P_1, P_2$ будут встречаться только в четных степенях, и значит, их можно заменить на $x$-выражения, пользуясь уравнением кривой $gy^2 = x^3 + Cx^2 + Ax + B$. Поразительно, что в получившемся выражении происходит множество сокращений, включая исчезновение параметра $g$. Данные уравнения приведены в следующем результате Монтгомери (см. [Montgomery 1987, 1992a]), здесь они обобщены на случай квадратичного искажения произвольной кривой, заданной уравнением (7.5).

**Теорема 7.2.6** (обобщенные тождества Монтгомери). *Пусть дана эллиптическая кривая $E$, определенная кубическим уравнением*

$$gy^2 = x^3 + Cx^2 + Ax + B,$$

*и две точки $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, обе отличные от $O$; обозначим через $x_\pm$ $x$-координаты точек $P_1 \pm P_2$, соответственно. Тогда если $x_1 \neq x_2$, то*

$$x_+x_- = \frac{(x_1x_2 - A)^2 - 4B(x_1 + x_2 + C)}{(x_1 - x_2)^2},$$

*а если* $x_1 = x_2$ *и* $2P_1 \neq O$, *то*

$$x_+ = \frac{(x_1^2 - A)^2 - 4B(2x_1 + C)}{4(x_1^3 + Cx_1^2 + Ax_1 + B)}.$$

Заметим, что в этой теореме $g$ несущественно в том смысле, что алгебра комбинированных $x$-координат не зависит от $g$. На самом деле, можно использовать $g$ лишь тогда, когда присутствует определенная начальная $y$-координата, но, конечно, основной упор в параметризации Монтгомери делается на то, чтобы игнорировать $y$-координаты. Вспомним, что случай $C = 0$ соответствует обычной форме Вейерштрасса (7.4). В то же время, как заметил Монтгомери, случай $B = 0$ особенно приятен. Например, получается простое соотношение

$$x_+ x_- = \frac{(x_1 x_2 - A)^2}{(x_1 - x_2)^2}.$$

Сейчас мы увидим, как соотношение такого сорта приводит к эллиптической алгебре с эффективными вычислениями.

Идея состоит в использовании такой цепи сложений для получения $[n]P$, в которой перед каждым сложением двух неравных точек $P_1, P_2$ получается так, что мы уже знаем, чему равно $P_1 - P_2$. Эта удивительная цепь строится с использованием цепи Люка, обсуждавшейся в разд. 3.6.3. В наших обозначениях на каждом промежуточном шаге из имеющейся пары $[k]P, [k+1]P$ мы будем образовывать либо пару $[2k]P, [2k+1]P$, либо пару $[2k+1]P, [2k+2]P$, в зависимости от битов числа $n$. В каждом из этих случаев выполняется одно удвоение и одно сложение. А при сложении нам уже известна разность между двумя слагаемыми, это сама точка $P$.

Чтобы обойтись без вычисления обратных по умножению, примем однородные координаты возможности (2), но опустим координату $Y$. Поскольку эти координаты однородные, когда у нас есть пара $[X : Z]$, фактически задано только отношение $X/Z$ (при $Z \neq 0$). Бесконечно удаленная точка распознается как пара $[0 : 0]$. Пусть даны точки $P_1, P_2$ с однородными координатами на эллиптической кривой, заданной уравнением (7.5), $P_1, P_2$ отличны от точки $O$, $P_1 \neq P_2$. Если

$$P_1 = [X_1, Y_1, Z_1],\ P_2 = [X_2, Y_2, Z_2],$$
$$P_1 + P_2 = [X_+, Y_+, Z_+],\ P_1 - P_2 = [X_-, Y_-, Z_-],$$

то на основании теоремы 7.2.6 нетрудно установить, что в случае $X_- \neq 0$ можно взять

$$\begin{aligned} X_+ &= Z_- \left((X_1X_2 - AZ_1Z_2)^2 - 4B(X_1Z_2 + X_2Z_1 + CZ_1Z_2)Z_1Z_2\right), \\ Z_+ &= X_-(X_1Z_2 - X_2Z_1)^2. \end{aligned} \tag{7.6}$$

Эти уравнения определяют пару $X_+, Z_+$ как функцию от шести величин $X_1, Z_1, X_2, Z_2, X_-, Z_-$, причем $Y_1, Y_2$ для нее абсолютно несущественны. Обозначим эту функцию через

$$[X_+ : Z_+] = addh([X_1 : Z_1], [X_2 : Z_2], [X_- : Z_-]),$$

«*h*» в названии функции подчеркивает однородность[2] каждой пары $[X : Z]$. Определение функции *addh* можно с легкостью распространить на все случаи, когда $X_-Z_- \neq 0$. Т. е. можно допустить, что одна из пар $[X_1 : Z_1]$, $[X_2 : Z_2]$ равна $[0 : 0]$. В частности, если $[X_1 : Z_1] = [0 : 0]$, а пара $[X_2 : Z_2]$ отлична от $[0 : 0]$, можно определить $addh([0 : 0], [X_2 : Z_2], [X_2 : Z_2])$ как $[X_2 : Z_2]$ (и в таком случае не использовать вышеуказанные уравнения). Аналогично поступаем, если $[X_2 : Z_2] = [0 : 0]$, а пара $[X_1 : Z_1]$ отлична от $[0 : 0]$. В случае когда $P_1 = P_2$, у нас есть функция удвоения

$$[X_+ : Z_+] = doubleh([X_1 : Z_1]),$$

где

$$\begin{aligned} X_+ &= \left(X_1^2 - AZ_1^2\right)^2 - 4B(2X_1 + CZ_1)Z_1^3, \\ Z_+ &= 4Z_1\left(X_1^3 + CX_1^2Z_1 + AX_1Z_1^2 + BZ_1^3\right). \end{aligned} \tag{7.7}$$

Функция *doubleh* работает во всех случаях, даже если $[X_1 : Z_1] = [0 : 0]$. Посмотрим, к примеру, как можно было бы вычислить координаты $[X : Z]$ для $[13]P$, где $P$ — точка на эллиптической кривой. Пусть, скажем, $[k]P = [X_k : Y_k]$. Имеем цепь сложений

$$[13]P = ([2]([2]P) + ([2]P + P)) + ([2]([2]P + P)),$$

которая выполняется следующим образом:

$$\begin{aligned} [X_2 : Z_2] &= doubleh([X_1 : Z_1]), \\ [X_3 : Z_3] &= addh([X_2 : Z_2], [X_1 : Z_1], [X_1 : Z_1]), \\ [X_4 : Z_4] &= doubleh([X_2 : Z_2]), \\ [X_6 : Z_6] &= doubleh([X_3 : Z_3]), \\ [X_7 : Z_7] &= addh([X_4 : Z_4], [X_3 : Z_3], [X_1 : Z_1]), \\ [X_{13} : Z_{13}] &= addh([X_7 : Z_7], [X_6 : Z_6], [X_1 : Z_1]). \end{aligned}$$

(Строго говоря, при этом мы должны предположить, что $X_1 \neq 0$.) В общем случае можно использовать следующий алгоритм, который, в сущности, содержит внутри себя алгоритм 3.6.7 для вычисления цепи Люка.

**Алгоритм 7.2.7** (эллиптическое умножение: метод Монтгомери). Этот алгоритм использует описанные выше функции $addh()$ и $doubleh()$ и предназначен для выполнения эллиптического умножения неотрицательного целого числа $n$ и точки $P = [X : \text{некоторое число} : Z]$ на кривой $E(F)$, где $XZ \neq 0$; в качестве результата возвращаются $[X : Z]$-координаты точки $[n]P$. Предполагаем, что задано $B$-битовое двоичное представление числа $n > 0$ в виде последовательности битов $(n_{B-1}, \ldots, n_0)$.

1. [Инициализация]
   if($n == 0$) return $O$; // Бесконечно удаленная точка.

[2] Буква «h» является первой буквой английского слова «homogeneous», что означает «однородный». — *Прим. перев.*

if($n == 1$) return $[X : Z]$; // Возвратить саму точку $P$.
if($n == 2$) return $doubleh([X : Z])$;

2. [Начало схемы сложений/удвоений Монтгомери]
$[U : V] = [X : Z]$; // Копировать координаты.
$[T : W] = doubleh([X : Z])$;

3. [Цикл по битам числа $n$, начиная со второго по старшинству]
for($B - 2 \geqslant j \geqslant 1$) {
if($n_j == 1$) {
$[U : V] = addh([T : W], [U : V], [X : Z])$;
$[T : W] = doubleh([T : W])$;
} else {
$[T : W] = addh([U : V], [T : W], [X : Z])$;
$[U : V] = doubleh([U : V])$;
}
}

4. [Окончательное вычисление]
if($n_0 == 1$) return $addh([U : V], [T : W], [X : Y])$;
return $doubleh([U : V])$;

Правила Монтгомери при $B = 0$ приводят к эффективному алгоритму, как показывает более простой вид функций $addh()$ и $doubleh()$. Более точно, каждую из функций $addh()$ и $doubleh()$ можно вычислить, затратив 9 умножений. В случае $B = 0, A = 1$ из каждой можно выбросить еще по одному умножению. Для других параметризаций также получаются неплохие результаты, и даже если эти вычисления не оптимальны в общем случае уравнения (7.4), они работают, так что процедуру эллиптического умножения во всей полноте (только без $y$-координат) для всех параметризаций, обсуждавшихся до сих пор, можно осуществлять с помощью алгоритма 7.2.7.

Было отмечено, что для получения аффинной $x$-координаты точки $[n]P$ нужно вычислить в поле величину $XZ^{-1}$. Если $n$ велико, это единичное вычисление обратного обходится, конечно, сравнительно недорого. Но иногда такое вычисление совсем не обязательно. К примеру, решая задачу разложения на множители, затронутую ниже, мы хотим выяснить, выполнено ли в группе эллиптической кривой равенство $[n]P = [m]P$; для этого нам достаточно будет проверить, обращается ли в нуль смешанное произведение $X_nZ_m - X_mZ_n$, и в этом вопросе опять можно обойтись без вычисления обратных. Существует еще один очень удобный факт подобного рода. Если некоторое кратное стало равно бесконечно удаленной точке, $[n]P = O$, то знаменатель $Z$ должен обратиться в нуль, и любые последующие кратные $[mn]P$ также будут иметь нулевой знаменатель $Z$. Поэтому нет необходимости искать точное значение кратного, если промежуточное кратное стало равно $O$. Равенство $Z = 0$ распространится и на результаты последующих применений функций эллиптического умножения.

Мы отмечали, что в алгоритме 7.2.7 обрабатываются только $x$-координаты кратных $[n]P$, и что неизвестность значений $y$ допустима в определенных практических реализациях. Сложить же две произвольные точки с однородны-

ми координатами при помощи описанного выше подхода нелегко, поскольку $y$-координаты исключены. Но потеряно не все — существует один полезный результат, который позволяет очень быстро определить, может ли сумма двух точек быть заданной третьей точкой. А именно, если заданы *только* $x$-координаты двух точек $P_1, P_2$, можно использовать следующий алгоритм для определения двух $x$-координат пары $P_1 \pm P_2$, хотя какая из этих координат будет соответствовать знаку «+», а какая — знаку «−», — неизвестно.

**Алгоритм 7.2.8** (сумма/разность без $y$-координат (Крэндалл)). **Для эллиптической кривой $E$, определяемой кубическим уравнением**

$$y^2 = x^3 + Cx^2 + Ax + B,$$

**нам заданы неравные $x$-координаты $x_1, x_2$ двух точек $P_1, P_2$, соответственно. Этот алгоритм возвращает квадратичный многочлен, чьими корнями являются (в неопределенном порядке) $x$-координаты точек $P_1 \pm P_2$.**

1. [Сформировать коэффициенты]
   $G = x_1 - x_2$;
   $\alpha = (x_1x_2 + A)(x_1 + x_2) + 2(Cx_1x_2 + B)$;
   $\beta = (x_1x_2 - A)^2 - 4B(x_1 + x_2 + C)$;
2. [Возвратить квадратичный многочлен]
   return $G^2X^2 - 2\alpha X + \beta$;
   // Этот многочлен обращается в нуль в точках $x_+$, $x_-$,
   // т. е. в $x$-координатах пары $P_1 \pm P_2$.

Оказывается, что дискриминант $4(\alpha^2 - \beta G^2)$ всегда будет квадратом в поле, так что если требуется конкретная пара $x$-координат точек $P_1 \pm P_2$, можно вычислить в поле величины

$$\left(\alpha \pm \sqrt{\alpha^2 - \beta G^2}\right) G^{-2}$$

и получить $x_+$, $x_-$, хотя, опять же, нельзя сказать, какой знак перед корнем соответствует конкретной координате (см. упр. 7.11). На основании этого алгоритма, таким образом, можно предложить тест, проверяющий, возможно ли выполнение какого-либо из равенств $P_3 = P_1 \pm P_2$ для множества из трех заданных точек с исключенными $y$-координатами. Данный тест полезен в определенных криптографических приложениях, таких как цифровая подпись (см. [Crandall 1996b]). Заметим, что отсутствующий в спецификации алгоритма случай $x_1 = x_2$ рассматривается непосредственно: одна из точек $P_1 \pm P_2$ есть $O$, а другая имеет $x$-координату, указанную в последней части теоремы 7.2.6.

Для получения дальнейшей информации об эллиптической арифметике см. [Cohen et al. 1998]. Вопрос об эффективных схемах вычисления кратного (возведения в степень) для эллиптической арифметики обсуждается позднее в разд. 9.3.

## 7.3. Теоремы Хассе, Дойринга и Ленстры

Одной из увлекательных и трудных задач является задача нахождения порядка группы эллиптической кривой над конечным полем, т. е. числа точек, включая $O$, на эллиптической кривой $E_{a,b}(F)$ при конечном поле $F$. Для поля $\mathbf{F}_p$, где простое число $p > 3$, мы можем сразу же выписать явное выражение для порядка $\#E$, если заметим, как мы это делали в простом алгоритме 7.2.1, что для пары $(x, y)$, которая является точкой, кубическая форма от $x$ должна быть квадратом в поле. Используя символ Лежандра, мы можем записать выражение

$$\#E\left(\mathbf{F}_p\right) = p + 1 + \sum_{x \in \mathbf{F}_p} \left(\frac{x^3 + ax + b}{p}\right), \tag{7.8}$$

представляющее собой нужное нам число точек $(x, y) \pmod p$, лежащих на такой кубической кривой $(\mathrm{mod}\ p)$, к которому, конечно, добавлена 1, соответствующая бесконечно удаленной точке. Это равенство можно обобщить на поля $\mathbf{F}_{p^k}$ следующим образом:

$$\#E\left(\mathbf{F}_{p^k}\right) = p^k + 1 + \sum_{x \in \mathbf{F}_{p^k}} \chi(x^3 + ax + b),$$

где $\chi$ есть квадратичный характер поля $\mathbf{F}_{p^k}$. (А именно, $\chi(u) = 1, -1, 0$, соответственно, в зависимости от того, является $u$ ненулевым квадратом в этом поле, неквадратом, или 0.) Знаменитый результат Х. Хассе состоит в следующем:

**Теорема 7.3.1** (Хассе). *Порядок $\#E$ кривой $E_{a,b}(\mathbf{F}_{p^k})$ удовлетворяет неравенству*

$$\left|(\#E) - (p^k + 1)\right| \leqslant 2\sqrt{p^k}.$$

Этот замечательный результат попадает в самое сердце теории эллиптических кривых, а значит, и ее приложений. Посмотрев на неравенство Хассе для поля $\mathbf{F}_p$, мы видим, что

$$p + 1 - 2\sqrt{p} < \#E < p + 1 + 2\sqrt{p}.$$

Существует интересная эвристическая взаимосвязь между этим неравенством и его альтернативой в виде соотношения (7.8). А именно, представьте себе символы Лежандра $\left(\frac{x^3+ax+b}{p}\right)$ «случайным блужданием», т. е. блужданием, которое описывается последовательностью подбрасываний монеты с исходами $\pm 1$, за исключением возможных символов $\left(\frac{0}{p}\right) = 0$. Из теории вероятностей известно, что ожидаемая абсолютная величина отклонения от нуля после суммирования $n$ таких случайных $\pm 1$ пропорциональна $\sqrt{n}$. Видно, что теорема Хассе дает «правильный» порядок величины отклонений от $p$ возможных порядков $\#E_{a,b}(\mathbf{F}_p)$. На более глубоком эвристическом уровне, однако, следует быть осторожным. Как мы упоминали в разд. 1.4.2, при действительно случайном блуждании ожидаемое отношение положения к величине $\sqrt{n}$ расходится примерно как $\ln \ln n$. Теорема Хассе утверждает, что этого не происходит —

указанное отношение ограничено числом 2. И в самом деле, существуют определенные тонкие нюансы статистики символов Лежандра, в которых обнаруживается отклонение от случайности (см. упр. 2.41)[3].

Чуть менее известна теорема, принадлежащая Дойрингу (см. [Deuring 1941]), утверждающая, что для каждого натурального числа $m \in (p+1-2\sqrt{p}, p+1+2\sqrt{p})$ существует некоторая пара $(a,b)$ из множества

$$\{(a,b) \ : \ a,b \in \mathbf{F}_p;\ 4a^3+27b^2 \neq 0\},$$

такая, что $\#E_{a,b}(\mathbf{F}_p) = m$. В действительности теорема Дойринга утверждает, что число кривых, с точностью до изоморфизма, которые имеют порядок $m$, есть так называемое кронекерово число классов для $(p+1-m)^2-4m$. В работе [Lenstra 1987] указанные результаты Хассе и Дойринга используются с целью получения некоторых сведений о статистике порядков кривых над заданным полем $\mathbf{F}_p$, к чему мы сейчас и переходим.

В приложениях, связанных с разложением на множители, с проверкой на простоту и с криптографией, мы занимаемся выбором случайной эллиптической кривой и затем задаем вопрос о вероятности того, что порядок этой кривой обладает некоторым специальным арифметическим свойством, т. е. является гладким, является легко разложимым на множители, является простым. Впрочем, существуют два возможных способа выбора случайной кривой. Один состоит в том, чтобы просто выбрать $a$, $b$ случайным образом и все. Но иногда нам хотелось бы иметь также случайную точку на этой кривой. При работе с настоящей эллиптической кривой над конечным полем точки на ней можно с легкостью находить посредством алгоритма 7.2.1. Но если мы работаем над кольцом $\mathbf{Z}_n$, где $n$ — составное, вызов процедуры извлечения квадратного корня в этом алгоритме вряд ли будет полезен. Тем не менее, можно совсем обойтись без алгоритма 7.2.1 и найти случайную кривую и точку на ней, выбрав точку еще до того, как кривая будет полностью определена! А именно, выбрать $a$ случайным образом, затем выбрать точку $(x_0,y_0)$ случайным образом, а затем выбрать $b$ так, чтобы точка $(x_0,y_0)$ лежала на кривой $y^2 = x^3+ax+b$, а именно, $b = y_0^2 - x_0^3 - ax_0$.

Используя эти два подхода к нахождению случайной кривой, мы можем формально выразить вопрос о вероятности того, что порядок кривой обладает заданным свойством. Пусть $p$ — простое число, большее 3, и $\mathcal{S}$ — множество целых чисел из интервала Хассе $(p+1-2\sqrt{p}, p+1+2\sqrt{p})$. Например, в качестве $\mathcal{S}$ можно взять множество $B$-гладких чисел из этого интервала при некотором выбранном нами значении $B$ (см. разд. 1.4.5), или взять в качестве $\mathcal{S}$ множество простых чисел из этого интервала, или множество удвоенных простых. Пусть $N_1(\mathcal{S})$ — число пар $(a,b) \in \mathbf{F}_p^2$ с условиями $4a^3+27b^2 \neq 0$ и $\#E_{a,b}(\mathbf{F}_p) \in \mathcal{S}$. Пусть, далее, $N_2(\mathcal{S})$ — число троек $(a,x_0,y_0) \in \mathbf{F}_p^3$, таких, что для $b = y_0^2 - x_0^3 - ax_0$ мы имеем $4a^3+27b^2 \neq 0$ и $\#E_{a,b}(\mathbf{F}_p) \in \mathcal{S}$. Что мы можем сказать относительно чисел $N_1(\mathcal{S}), N_2(\mathcal{S})$? Что касается первого числа, сначала существует $p^2$ возможностей выбора $a,b$, и каждое число $\#E_{a,b}(\mathbf{F}_p)$ попа-

[3] См. также *Преображенская Т. А.* О распределении значений $L$-рядов Дирихле: Дисс. канд. физ.-мат. наук, гл. 3. — М.: МГУ, 2006. — 64 с. — *Прим. ред.*

дает в интервал длины $4\sqrt{p}$, так что можно ожидать, что $N_1(\mathcal{S})$ будет порядка $\frac{1}{4}(\#\mathcal{S})p^{3/2}$. Аналогично, можно ожидать, что $N_2(\mathcal{S})$ будет порядка $\frac{1}{4}(\#\mathcal{S})p^{5/2}$. Т. е. в каждом случае ожидаемая вероятность того, что порядок кривой попадет в множество $\mathcal{S}$, примерно такая же, как вероятность того, что случайное целое число из интервала $(p+1-2\sqrt{p}, p+1+2\sqrt{p})$ попадет в множество $\mathcal{S}$. Следующая теорема утверждает, что примерно так и происходит.

**Теорема 7.3.2** (Ленстра). *Существует положительная постоянная $c$, такая, что если простое число $p > 3$, и $\mathcal{S}$ — множество целых чисел из интервала $(p+1-2\sqrt{p}, p+1+2\sqrt{p})$, состоящее не менее, чем из 3-х элементов, то*

$$N_1(\mathcal{S}) > c(\#\mathcal{S})p^{3/2}/\ln p, \quad N_2(\mathcal{S}) > c(\#\mathcal{S})p^{5/2}/\ln p.$$

Эта теорема доказана в работе [Lenstra 1987], где также приведены верхние оценки, примерно такие же по порядку величины, как и нижние.

## 7.4. Метод эллиптических кривых

Один из субэкспоненциальных методов разложения на множители, обладающий величайшим изяществом и практической ценностью — это метод эллиптических кривых (elliptic curve method — ECM) Х. Ленстры. В его изяществе мы вскоре убедимся. Его практическую ценность выражает то обстоятельство, что в отличие от QS или NFS, сложность разложения числа $n$ при помощи ECM в основном зависит от величины наименьшего простого делителя числа $n$, и лишь в незначительной степени зависит от величины самого числа $n$. По этой причине в последние годы были открыты многие делители поистине гигантских чисел, многие из которых лежат далеко за пределами возможностей QS или NFS.

Позже в этом разделе мы продемонстрируем некоторые конкретные современные достижения ECM, которые подтверждают значительную мощь этого метода.

### 7.4.1. Базовый алгоритм ECM

Этот алгоритм ECM использует многие из тех концепций эллиптической арифметики, которые были развиты в предыдущих разделах. Однако мы намерены применять эту арифметику к объекту $E_{a,b}(\mathbf{Z}_n)$, представляющему собой нечто отличное от настоящей эллиптической кривой, когда $n$ — составное число.

**Определение 7.4.1.** Для элементов $a, b$ кольца $\mathbf{Z}_n$ с условием $\text{НОД}(n, 6) = 1$ и условием на дискриминант $\text{НОД}(4a^3 + 27b^2, n) = 1$ эллиптическая псевдокривая над этим кольцом есть множество

$$E_{a,b}(\mathbf{Z}_n) = \{(x, y) \in \mathbf{Z}_n \times \mathbf{Z}_n : y^2 = x^3 + ax + b\} \cup \{O\},$$

где $O$ — бесконечно удаленная точка. (Так что эллиптическая кривая над полем $\mathbf{F}_p = \mathbf{Z}_p$ из определения 7.1.1 есть также эллиптическая псевдокривая.)

(Кривые, заданные в виде (7.5), также рассматриваются как псевдокривые при выполнении соответствующего условия на дискриминант.) В разд. 7.1 мы видели, что при простом $n$ бесконечно удаленная точка означает единственную дополнительную проективную точку на нашей кривой, которой не соответствует никакая аффинная точка. При составном $n$ существуют и другие проективные точки, не соответствующие аффинным точкам, но в нашем определении псевдокривой мы, несмотря на это, допускаем только одну дополнительную точку, ту, которая соответствует проективному решению $[0,1,0]$. Из-за того, что мы так (преднамеренно) обсчитались в нашем определении, псевдокривая $E_{a,b}(\mathbf{Z}_n)$ с операциями определения 7.1.2 не образует группу (при составном $n$). В частности, существуют пары точек $P, Q$, для которых сумма «$P+Q$» не определена. Это обусловлено строением углового коэффициента $m$ в определении 7.1.2; поскольку $\mathbf{Z}_n$ не является полем при составном $n$, можно столкнуться с тем, что операция взятия обратного применяется к ненулевому элементу $\mathbf{Z}_n$, который не является обратимым. Этим нарушением аксиом группы и обусловлено название «псевдокривая», но, к счастью, существуют мощные приложения понятия псевдокривой. В частности, алгоритм 2.1.4 (расширенный алгоритм Евклида), если его вызвать для нахождения обратного к ненулевому элементу $\mathbf{Z}_n$, не являющемуся на самом деле обратимым, вместо этого возвратит нетривиальный делитель числа $n$. Оригинальная идея Ленстры состоит именно в том, что эта невозможность вычисления обратного позволяет нам разложить на множители составное число $n$.

Попутно заметим, что в понятии эллиптического умножения на псевдокривой присутствует зависимость от использованной цепи сложений. К примеру, точка $[5]P$ может прекрасно вычисляться, если мы вычисляем ее посредством цепи $P \to [2]P \to [4]P \to [5]P$, но предпоследнее эллиптическое сложение может оказаться невыполнимым, если мы попытаемся вычислить эту точку посредством цепи $P \to [2]P \to [3]P \to [5]P$. Тем не менее, если две различные цепи сложений для получения $[k]P$ выполнимы, то они дадут один и тот же результат.

**Алгоритм 7.4.2** (метод эллиптических кривых (elliptic curve method — ECM) Ленстры). Получив на входе составное число $n$, которое требуется разложить на множители, причем $\text{НОД}(n,6)=1$, и $n$ не является точной степенью, этот алгоритм пытается обнаружить нетривиальный делитель числа $n$. Имеется настраиваемый параметр $B_1$, называемый «пределом первой стадии», что подразумевает наличие дальнейших стадий современного алгоритма ECM, которые могут идти вслед за этим.

1. [Выбрать предел $B_1$]
   $B_1 = 10000$; // Или любое другое подходящее начальное значение // «предела первой стадии» $B_1$.
2. [Найти кривую $E_{a,b}(\mathbf{Z}_n)$ и точку $(x,y) \in E$]
   Выбрать случайные $x, y, a \in [0, n-1]$;
   $b = (y^2 - x^3 - ax) \bmod n$;
   $g = \text{НОД}(4a^3 + 27b^2, n)$;

if($g == n$) goto [Найти кривую ...];
if($g > 1$) return $g$; // Делитель найден.
$E = E_{a,b}(\mathbf{Z}_n)$; // Эллиптическая псевдокривая
$P = (x, y)$; // и точка на ней.

3. [Домножать на степени простых]
for($1 \leqslant i \leqslant \pi(B_1)$) { // Цикл по простым числам $p_i$.
Найти наибольшее натуральное число $a_i$, такое что $p_i^{a_i} \leqslant B_1$;
for($1 \leqslant j \leqslant a_i$) { // $j$ есть просто счетчик.
$P = [p_i]P$, с выходом из функций эллиптической алгебры, если вычисление некоторого $d^{-1}$ для знаменателя $d$ углового коэффициента при сложении сигнализирует о нетривиальном $g = \text{НОД}(n, d)$. В таком случае возвратить $g$;
// Делитель найден.
}
}

4. [Неудача]
Возможно, увеличить $B_1$; // См. в тексте.
goto [Найти кривую ...];

В базовом алгоритме ECM мы рассчитываем на то, что хотя составное число $n$ допускает лишь псевдокривую, незаконная эллиптическая операция — в данном случае вычисление обратного, необходимое для вычисления углового коэффициента из определения 7.1.2 — является сигналом того, что для некоторого простого $p|n$ справедливо соотношение

$$[k]P = O, \text{ где } k = \prod_{p_i^{a_i} \leqslant B_1} p_i^{a_i},$$

и это соотношение выполнено для настоящей эллиптической кривой $E_{a,b}(\mathbf{F}_p)$. Кроме того, нам известно, что по теореме Хассе 7.3.1 порядок $\#E_{a,b}(\mathbf{F}_p)$ лежит в интервале $(p + 1 - 2\sqrt{p}, p + 1 + 2\sqrt{p})$. Очевидно, мы можем ожидать обнаружение делителя, если множитель $k$ делится на $\#E(\mathbf{F}_p)$, что, фактически, происходит, если *этот порядок является $B_1$-гладким.* (Здесь мы допускаем некоторую неточность, поскольку для того, чтобы порядок был $B_1$-гладким требуется лишь, чтобы каждый из его простых делителей не превосходил $B_1$, а в приведенном тексте алгоритма у нас стоит более сильное условие, что каждый делитель порядка, являющийся степенью простого, не превосходит $B_1$. Мы могли бы изменить неравенство для определения $a_i$ на $p_i^{a_i} \leqslant n + 1 + 2\sqrt{n}$, но на практике связанные с этим затраты слишком велики в сравнении с той незначительной выгодой, которую это может принести.) Так что мы будем воспринимать предел первой стадии $B_1$ как границу гладкости настоящих порядков кривых, причем множество порядков определяется неизвестным простым делителем $p$.

Полезно сравнить алгоритм ECM с $(p - 1)$-методом Полларда (алгоритм 5.4.1). В $(p - 1)$-методе имеется только одна группа $\mathbf{Z}_p^*$ (порядка $p - 1$), и мы достигаем цели, если порядок этой группы является $B$-гладким. В алго-

ритме ECM имеется целое семейство групп эллиптических кривых, из которого осуществляется случайный выбор, и каждая новая группа дает новые шансы на успех.

Используя эти идеи, мы можем провести эвристическую оценку сложности ECM. Предположим, что число $n$, которое требуется разложить на множители, является составным, взаимно просто с числом 6 и не является точной степенью. Пусть $p$ обозначает наименьший простой делитель числа $n$, и $q$ обозначает простой делитель числа $n$, отличный от $p$. Алгоритму 7.4.2 удастся разложить $n$, если мы выбрали $a, b, P$ на шаге [Найти кривую ...], и для некоторого значения $k$ вида

$$k = p_l^a \prod_{i<l} p_i^{a_i},$$

где $l \leqslant \pi(B_1)$ и $a \leqslant a_l$, мы имеем

$$[k]P = O \text{ на кривой } E_{a,b}(\mathbf{F}_p), \quad [k]P \neq O \text{ на кривой } E_{a,b}(\mathbf{F}_q).$$

Вероятность одновременного наступления этих двух событий в основном определяется вероятностью первого, так что вторым событием можно пренебречь. Как мы упоминали выше, первое событие наступает, если порядок $\#E_{a,b}(\mathbf{F}_p)$ является $B_1$-гладким. По теореме 7.3.2 вероятность успеха $prob(B_1)$ больше, чем

$$c\frac{\psi(p+1+2\sqrt{p}, B_1) - \psi(p+1-2\sqrt{p}, B_1)}{\sqrt{p}\ln p}.$$

Здесь $\psi(x,y)$ обозначает то же самое, что и в (1.42). Поскольку требуется порядка $B_1$ арифметических операций для выполнения проверки одной кривой на шаге [Домножать на степени простых], нам хотелось бы выбрать $B_1$ так, чтобы минимизировать выражение $B_1/prob(B_1)$. В предположении, что $prob(B_1)$ есть величина порядка

$$c\,\frac{\psi(\frac{3}{2}p, B_1) - \psi(\frac{1}{2}p, B_1)}{p\ln p},$$

чтобы можно было использовать оценки, обсуждавшиеся в разд. 1.4.5, мы получаем, что этот минимум достигается при

$$B_1 = \exp\left((\sqrt{2}/2 + o(1))\sqrt{\ln p \ln\ln p}\right),$$

и для данного значения $B_1$ оценка сложности $B_1/prob(B_1)$ имеет следующий вид:

$$\exp\left((\sqrt{2} + o(1))\sqrt{\ln p \ln\ln p}\right)$$

(см. упр. 7.12). Разумеется, изначально $p$ неизвестно, так что выбор подходящего начального значения $B_1$ на шаге [Выбрать предел $B_1$] может быть лишь прогнозом. По этой причине наш алгоритм предписывает начать с нижней границы значения $B_1$, равной 10000, а затем, возможно, увеличить это значение на шаге [Неудача]. На практике поступают так: с одним значением $B_1$ работают до тех пор, пока не приходят к убеждению, что требуется увеличение

этого значения, например, в два раза, и затем производят следующую итерацию этой процедуры. Разумеется, на шаге [Неудача] можно прервать работу и полностью отказаться от попытки разложения на множители. При постепенном увеличении значения $B_1$ в алгоритме ECM можно ожидать успеха, когда $B_1$ достигнет указанного интервала экстремальных значений, и время, затраченное на меньшие значения $B_1$, дает незначительный вклад в общее время работы.

Подводя итог можно сказать, что ожидаемая эвристическая сложность алгоритма ECM при нахождении нетривиального разложения числа $n$ с наименьшим простым делителем $p$ есть $L(p)^{\sqrt{2}+o(1)}$ арифметических операций с целыми числами порядка $n$ (в обозначениях (6.1)). (Напомним, что «$o(1)$» обозначает величину, стремящуюся к 0 (при $p$, стремящемся к бесконечности).) Так что чем больше наименьший простой делитель числа $n$, тем больше ожидаемое число арифметических операций. В самом худшем случае, когда $n$ является произведением двух простых чисел одного порядка, ожидаемое число операций имеет вид $L(n)^{1+o(1)}$, что в точности совпадает с эвристической сложностью метода квадратичного решета (см. разд. 6.1.1). Однако из-за того, что типичная операция в алгоритме ECM требует большего количества разрядов, для чисел, соответствующих худшему случаю, обычно предпочитают использовать метод QS или метод NFS.

Если нам предоставили число $n$, про которое заранее не известно, что оно соответствует худшему случаю, рекомендуется сначала попытаться использовать ECM, и лишь после того, как этот метод отработает изрядное количество времени, следует запустить QS или NFS. Но если это число $n$ велико настолько, что можно заранее исключить из рассмотрения QS и NFS, то в данной ситуации ECM остается единственным возможным решением. Кто знает, может быть нам повезет!

«Везение» здесь может быть двоякого рода: или рассматриваемое число в самом деле имеет настолько малый простой делитель, что его можно обнаружить при помощи ECM, или при выполнении алгоритма ECM удачный выбор параметров произойдет раньше, чем мы того ожидаем, и будет обнаружен впечатляющий делитель. На самом деле, одной интересной особенностью алгоритма ECM является то, что разброс ожидаемого количества операций весьма велик, поскольку нам достаточно наступления лишь одного благоприятного события.

Интересно, что эвристическая оценка сложности алгоритма ECM может быть сделана абсолютно строгой, за исключением предположения о том, что целые числа из интервала Хассе с такой же вероятностью являются гладкими, как и типичные целые числа из более широкого интервала $(p/2, 3p/2)$ (см. [Lenstra 1987]).

Далее мы обсудим некоторые оптимизации алгоритма ECM. Эти улучшения не оказывают ощутимого влияния на оценку сложности. Но они действительно оказываются весьма полезными на практике.

### 7.4.2. Оптимизации алгоритма ECM

Как и в $(p-1)$-методе Полларда (см. разд. 5.4), обобщением которого является ECM, существует естественное продолжение этого метода в виде второй стадии. Принимая во внимание те замечания, которые были сделаны вслед за алгоритмом 7.4.2, предположим, что порядок $\#E_{a,b}(\mathbf{F}_p)$ не является $B_1$-гладким, какой бы разумный выбор значения $B_1$ мы не делали, так что, судя по всему, нашему базовому алгоритму не удастся обнаружить делитель. Но может случиться так, что

$$\#E(\mathbf{F}_p) = q \prod_{p_i^{a_i} \leqslant B_1} p_i^{a_i},$$

где $q$ — простое число, превосходящее $B_1$. Когда одно выходящее за предел простое число является частью разложения неизвестного порядка на множители, не нужно умножать текущую точку на *каждое* простое число из интервала $(B_1, q]$. Вместо этого можно использовать точку

$$Q = \left[ \prod_{p_i \leqslant B_1} p_i^{a_i} \right] P,$$

которая, по существу, «остается в живых» после первой стадии алгоритма ECM 7.4.2, и проверить точки

$$[q_0]Q,\ [q_0 + \Delta_0]Q,\ [q_0 + \Delta_0 + \Delta_1]Q,\ [q_0 + \Delta_0 + \Delta_1 + \Delta_2]Q, \ldots,$$

где $q_0$ — наименьшее простое, превосходящее $B_1$, а $\Delta_i$ — разности между соседними простыми числами после $q_0$. Идея состоит в том, что некоторые точки

$$R_i = [\Delta_i]Q$$

можно *сохранить* для многократного использования, а затем быстро проверить наши простые числа, большие $B_1$, путем последовательных эллиптических сложений с использованием соответствующих $R_i$. Основным преимуществом здесь является то, что умножение точки на простое число $q$ требует $O(\ln q)$ эллиптических операций, в то время как сложение с использованием предварительно вычисленной точки $R_i$ есть, очевидно, одна операция.

Кроме этой оптимизации в виде «второй стадии» и ее вариантов можно задействовать и другие оптимизации, такие, как

(1) Специальная параметризация, чтобы с легкостью получать случайные кривые.
(2) Выбор кривых, про порядок которых известно, что он делится на 12 или 16 (см. [Montgomery 1992a], [Brent et al. 2000]).
(3) Оптимизации арифметики больших целых чисел и собственно эллиптической алгебры, например, с помощью быстрого преобразования Фурье (БПФ).
(4) Быстрые алгоритмы, примененные ко второй стадии, такие, как «расширенное БПФ», которое, в сущности, представляет собой схему вычисле-

ния значений многочлена, примененную к множествам предварительно вычисленных $x$-координат.

Вместо того, чтобы подробно изучать эти оптимизации, каждый раз приводя текст соответствующего алгоритма, мы пойдем другим путем: кратко обсудим их, а затем приведем один-единственный практический алгоритм, содержащий многие из этих оптимизаций.

Что касается указанной выше оптимизации (1), поразительной особенностью нашего окончательного алгоритма будет то, что $y$-координаты в нем вообще не будут участвовать. А именно, наш алгоритм будет использовать параметризацию Монтгомери

$$gy^2 = x^3 + Cx^2 + x$$

с эллиптическим умножением, осуществляемым посредством алгоритма 7.2.7. Так что точка будет иметь однородный общий вид $P = [X, \text{некоторое число}, Z] = [X : Z]$ (см. разд. 7.2, в котором обсуждаются эти обозначения), и нам необходимо отслеживать лишь вычеты $X, Z \pmod n$. Как мы упоминали вслед за алгоритмом 7.2.7, на появление бесконечно удаленной точки $O$ в процессе вычислений на кривой над полем $\mathbf{F}_p$, где $p|n$, указывает обращение в нуль знаменателя $Z$, и этот знаменатель так и останется равным нулю при следующих вызовах функций *addh*() и *doubleh*(). Таким образом, рассматриваемая параметризация позволяет нам каждый раз проверять значение НОД$(n, Z)$, и если когда-либо это значение станет больше 1, то это, скорее всего, и есть неизвестный делитель $p$. На практике мы «аккумулируем» $Z$-координаты и вычисляем НОД лишь изредка, например, первый раз — после первой стадии и, как мы далее увидим, второй и последний раз — после второй стадии.

Что касается оптимизации (2), Суяма заметил, что в условиях параметризации Монтгомери порядок группы $\#E$ делится на 4. Но можно пойти дальше и обеспечить делимость этого порядка на 8, 12, и даже на 16. Так, применительно к оптимизации (2) мы можем использовать один удобный результат (см. [Brent et al. 2000]).

**Теорема 7.4.3** (выбор кривой для алгоритма ECM). *Определим эллиптическую кривую $E_\sigma(\mathbf{F}_p)$ кубическим уравнением*

$$y^2 = x^3 + C(\sigma)x^2 + x,$$

*где зависимость $C$ от принадлежащего полю параметра $\sigma \neq 0, 1, 5$ выражается формулами*

$$\begin{aligned} u &= \sigma^2 - 5, \\ v &= 4\sigma, \\ C(\sigma) &= \frac{(v-u)^3(3u+v)}{4u^3v} - 2. \end{aligned}$$

*Тогда порядок кривой $E_\sigma$ делится на 12, и кроме того, либо на кривой $E$, либо на ее искажении $E'$ (см. определение 7.2.5) существует точка, $x$-координата которой равна $u^3v^{-3}$.*

Теперь каждую попытку использования новой кривой можно запустить, просто выбрав случайное $\sigma$. Затем мы используем алгоритм 7.2.7, начав с однородных $x$-координат в виде $X/Z = u^3/v^3$ и игнорируя в процессе работы алгоритма разложения на множители все $y$-координаты. Более того, мы даже не заботимся о том, где лежит начальная точка — на кривой $E$ или на ее искажении, вновь благодаря тому, что $y$-координаты можно не знать.

Что касается оптимизации (3), существуют идеи, которые позволяют уменьшить количество вычислений на второй стадии. Один из трюков, который применяют некоторые исследователи, состоит в использовании второй стадии с «парадоксом дней рождения», а именно, в использовании «полуслучайных» кратных для двух множеств координат, что иногда дает выигрыш в быстродействии (см. [Brent et al. 2000]). Но существуют также некоторые идеи, которые применяют, действуя по сценарию обычной проверки всех выходящих за предел простых чисел $q$ вплоть до некоторого «предела второй стадии» $B_2 > B_1$ без использования каких-либо специальных схем для сравнения списков. Следующий весьма практичный метод сокращает вычислительные затраты асимптотически всего лишь до двух (и менее) умножений $(\mathrm{mod}\ n)$ на каждое выходящее за предел простое число. Выше мы уже обсуждали, что если $q_n, q_{n+1}$ — последовательные простые числа, то можно прибавлять некоторое сохраненное кратное $[\Delta_n]Q$ к текущему значению $[q_n]Q$, получая следующую точку $[q_{n+1}]Q$, что составляет всего лишь одну эллиптическую операцию на каждое простое число $q_m$. Это уже впечатляет, однако вспомним, что эллиптическая операция представляет собой, по сути, некоторое количество умножений $(\mathrm{mod}\ n)$. Мы можем легко (и достаточно сильно) понизить сложность следующим образом. Пусть для некоторого простого числа $r$ нам известно кратное $[r]Q = [X_r : Z_r]$, и у нас в руках предварительно вычисленное и сохраненное множество кратных $[\Delta]Q = [X_\Delta : Z_\Delta]$ для разностей $\Delta$, где $\Delta$ пробегало некоторое относительно малое конечное множество $\{2, 4, 6, \ldots\}$. Тогда проверку выходящего за предел простого числа $s$, которое ненамного больше $r$, можно провести, замечая, что о «точном попадании»

$$[s]Q = [r + \Delta]Q = O$$

можно судить по тому, что смешанное произведение

$$X_r Z_\Delta - X_\Delta Z_r$$

имеет нетривиальный НОД с числом $n$. Так, с достаточным количеством кратных $[\Delta]Q$ и несколькими дополнительными точками $[r]Q$ в руках мы можем проверять выходящие за предел простые числа и тратить 3 умножения $(\mathrm{mod}\ n)$ на одного кандидата. В самом деле, кроме 2-х умножений на смешанное произведение нам необходимо аккумулировать произведение $\prod(X_r Z_\Delta - X_\Delta Z_r)$, чтобы в итоге вычислить НОД этого произведения с числом $n$. Но можно еще более сократить работу, замечая, что

$$X_r Z_\Delta - X_\Delta Z_r = (X_r - X_\Delta)(Z_r + Z_\Delta) + X_\Delta Z_\Delta - X_r Z_r.$$

Так, можно сохранить предварительно вычисленные значения $X_\Delta, Z_\Delta, X_\Delta Z_\Delta$ и использовать отдельные значения $X_r, Z_r, X_r Z_r$ для надлежащим образом от-

стоящих простых чисел $r$, что уменьшает стоимость второй стадии асимптотически до 2-х умножений (mod $n$) на выходящее за предел простое число, одного — на правую часть указанного выше тождества, и одного — на аккумулирование.

Как показано в работе [Brent et al. 2000] существуют и другие трюки, которые подобным образом сокращают затраты на вторую стадию ECM. Один из них также относится к оптимизации (3) и состоит в добавлении к разнообразным тождествам концепции умножения на основе преобразования (см. разд. 9.5.3). Эти методы наиболее уместны, когда $n$ достаточно велико, иными словами, когда $n$ находится в той области, где умножение на основе преобразования опережает «школьное» умножение. В вышеупомянутое тождество для смешанных произведений на самом деле можно подставлять хранимые преобразования (например, дискретные преобразования Фурье (ДПФ))

$$\hat{X}_r, \hat{Z}_r,$$

и в таком случае произведение $(X_r - X_\Delta)(Z_r + Z_\Delta)$ теперь будет требовать лишь 1/3 умножения (на основе преобразования). Это поразительное сокращение возможно благодаря тому, что отдельное произведение указанного вида будет вычисляться в спектральном пространстве и потому асимптотически ничего не стоит, обратное же преобразование дает вклад 1/3. Подобные рассуждения применимы и к аккумулированию произведений. На этом пути количество умножений может быть снижено до 1 на каждое выходящее за предел простое число. Эллиптическая арифметика сама по себе также допускает оптимизацию на основе преобразований. В условиях рассматриваемой параметризации Монтгомери соответствующие функции арифметики на кривой достаточно вырождены и задаются уравнениями (7.6) и (7.7). И вновь умножение на основе преобразования позволяет сократить количество умножений, необходимых для функции *addh*(), с 6-и до 4-х умножений на основе преобразования, с возможностью подобного сокращения для *doubleh*() (см. замечания после алгоритма 7.4.4).

Что касается оптимизации (4), схема Монтгомери для вычисления значений многочлена (иногда называемая «расширенным БПФ» из-за того, что в ней специальным образом вычисляются значения больших многочленов посредством БПФ) на второй стадии имеет в своей основе вычисление двух множеств точек:

$$S = \{[m_i]P : i = 1, \ldots, d_1\}, \quad T = \{[n_j]P : j = 1, \ldots, d_2\},$$

где $P$ — точка, остающаяся после первой стадии ECM, $d_1|d_2$, а целые числа $m_i, n_j$ выбираются специальным образом в надежде на то, что некоторая комбинация $m_i \pm n_j$ делит (единственное) выходящее за предел простое число $q$. Об этом благоприятном событии, в свою очередь, будет свидетельствовать тот факт, что некоторой $x$-координате из списка $S$ соответствует некоторая $x$-координата из списка $T$ в том смысле, что их разность имеет нетривиальный НОД с числом $n$. В другом контексте с задачей сравнения списков мы встретимся при подготовке к алгоритму 7.5.1. Поскольку алгоритм 7.5.1 может требовать слишком много машинной памяти на сортировку и т. д., можно

поступить так: определить многочлен степени $d_1$

$$f(x) = \prod_{s \in S} (x - X(s)) \bmod n,$$

где функция $X(\ )$ возвращает аффинную $x$-координату точки. Затем можно вычислить значения этого многочлена в $d_2$ точках $x \in \{X(t) : t \in T\}$. Или можно вычислить НОД многочлена $f(x)$ и многочлена $g(x) = \prod_t (x - X(t))$. В каждом из этих случаев можно найти соответствия между множествами точек $S$, $T$ за $O\left(d_2^{1+\epsilon}\right)$ операций в кольце вычетов, что выгодно отличается от выполнения $d_1 d_2$ сравнений. Кстати, идее Монтгомери предшествовал подход из [Montgomery and Silverman 1990], относящийся к расширениям $(p-1)$-метода Полларда.

При вызове подобных средств для эффективных вычислений на второй стадии придерживаются эмпирического правила, которое гласит, что на вторую стадию следует тратить лишь определенную долю (скажем, от $1/4$ до $1/2$, в зависимости от многих обстоятельств) общего времени. Это правило возникло в среде современных пользователей ECM, а его обоснование кроется в машинозависимой сложности разнообразных операций на стадиях. На практике все это означает, что предел второй стадии должен быть примерно на два порядка величины больше предела первой стадии, или

$$B_2 \approx 100 B_1.$$

Это хорошее практическое правило, которое легко и эффективно уменьшает количество степеней свободы, связанных с ECM в целом. Теперь время на обработку одной кривой (при наличии обеих стадий) есть функция, зависящая только от $B_1$. В дополнение к этому существуют разнообразные таблицы предполагаемых хороших значений $B_1$ в терминах «ожидаемых» размеров неизвестных делителей числа $n$ (см. [Silverman and Wagstaff 1993], [Zimmermann 2000]).

Сейчас мы продемонстрируем расширенный алгоритм ECM в том виде, в котором ему принадлежат определенные достижения, и в котором он сейчас широко используется. Несмотря на то что здесь представлены не все возможные оптимизации, мы попытались отразить многие из вышеупомянутых приемов. Их здесь вполне достаточно для построения серьезной практической реализации. В представленный ниже вариант ECM вошли разнообразные оптимизации, принадлежащие Бренту, Крэндаллу, Монтгомери, Вольтману и Циммерману.

**Алгоритм 7.4.4** (алгоритм ECM без вычисления обратных). **Получив составное число $n$ для разложения на множители, причем $\text{НОД}(n, 6) = 1$, данный алгоритм пытается обнаружить нетривиальный делитель числа $n$. Алгоритм обходится без вычисления обратных по модулю $n$ и нуждается лишь в умножении по модулю для больших целых чисел (см., однако, текст, следующий за алгоритмом).**

1. [Выбрать критерии]
   $B_1 = 10000$; // Предел первой стадии (должен быть четным).
   $B_2 = 100 B_1$; // Предел второй стадии (должен быть четным).

$D = 100$; // Общий объем занимаемой памяти — около $3D$ // целых чисел порядка $n$.

2. [Выбрать случайную кривую $E_\sigma$]
   Выбрать случайное $\sigma \in [6, n-1]$; // См. теорему 7.4.3.
   $u = (\sigma^2 - 5) \bmod n$;
   $v = 4\sigma \bmod n$;
   $C = ((v-u)^3(3u+v)/(4u^3v) - 2) \bmod n$;
   // Замечание: $C$ определяет кривую $y^2 = x^3 + Cx^2 + x$,
   // однако, $C$ можно хранить в виде числитель/знаменатель.
   $Q = [u^3 \bmod n : v^3 \bmod n]$; // Начальная точка представлена // как $[X : Z]$.
3. [Выполнить первую стадию]
   for($1 \leqslant i \leqslant \pi(B_1)$) { // Цикл по простым числам $p_i$.
      Найти наибольшее целое число $a$, такое, что $p_i^a \leqslant B_1$;
      $Q = [p_i^a]Q$; // Посредством алгоритма 7.2.7,
      // возможно используя оптимизации на основе БПФ
      // (см. последующий текст).
   }
   $g = \text{НОД}(Z(Q), n)$; // Точка имеет вид $Q = [X(Q) : Z(Q)]$.
   if($1 < g < n$) return $g$; // Возвратить нетривиальный делитель числа $n$.
4. [Перейти ко второй стадии] // Вторая стадия без вычисления обратных.
   $S_1 = doubleh(Q)$;
   $S_2 = doubleh(S_1)$;
   for($d \in [1, D]$) { // В этом цикле вычисляются $S_d = [2d]Q$.
      if($d > 2$) $S_d = addh(S_{d-1}, S_1, S_{d-2})$;
      $\beta_d = X(S_d)Z(S_d) \bmod n$; // Также сохранить произведения $XZ$.
   }
   $g = 1$;
   $B = B_1 - 1$; // $B$ нечетно.
   $T = [B - 2D]Q$; // Посредством алгоритма 7.2.7.
   $R = [B]Q$; // Посредством алгоритма 7.2.7.
   for($r = B$; $r < B_2$; $r = r + 2D$) {
      $\alpha = X(R)Z(R) \bmod n$;
      for(для простых $q \in [r+2, r+2D]$) { // Цикл по простым числам.
         $\delta = (q - r)/2$; // Расстояние до текущего простого.
      // Следующий шаг допускает оптимизацию на основе преобразования.
         $g = g((X(R) - X(S_\delta))(Z(R) + Z(S_\delta)) - \alpha + \beta_\delta) \bmod n$;
      }
      $(R, T) = (addh(R, S_D, T), R)$;
   }
   $g = \text{НОД}(g, n)$;
   if($1 < g < n$) return $g$; // Возвратить нетривиальный делитель числа $n$.
5. [Неудача]
   goto [Выбрать случайную кривую ...]; // Или увеличить // пределы $B_1$, $B_2$ и т.п.

В предложенной здесь частной реализации второй стадии используются $D$ кратных точек $[2d]Q$ для соответствующих разностей и хранимое произведение $XZ$ для каждой такой точки, что в сумме дает $3D$ хранимых целых чисел порядка $n$. Представленная схема второй стадии асимптотически (по существу, для больших $n$ и больших параметров памяти $D$) тратит два умножения по модулю $n$ на каждое выходящее за предел простое число, и может тратить еще меньше, если допустить вычисления обратных элементов для больших целых чисел (а представленный алгоритм обходится без них) в процессе второй стадии. Также, вероятно, неэкономно каждый раз заново вычислять выходящие за предел простые числа при каждом выборе эллиптической кривой. Если достаточно памяти, все эти простые числа можно предварительно вычислить при помощи решета на шаге [Выбрать критерии]. Еще одна оптимизация, которую мы не разъяснили в алгоритме, заключается в том, что когда мы проверяем, имеет ли смешанное произведение $XZ' - X'Z$ нетривиальный НОД с числом $n$, у нас, фактически, проверяются комбинации вида $P \pm P'$, поскольку любая точка со знаком «+» и «−» имеет одну и ту же $x$-координату. Это означает, что если два простых числа равноудалены от «центрального значения» $r$, по сути, $q', r, q$ образуют арифметическую прогрессию, то при проверке одного смешанного произведения на самом деле обрабатываются оба простых числа.

Оформляя практический вариант ECM в виде алгоритма 7.4.4, в определенный момент мы вынуждены были остановиться, решив, что некоторые специальные и сложные оптимизации следует исключить из рассмотрения. Многие оптимизации, не вошедшие в этот алгоритм, были выполнены авторами работ [Montgomery 1987, 1992a], [Zimmermann 2000] и [Woltman 2000] и оказались весьма полезными. Разнообразные оптимизации Циммермана вылились в открытие им в 1998 г. 49-значного делителя числа $M_{2071} = 2^{2071} - 1$. Вольтман реализовал (специфичные для случаев $n = 2^m \pm 1$) варианты алгоритмов взвешенного дискретного преобразования (ВДП) 9.5.17, 9.5.19, идеи эллиптического умножения с цепями сложений на основе последовательности Люка (см. алгоритм 3.6.7), а также технику с применением БПФ из работ [Crandall and Fagin 1994], [Crandall 1999b], благодаря которой сама эллиптическая алгебра выполняется в спектральном пространстве. В соответствии с вышесказанным, можно выполнить каждую из необходимых операций удвоения и сложения ($doubleh(), addh()$ в алгоритме 7.2.7, соответственно) со сложностью, эквивалентной 4-м умножениям целых чисел. Другими словами, благодаря хранению преобразований, каждая из указанных операций требует лишь 12 БПФ, каждые 3 из которых эквивалентны одному умножению целых чисел в алгоритме 7.2.7, так что мы получаем эквивалентность 4-м умножениям. Одним из достижений на этом пути стало открытие Карри и Вольтманом 53-значного делителя числа $M_{667} = 2^{667} - 1$. Поскольку эти данные имеют громадное значение для тех, кто желает протестировать алгоритм ECM, мы укажем следующие конкретные параметры. Карри использовал случайный инициализатор

$$\sigma = 8689346476060549$$

и пределы стадий

$$B_1 = 11000000,\ B_2 = 100B_1,$$

что позволило получить разложение числа $2^{667}-1$ на множители в виде

$$1943118631 \cdot 531132717139346021081 \cdot 97814658398863776553621 7 \cdot$$
$$536251126919238435081179423115164281730219033003445 67 \cdot P,$$

где в отношении последнего делителя $P$ доказано, что он является простым. Это прекрасный пример серьезного достижения ECM, который, на момент написания этой книги, содержит один из самых больших делителей, найденных ECM до сих пор, и он кажется еще более прекрасным, если посмотреть на порядок группы $\#E(\mathbf{F}_p)$ для приведенного выше 53-значного $p$ (и для данного случайного инициализатора $\sigma$), равный

$$2^4 \cdot 3^9 \cdot 3079 \cdot 152077 \cdot 172259 \cdot 1067063 \cdot 3682177 \cdot 3815423 \cdot 8867563 \cdot 15880351.$$

В самом деле, наибольший простой делитель $\#E$ здесь больше, чем $B_1$, и конечно, как указывают Карри и Вольтман, этот 53-значный делитель числа $M_{667}$ был найден на второй стадии. Заметим, что хотя эти исследователи использовали специальные оптимизации и алгоритмы, мы можем отыскать этот конкретный делитель, используя информацию, заключенную в приведенных параметрах, и разложить число $M_{667}$ на множители при помощи нашего алгоритма 7.4.4. Еще одним успехом явился 54-значный делитель числа $n = b^4 - b^2 + 1$, где $b = 6^{43} - 1$, найденный в январе 2000 г. Н. Лигеросом и М. Мизони. Это разложение можно подвергнуть такому же «разбору» порядка группы и т. п., какому было выше подвергнуто 53-значное открытие (см. [Zimmermann 2000]).

Другие успехи были достигнуты благодаря методу вычисления значений многочлена, предложенному Монтгомери и затронутому нами выше. Его метод был использован при открытии 47-значного делителя числа $5 \cdot 2^{256} + 1$, и некоторое время это оставалось одним из рекордов ECM. Подход с вычислением значений многочлена требует большого объема памяти, однако, как мы объясняли выше, он может значительно ускорить вторую стадию.

На случай, если читатель захочет встать на путь реализации ECM — практика, которая может оказаться весьма полезной — мы приведем здесь некоторые результаты в обозначениях алгоритма 7.4.4. Делитель числа Ферма, содержащий 33 десятичных знака и выписанный в разд. 1.3.2, а именно,

$$1687688170295169723830241270169 61 \mid F_{15},$$

был найден в 1997 г. Крэндаллом и ван Халевином с использованием следующих параметров: $B_1 = 10^7$ для предела первой стадии, с выбором $B_2 = 50B_1$ в качестве предела второй стадии и удачным выбором $\sigma = 253301772$, задающим успешную эллиптическую кривую $E_\sigma$. После того как был открыт этот 33-значный простой делитель $p$, Брент разложил порядок группы кривой $E_\sigma(\mathbf{F}_p)$:

$$\#E_\sigma(\mathbf{F}_p) = (2^5 \cdot 3 \cdot 1889 \cdot 5701 \cdot 9883 \cdot 11777 \cdot 5909317) \cdot 91704181,$$

где мы намеренно заключили «гладкую» часть порядка в скобки, а простое число 91704181 выходит за предел. Ясно, что $B_1$ «можно было бы» взять порядка 6 миллионов, а $B_2$ — порядка 100 миллионов. Но бесспорно, как сказал К. Зигель, «нельзя понять реальную сложность задачи, не попытавшись решить ее». В статье [Brent et al. 2000] указаны другие тестовые значения для недавно найденных делителей других чисел Ферма. Эти данные крайне полезны для отладки алгоритма. В самом деле, можно осуществить очень быструю проверку программы, если взять конкретное разложение известного порядка кривой и, начав с точки $P$, произвести домножения на заданное количество простых чисел, ожидая, что найденный делитель подтвердит правильность программы.

Как уже отмечалось, ECM особенно подходит тогда, когда неизвестный простой делитель не слишком велик, даже если само число $n$ очень велико. Практика показывает, что делители, найденные посредством ECM, достаточно редки в области 30 десятичных знаков, еще более редки в области 40 знаков, и в настоящий момент их численность падает до нуля где-то в районе 60 знаков.

## 7.5. Подсчет числа точек на эллиптической кривой

В разд. 7.3 мы видели, что число точек на эллиптической кривой, определенной над простым конечным полем $\mathbf{F}_p$, есть целое число из интервала $\left((\sqrt{p}-1)^2, (\sqrt{p}+1)^2\right)$. В этом разделе мы обсудим, что можно предпринять для практического нахождения этого числа.

### 7.5.1. Метод Шенкса—Местре

Для небольших простых чисел, скажем, для $p$, меньших 1000, можно просто явно вычислить сумму (7.8) для порядка $\#E_{a,b}(\mathbf{F}_p)$. Но без использования каких-либо специальных оптимизаций (таких, как быстрые алгоритмы для вычисления последовательных значений многочлена) это требует $O(p \ln p)$ операций в поле на все $O(p)$ экземпляров $(p-1)/2$-х степеней. Асимптотически более быстрый метод состоит в выборе точки $P$ на $E$ и нахождении всех кратных $[n]P$ для $n \in (p+1-2\sqrt{p}, p+1+2\sqrt{p})$, среди которых производится поиск $[n]P = O$. (Заметим, что так мы найдем лишь кратное порядка точки $P$ — оно будет искомым порядком, если окажется, что порядок точки $P$ имеет единственное кратное в интервале $(p+1-2\sqrt{p}, p+1+2\sqrt{p})$, что весьма вероятно.) Но такой подход требует $O(\sqrt{p}\ln p)$ операций в поле (с довольно большой константой в символе $O$, что обусловлено эллиптической арифметикой), и для больших $p$, скажем, больших, чем $10^{10}$, этот метод становится громоздким. Существуют более быстрые алгоритмы с оценкой $O\left(\sqrt{p}\ln^k p\right)$, которые не используют прямую эллиптическую алгебру (см. упр. 7.26), но они также не годятся для простых чисел, представляющих интерес в современном контексте, а именно, $p \approx 10^{50}$ и выше — эта приближенная граница в значительной степени обуслов-

лена практической криптографией. Однако не стоит отчаиваться, поскольку существуют изощренные современные алгоритмы и их оптимизации, которые доводят границу, связанную с подсчетом точек, до приемлемого уровня.

Существует изящный и зачастую полезный алгоритм с оценкой $O(p^{1/4+\varepsilon})$ для установления порядка кривой. Мы уже встречались с его основной идеей в алгоритме 5.3.1 — методе Шенкса *«детские шаги, гигантские шаги»* (для дискретных логарифмов). По существу этот алгоритм эксплуатирует удивительный ответ на следующий вопрос: если заданы два числовых списка длины $N$, скажем, $A = \{A_0, \ldots, A_{N-1}\}$ и $B = \{B_0, \ldots, B_{N-1}\}$, сколько потребуется операций (сравнений), чтобы определить, что $A \cap B$ пусто? А если непусто, каково в точности пересечение $A \cap B$? Наивным методом было бы просто сравнить $A_1$ с каждым $B_i$, затем сравнить $A_2$ с каждым $B_i$, и т. д. Эта неэффективная процедура дает, конечно, сложность $O(N^2)$. Гораздо лучше следующая процедура:

(1) отсортировать каждый из списков $A$, $B$, скажем, в неубывающем порядке;
(2) просмотреть отсортированные списки, отмечая результаты сравнений.

Как известно, шаг сортировки (1) требует $O(N \ln N)$ операций (сравнений), а шаг просмотра (2) можно выполнить всего лишь за $O(N)$ операций. Хотя концепция метода достаточно ясна, мы сочли нужным поместить здесь общий алгоритм поиска пересечения списков в явном виде. В следующей схеме множества $A, B$ на входе являются мультимножествами, т. е. допускаются повторения, но по окончании множество $A \cap B$ на выходе не содержит повторений. Мы будем предполагать, что задана функция $sort()$, которая возвращает отсортированный список, который состоит из тех же элементов, размещенных в неубывающем порядке. Например, $sort(\{3,1,2,1\}) = \{1,1,2,3\}$.

**Алгоритм 7.5.1** (поиск пересечения двух списков). **Получив на входе два конечных числовых списка $A = \{a_0, \ldots, a_{m-1}\}$ и $B = \{b_0, \ldots, b_{n-1}\}$, этот алгоритм возвращает множество, являющееся пересечением $A \cap B$, записанным в строго возрастающем порядке. Отметим, что дублирование в нем корректно устраняется, например, если $A = \{3,2,4,2\}$, $B = \{1,0,8,3,3,2\}$, то $A \cap B$ будет возвращено как $\{2,3\}$.**

1. [Инициализация]
   $A = sort(A)$; // Отсортировать в неубывающем порядке.
   $B = sort(B)$;
   $i = j = 0$;
   $S = \{\ \}$; // Вначале множество пересечения пусто.
2. [Стадия просмотра]
   while$((i < \#A)$ and $(j < \#B))$ {
      if$(a_i \leqslant b_j)$ {
         if$(a_i == b_j)$ $S = S \cup \{a_i\}$; // Добавить совпадение в множество $S$.
         $i = i + 1$;
         while$((i < (\#A) - 1)$ and $(a_i == a_{i-1}))$ $i = i + 1$;
      } else {

```
            j = j + 1;
            while((j < (#B) − 1) and (b_j == b_{j−1})) j = j + 1;
        }
    }
    return S;                                  // Возвратить пересечение A ∩ B.
```

Заметим, что мы привели алгоритм для произвольных длин списков, т. е. в нем не требуется, чтобы $\#A = \#B$. Из упомянутой выше оценки сложности сортировки следует, что в целом алгоритм имеет сложность $O(Q \ln Q)$ операций, где $Q = \max\{\#A, \#B\}$. Кстати говоря, существуют и другие альтернативные способы поиска пересечения списков (см. упр. 7.13).

А теперь о том, как Шенкс применил эту концепцию поиска пересечения списков в задаче о порядке кривой. Представьте, что мы можем найти для точки $P \in E$ соотношение вида

$$[p + 1 + u]P = \pm[v]P,$$

или, что то же самое, поскольку всегда $-(x, y) = (x, -y)$, мы обнаружили совпадение $x$-координат точек $[p + 1 + u]P$ и $vP$. Такое совпадение означает, что

$$[p + 1 + u \mp v]P = O.$$

Поиск этого совпадения может оказаться весьма утомительным, поскольку множитель слева теперь должен быть кратным порядку точки $P$, и может оказаться даже самим порядком кривой. Определим целое число $W = \lceil p^{1/4}\sqrt{2} \rceil$. Мы можем представить целые числа $k$, такие, что $|k| < 2\sqrt{p}$, в виде $k = \beta \mp \gamma W$, где $\beta$ изменяется в пределах $[0, W - 1]$, а $\gamma$ — в пределах $[0, W]$. (Мы используем буквы $\beta, \gamma$ для напоминания, соответственно, о «baby-steps» и «giant-steps» Шенкса[4].) Таким образом, мы можем сформировать список $x$-координат точек

$$\{[p + 1 + \beta]P : \beta \in [0, \ldots, W - 1]\},$$

назвав этот список $A$ (причем $\#A = W$), и отдельно список $x$-координат точек

$$\{[\gamma W]P : \gamma \in [0, \ldots, W]\},$$

назвав этот список $B$ (причем $\#B = W + 1$). Когда мы найдем совпадение, мы можем непосредственно проверить, какое из кратных $[p + 1 + \beta \mp \gamma W]P$ (или оба) является бесконечно удаленной точкой. Понятно, что порождение точек для детских шагов и гигантских шагов требует $O\left(p^{1/4}\right)$ эллиптических операций, а алгоритм пересечения — $O\left(p^{1/4} \ln p\right)$ операций, что в сумме дает сложность $O\left(p^{1/4+\varepsilon}\right)$.

К сожалению, нахождение кратного точки, которое обращается в нуль, полностью не решает задачу. Может оказаться, что найдено более одного нулевого кратного (именно по этой причине мы сформулировали алгоритм 7.5.1 так, чтобы он возвращал *все* элементы пересечения). Однако, как только выбранная

[4] $\beta$ — «baby-steps», «детские шаги», и $\gamma$ — «giant-steps», «гигантские шаги» (*англ.*). Для русскоязычного читателя, вероятно, больше подошли бы буквы $\delta$ — «детские шаги» и $\gamma$ — «гигантские шаги». —*Прим. перев.*

точка имеет порядок, больший $4\sqrt{p}$, наш алгоритм найдет единственное кратное этого порядка в нужном интервале, и это будет истинный порядок кривой. Иногда может случиться, что группа имеет малый показатель (т. е. все точки имеют малый порядок), и метод Шенкса никогда не найдет истинный порядок группы, если каждый раз будет выбирать по одной точке. Существуют два способа выйти из этого тупика. Один состоит в проведении итераций метода Шенкса с последовательным выбором точек и построением более широких подгрупп, которые не обязательно циклические. Если порядок такой подгруппы имеет единственное кратное в интервале Хассе, то это кратное и есть порядок кривой. Вторая идея гораздо более проста в реализации и основана на следующем результате Местре, см. [Cohen 2000], [Schoof 1995].

**Теорема 7.5.2** (Местре). *Для эллиптической кривой $E(\mathbf{F}_p)$ и ее искажения $E'(\mathbf{F}_p)$ посредством квадратичного невычета* mod $p$ *справедливо соотношение*

$$\#E + \#E' = 2p + 2.$$

*При $p > 457$ по меньшей мере на одной из эллиптических кривых $E, E'$ существует точка, порядок которой больше $4\sqrt{p}$. Более того, если $p > 229$, по меньшей мере одна из этих двух кривых обладает точкой $P$ с тем свойством, что единственное целое число $m \in (p + 1 - 2\sqrt{p}, p + 1 + 2\sqrt{p})$, для которого $[m]P = O$, есть истинный порядок кривой.*

Отметим, что соотношение $\#E + \#E' = 2p + 2$ доказать нетрудно (см. упр. 7.16), и основным результатом этой теоремы является утверждение о единственности $m$ в указанном интервале Хассе для порядков кривых. Дальнейшее несложное рассуждение показывает, что существует положительная постоянная $c$ (которая не зависит от $p$ и от эллиптической кривой), такая, что число точек $P$, удовлетворяющих условиям теоремы, превосходит $cp/\ln\ln p$ (см. упр. 7.17), так что такие точки встречаются довольно часто. Теперь идея состоит в том, чтобы использовать метод Шенкса на кривой $E$, и если это не приводит к успеху (из-за того, что порядок выбранной точки имеет более одного кратного в интервале Хассе) — использовать его на $E'$, а если и это не приводит к успеху — опять использовать его на $E$, и т. д. Согласно теореме, если мы повторим эти действия достаточное количество раз, то в конце концов добьемся успеха. Это приводит к эффективному алгоритму подсчета числа точек для кривых $E(\mathbf{F}_p)$, когда $p$ не превосходит границы, величина которой порядка $10^{30}$. В нижеследующем алгоритме через $x(P)$ мы обозначаем $x$-координату точки $P$. При выборе удобного сценария, когда все $x$-координаты задаются отношениями $X/Z$, тот факт, что знаменатель $Z = 0$, указывает, как обычно, на бесконечно удаленную точку.

**Алгоритм 7.5.3** (установление порядка кривой по Шенксу—Местре). Получив эллиптическую кривую $E = E_{a,b}(\mathbf{F}_p)$, этот алгоритм возвращает ее порядок $\#E$. Предполагаем, что для списка $S = \{s_1, s_2, \ldots\}$ и элемента $s \in S$ задана функция индекса $ind(S, s)$, которая возвращает некоторый индекс $i$, такой что $s_i = s$. В конце алгоритма также определена возвращающая список функция $shanks()$. Эта функция изменяет содержимое двух глобальных списков координат $A$ и $B$.

1. [Проверить величину числа $p$]
   if($p \leqslant 229$) return $p + 1 + \sum_x \left(\frac{x^3+ax+b}{p}\right)$; // Равенство (7.8).
2. [Инициализация поиска по Шенксу]
   Найти квадратичный невычет $g \pmod p$;
   $W = \lceil p^{1/4}\sqrt{2}\rceil$; // Параметр для гигантских шагов.
   $(c, d) = (g^2a, g^3b)$; // Параметры искажения.
3. [Цикл Местре] // Будем искать точку $P$ из теоремы 7.5.2.
   Выбрать случайное $x \in [0, p-1]$;
   $\sigma = \left(\frac{x^3+ax+b}{p}\right)$;
   if($\sigma == 0$) goto [Цикл Местре];
   // С этого момента мы имеем кривую определенной сигнатуры $\sigma = \pm 1$.
   if($\sigma == 1$) $E = E_{a,b}$; // Установить переменную
   // равной первоначальной кривой.
   else {
   $E = E_{c,d}$;
   $x = gx$; // Установить переменные равными
   // искажению кривой и допустимому $x$.
   }
   Определить начальную точку $P \in E$ так, что $x(P) = x$;
   $S = shanks(P, E)$; // Поиск пересечения по Шенксу.
   if($\#S \neq 1$) goto [Цикл Местре]; // Пока не будет найдено
   // ровно одно совпадение.
   Установить $s$ равным (единственному) элементу списка $S$;
   $\beta = ind(A, s)$; $\gamma = ind(B, s)$; // Найти индексы
   // единственного совпадения.
   Выбрать знак в выражении $t = \beta \pm \gamma W$ так, чтобы $[p + 1 + t]P == O$ на кривой $E$;
   return $p + 1 + \sigma t$; // Искомый порядок первоначальной кривой $E_{a,b}$.
4. [Функция $shanks()$]
   $shanks(P, E)$ { // Предполагается, что $P$ лежит на кривой $E$.
   $A = \{x([p + 1 + \beta]P) : \beta \in [0, W - 1]\}$; // Детские шаги.
   $B = \{x([\gamma W]P) : \gamma \in [0, W]\}$; // Гигантские шаги.
   return $S = A \cap B$; // Посредством алгоритма 7.5.1.
   }

Отметим, что присвоить значение точке $P$ на основании случайного $x$ можно либо как $P = (x, y, 1)$, где $y$ — квадратный корень из соответствующей кубической формы, либо как $P = [x : 1]$ в случае параметризации Монтгомери (и значит, отсутствия $y$-координат) — по своему усмотрению. (В случае последней параметризации алгоритм следует несколько модифицировать для работы с той записью, которая соответствует теореме 7.2.6.) Подобным образом в функции $shanks()$ можно использовать алгоритм 7.2.7 (или более эффективное и специальное применение функций $addh()$, $doubleh()$) для получения необходимых кратных точек в виде $[X : Z]$, а затем построить списки $A, B$ из чисел $XZ^{-1}$. Можно даже представить себе реализацию всей процедуры без вычис-

ления обратных путем разработки аналога детских шагов и гигантских шагов для списков *пар* $(x, z)$ и поиска совпадений не в виде $x = x'$, а в виде $xz' = zx'$.

Условие применимости подхода Шенкса—Местре в виде $p > 229$ не является искусственным. При $p = 229$ возможна ситуация, когда существование одноэлементного множества совпадений $s$ не гарантировано (см. упр. 7.18).

## 7.5.2. Метод Шуфа

Рассмотрев схемы подсчета точек, которые имеют сложность в пределах от $O\left(p^{1+\epsilon}\right)$ до $O\left(p^{1/2+\epsilon}\right)$ и $O\left(p^{1/4+\epsilon}\right)$, обратимся к изящному алгоритму подсчета точек, который принадлежит Шуфу и имеет полиномиальную сложность — $O\left(\ln^k p\right)$, где $k$ фиксировано. Основная идея Шуфа состоит в том, чтобы найти вычеты порядка кривой $\#E \pmod l$ для достаточного количества небольших простых чисел $l$, чтобы затем восстановить искомый порядок при помощи китайской теоремы об остатках. Вначале рассмотрим относительно тривиальный случай нахождения $\#E \pmod 2$. Итак, порядок любой группы является четным тогда и только тогда, когда в ней существует элемент порядка 2. Для точки $P \neq O$ равенство $2P = O$ справедливо в том и только в том случае, когда в вычисляемый угловой коэффициент (из определения 7.1.2) входит нулевая $y$-координата, а потому нам известно, что точки порядка 2 — это в точности те точки, которые имеют вид $P = (x, 0)$. Поэтому порядок кривой является четным тогда и только тогда, когда задающий кривую многочлен $x^3 + ax + b$ имеет корни в поле $\mathbf{F}_p$. Это, в свою очередь, можно проверить с помощью НОД многочленов, как было указано в алгоритме 2.3.10.

Чтобы рассмотреть нахождение $\#E \pmod l$ для малых простых чисел $l > 2$, нам необходимо ввести несколько новых средств для работы с эллиптическими кривыми над конечными полями. Допустим, у нас есть эллиптическая кривая $E(\mathbf{F}_p)$, но мы рассматриваем на этой кривой точки с координатами, принадлежащими алгебраическому замыканию $\overline{\mathbf{F}}_p$ поля $\mathbf{F}_p$. Возведение в $p$-ю степень является автоморфизмом поля $\overline{\mathbf{F}}_p$, оставляющим на месте элементы поля $\mathbf{F}_p$, так что этот автоморфизм, примененный к координатам точки $(x, y) \in E(\overline{\mathbf{F}}_p)$, переводит эту точку в другую точку на кривой $E(\overline{\mathbf{F}}_p)$. А поскольку в правила сложения точек входят рациональные выражения от $\mathbf{F}_p$-коэффициентов определяющего кривую уравнения, ясно, что данное отображение является групповым автоморфизмом кривой $E(\overline{\mathbf{F}}_p)$. Это знаменитый эндоморфизм Фробениуса Ф. Так что если $(x, y) \in E(\overline{\mathbf{F}}_p)$, то $\Phi(x, y) = (x^p, y^p)$ (а также, по определению, $\Phi(O)$ равно $O$), и это отображение является групповым автоморфизмом кривой $E(\overline{\mathbf{F}}_p)$. Может возникнуть вопрос: в чем смысл рассмотрения алгебраического замыкания поля $\mathbf{F}_p$, когда на самом деле нас интересуют точки над самим полем $\mathbf{F}_p$? Ответ дает замечательная теорема: если порядок группы эллиптической кривой $E(\mathbf{F}_p)$ равен $p + 1 - t$, то

$$\Phi^2(P) - [t]\Phi(P) + [p]P = O$$

для каждой точки $P \in E(\overline{\mathbf{F}}_p)$. Иными словами, автоморфизм Фробениуса удовлетворяет квадратному уравнению, причем след (сумма корней многочлена

$x^2 - tx + p$) равен $t$, а это число даст нам порядок группы $E(\mathbf{F}_p)$. Далее в игру вступает вторая идея.

Для любого положительного целого числа $n$ рассмотрим те точки $P$ кривой $E(\overline{\mathbf{F}}_p)$, для которых $[n]P = O$. Это множество точек обозначается через $E[n]$, оно состоит из точек рассматриваемой группы, порядок которых делит $n$, одним словом, из точек $n$-кручения. Ключевыми являются два несложных факта относительно множества $E[n]$. Это множество является подгруппой в $E(\overline{\mathbf{F}}_p)$, и $\Phi$ отображает $E[n]$ в себя. Таким образом, справедливо соотношение

$$\Phi^2(P) - [t \bmod n]\Phi(P) + [p \bmod n]P = O \text{ для всех } P \in E[n]. \qquad (7.9)$$

Блестящая идея Шуфа (см. [Schoof 1985], [Schoof 1995]) состояла в том, чтобы использовать это соотношение для нахождения вычета $t \bmod n$ методом подбора, производимого до тех пор, пока не будет найдено правильное значение, удовлетворяющее (7.9). Для этого используются полиномы деления. Эти полиномы моделируют эллиптическое умножение, а заодно отбирают точки $n$-кручения.

**Определение 7.5.4.** С эллиптической кривой $E_{a,b}(\mathbf{F}_p)$ мы связываем полиномы деления $\Psi_n(X,Y) \in \mathbf{F}_p[X,Y]/(Y^2 - X^3 - aX - b)$, которые определяются следующим образом:

$$\begin{aligned}
\Psi_{-1} &= -1, \quad \Psi_0 = 0, \quad \Psi_1 = 1, \quad \Psi_2 = 2Y, \\
\Psi_3 &= 3X^4 + 6aX^2 + 12bX - a^2, \\
\Psi_4 &= 4Y\left(X^6 + 5aX^4 + 20bX^3 - 5a^2X^2 - 4abX - 8b^2 - a^3\right),
\end{aligned}$$

а все последующие случаи задаются соотношениями

$$\begin{aligned}
\Psi_{2n} &= \Psi_n\left(\Psi_{n+2}\Psi_{n-1}^2 - \Psi_{n-2}\Psi_{n+1}^2\right)/(2Y), \\
\Psi_{2n+1} &= \Psi_{n+2}\Psi_n^3 - \Psi_{n+1}^3\Psi_{n-1}.
\end{aligned}$$

Отметим, что при построении полиномов деления любое вхождение степени переменной $Y$, выше первой, должно быть редуцировано в соответствии с соотношением $Y^2 = X^3 + aX + b$. Некоторые важные в вычислительном отношении свойства полиномов деления собраны в следующую теорему.

**Теорема 7.5.5** (свойства полиномов деления). *Полином деления $\Psi_n(X,Y)$ при нечетном $n$ является полиномом от одной переменной $X$, а при четном $n$ он равен переменной $Y$, умноженной на полином от одной переменной $X$. Если $n$ нечетно и не кратно $p$, то $\deg(\Psi_n) = (n^2-1)/2$. Если $n$ четно и не кратно $p$, то степень полинома $\Psi_n$ по переменной $X$ равна $(n^2-4)/2$. Для точки $(x,y) \in E(\overline{\mathbf{F}}_p) \setminus E[2]$ равенство $[n]P = O$ справедливо тогда и только тогда, когда $\Psi_n(x) = 0$ (если $n$ нечетно) и $\Psi_n(x,y) = 0$ (если $n$ четно). Далее, если $(x,y) \in E(\overline{\mathbf{F}}_p) \setminus E[n]$, то*

$$[n](x,y) = \left(x - \frac{\Psi_{n-1}\Psi_{n+1}}{\Psi_n^2}, \frac{\Psi_{n+2}\Psi_{n-1}^2 - \Psi_{n-2}\Psi_{n+1}^2}{4y\Psi_n^3}\right).$$

Заметим, что если в последнем утверждении $y = 0$, то $n$ должно быть нечетным (поскольку равенство $y = 0$ указывает на точку порядка 2, а нам

дано, что $(x, y) \notin E[n]$), так что $y^2$ делит числитель рационального выражения для второй координаты. В этом случае естественно положить это выражение равным 0.

Следует отметить, что для нечетного простого числа $l \neq p$ существует *единственное* целое число $t$ из промежутка $[0, l-1]$, такое что

$$\left(x^{p^2}, y^{p^2}\right) + [p \bmod l](x, y) = [t]\left(x^p, y^p\right) \text{ для всех } (x, y) \in E[l] \setminus \{O\}. \quad (7.10)$$

Действительно, это сразу следует из соотношения (7.9) и следствия из теоремы 7.5.5 о том, что $E(\overline{\mathbf{F}}_p)$ в самом деле содержит точки порядка $l$. Если это единственное целое число $t$ будет найдено, то мы будем знать, что порядок группы $E(\mathbf{F}_p)$ сравним с $p + 1 - t$ по модулю $l$.

С вычислительной точки зрения приведенное соотношение означает, что использование полиномов деления позволяет проверить различные варианты выбора $t$ и найти среди них подходящий. Для этого поступают следующим образом.

(1) Под точками подразумевают пары многочленов из $\mathbf{F}_p[X, Y]$.
(2) Поскольку эти точки лежат на кривой $E$, мы можем постоянно производить редукцию по модулю многочлена $Y^2 - X^3 - aX - b$, чтобы хранить степени переменной $Y$ не выше первой, а поскольку точки, которые мы рассматриваем, лежат в $E[n]$, мы можем также производить редукцию по модулю полинома $\Psi_n$, чтобы степени переменной $X$ также держались под контролем. Наконец, коэффициенты многочленов лежат в поле $\mathbf{F}_p$, так что в любой удобный момент можно проводить редукцию коэффициентов по модулю $p$. Эти три типа редукций могут применяться в любом порядке.
(3) Большие степени переменных $X, Y$ должны быть редуцированы схемой возведения в степень, наподобие той, которая приведена в алгоритме 2.1.5, в которую добавлены соответствующие операции mod (взятие остатка) для многочленов, выполняемые в процессе ее работы для постоянного понижения степени.
(4) Сложение в левой части соотношения (7.10) моделируется с помощью формул определения 7.1.2.

На первый взгляд может показаться, что необходимо явное вычисление обратного многочлена из фундаментального определения эллиптической операции. Обратный многочлен можно вычислить с помощью алгоритма 2.2.2, но в этом нет необходимости, как показывает следующее наблюдение. Различные алгоритмы эллиптического сложения, рассмотренные нами ранее, показывают, что без вычисления обратных элементов можно обойтись, если использовать специальные представления координат. На практике оказывается удобным работать либо с проективным представлением точек (алгоритм 7.2.3), либо с «рациональным» вариантом этого представления. Опишем последнее представление, поскольку оно хорошо подходит для вычислений с полиномами деления, особенно это касается свойства умножения точек из теоремы 7.5.5. Будем рассматривать точку как $P = (U/V, F/G)$, где $U, V, F, G$ все являются многочле-

нами, предположительно зависящими от переменных $X, Y$. Существует альтернативная стратегия, состоящая в использовании проективных координат согласно упр. 7.29. И ту и другую стратегию упрощает то, что в алгоритме Шуфа любая полученная нами точка *всегда* будет иметь определенный вид. Например, в приведенном ниже тексте нашего алгоритма используется выбор параметризации $P = (U/V, F/G)$, и мы всегда будем иметь точку вида

$$P = (N(X)/D(X), YM(X)/C(X)),$$

что обусловлено алгеброй полиномов деления. Далее, следует представлять себе четыре указанных многочлена редуцированными по модулю $\Psi_n$ и по модулю $p$ в смысле указанного выше пункта (2). Другое усовершенствование, которое мы находим практически эффективным, состоит в привлечении функции умножения больших многочленов в виде нашего алгоритма 9.6.1 (см. также его альтернативы, например, в упр. 9.70), который имеет особые преимущества из-за того, что степень $\deg(\Psi_n)$ очень велика, и потому обычная арифметика многочленов требует больших усилий. Еще большая эффективность достигается, когда мы используем наш алгоритм 9.6.4 для получения остатка от деления на эти полиномы высокой степени (операция mod).

**Алгоритм 7.5.6** (явный вид алгоритма Шуфа для порядка кривой). Пусть $p > 3$ — простое число. Для кривой $E_{a,b}(\mathbf{F}_p)$ этот алгоритм возвращает значение $t \pmod l$, где $l$ — простое число (много меньшее $p$), а для порядка кривой справедливо равенство $\#E = p+1-t$. Точный порядок кривой, таким образом, получается посредством выполнения этого алгоритма для достаточного количества простых чисел $l$, таких, что $\prod l > 4\sqrt{p}$, за которым следует применение китайской теоремы об остатках для восстановления точного значения $t$. Предполагаем, что для разумной верхней границы $L \geqslant l$, которую не превосходят возможные используемые значения $l$, мы имеем предварительно вычисленными полиномы деления $\Psi_{-1}, \ldots, \Psi_{L+1} \bmod p$, старшие коэффициенты которых могут быть сделаны равными единице (путем деления по модулю $p$ каждого многочлена на его старший коэффициент), например, с целью применения алгоритма 9.6.4.

1. [Проверить $l = 2$]
   if($l == 2$) {
   $g(X) = \text{НОД}(X^p - X, X^3 + aX + b)$; // НОД многочленов
   // в кольце $\mathbf{F}_p[X]$.
   if($g(X)$ != 1) return 0; // $t \equiv 0 \pmod 2$,
   // так что порядок $\#E$ — четный.
   return 1; // $\#E$ — нечетный.
   }
2. [Анализ соотношения (7.10)]
   $\overline{p} = p \bmod l$;
   $u(X) = X^p \bmod (\Psi_l, p)$;
   $v(X) = (X^3 + aX + b)^{(p-1)/2} \bmod (\Psi_l, p)$;
   // Т.е. $v(X) = Y^{p-1} \bmod (\Psi_l, p)$.
   $P_0 = (u(X), Yv(X))$; // $P_0 = (X^p, Y^p)$.

$P_1 = (u(X)^p \bmod (\Psi_l, p), Yv(X)^{p+1} \bmod (\Psi_l, p))$;
// $P_1 = (X^{p^2}, Y^{p^2})$.
Выразить $P_2 = [\overline{p}](X,Y)$ в рациональном виде $\left(\frac{N(X)}{D(X)}, Y\frac{M(X)}{C(X)}\right)$, например, используя теорему 7.5.5;
if($P_1 + P_2 == O$) return 0; // $\#E = p + 1 - t$ и $t \equiv 0 \pmod{l}$.
$P_3 = P_0$;
for($1 \leqslant k \leqslant l/2$) {
  if($X$-координаты точек $(P_1 + P_2)$ и $P_3$ совпадают) {
    if($Y$-координаты также совпадают) return $k$; // Проверка $Y$-координат.
    return $l - k$;
  }
  $P_3 = P_3 + P_0$;
}

Когда после сложения сравниваются соответствующие координаты точек $(P_1 + P_2)$ и $P_3$, в общей постановке вопрос звучит так: выполнено ли соотношение

$$(N_1/D_1, YM_1/C_1) + (N_2/D_2, YM_2/C_2) = (N_3/D_3, YM_3/C_3),$$

которое, разумеется, должно проверяться с использованием обычных правил эллиптического сложения. Содержащую многочлены точку $P_1 + P_2$ в левой части можно привести — с использованием правил эллиптической арифметики алгоритма 7.2.2, координатами в котором теперь являются, конечно же, наши отношения многочленов — к виду $(N'/D', YM'/C')$, и эту пару отношений многочленов нужно сравнить с парой $(N_3/D_3, YM_3/C_3)$. При таком сравнении, в свою очередь, мы проверяем, обращаются ли в нуль $\bmod (\Psi_l, p)$ смешанные произведения $(N_3D' - N'D_3)$ и $(M_3C' - M'C_3)$. А для проверки того, что $P_1 + P_2 = O$, мы выясняем, выполнено ли соотношение $M_1/C_1 = -M_2/C_2$, которое также сводится к простому соотношению для смешанного произведения. Это делается для того, чтобы вся реализация, которую мы описываем, использовала лишь умножение многочленов и редукцию $\bmod (\Psi_l, p)$ (и не использовала вычисление обратных многочленов). И как мы уже упоминали, и умножение многочленов, и операция mod могут быть сделаны весьма эффективными.

На тот случай, если читателем будет предпринята попытка реализовать алгоритм 7.5.6, приведем здесь некоторые подробности вычислений, с целью, скажем так, «отладки алгоритма». Для $p = 101$ и кривой

$$Y^2 = X^3 + 3X + 4$$

над полем $\mathbf{F}_p$ при выборе в качестве $l$ значений $l = 2, 3, 5, 7$ алгоритм выдает результаты $t \bmod 2 = 0$, $t \bmod 3 = 1$, $t \bmod 5 = 0$, $t \bmod 7 = 3$, из которых мы заключаем, что $\#E = 92$[5]. (Можно было бы пропустить простое число

[5]Поскольку $t = 10$ и $\#E = p + 1 - t$. — *Прим. ред.*

$l = 5$, поскольку произведение остальных простых чисел превосходит $4\sqrt{p}$.) В процессе работы получаем, например,

$$\begin{aligned}
\Psi_3 &= 98 + 16X + 6X^2 + X^4, \\
\left(X^{p^2}, Y^{p^2}\right) &= \left(32 + 17X + 13X^2 + 92X^3,\ Y(74 + 96X + 14X^2 + 68X^3)\right), \\
[2](X,Y) &= \left(\frac{12 + 53X + 89X^2}{16 + 12X + 4X^3},\ Y\frac{74 + 10X + 5X^2 + 64X^3}{27 + 91X + 96X^2 + 37X^3}\right), \\
(X^p, Y^p) &= \left(70 + 61X + 83X^2 + 44X^3,\ Y(43 + 76X + 21X^2 + 25X^3)\right),
\end{aligned}$$

при этом следует отметить, что каждый многочлен, входящий в состав координат этих точек, редуцирован mod $(\Psi_3, p)$. (Обратим внимание на то, что $\overline{p}$ из шага [Анализ ...] в данном случае[6] равно 2, по этой причине мы рассматриваем $[2](X,Y)$.) Оказывается, что последняя из выписанных здесь точек в самом деле является эллиптической суммой двух предыдущих, в согласии с тем утверждением, что $t \bmod 3 = 1$.

Существует одно важное усовершенствование, которое мы намеренно оставили в стороне для большей ясности. Оно состоит в том, что с тем же успехом можно работать со степенями простых чисел. Иными словами, $l = q^a$ можно напрямую использовать в нашем алгоритме (при этом игнорируя блок вычисления НОД для случая $l = 2$, когда $l = 4, 8, 16, \ldots$) и несколько сократить объем вычислений. Все, что при этом требуется — это то, чтобы в итоге произведение всех использованных степеней $l$ (но не более одной степени на каждое простое число) превосходило $4\sqrt{p}$.

Посредством этой базовой схемы Шуфа мы имеем возможность устанавливать порядки кривых в области простых чисел $p \approx 10^{80}$, используя при этом степени простых чисел $l < 100$. Иногда в литературе говорится, что использование чисел $l$, намного больших, чем, скажем, 30, представляется маловероятным. Однако с помощью упомянутых оптимизаций, в особенности алгоритмов умножения/операции mod для больших многочленов из гл. 9, простое число Шуфа $l$ можно довести до 100 и, может быть, продвинуть еще дальше.

Если прекращать использование алгоритма 7.5.6 до того, как китайская теорема об остатках даст однозначный ответ (т. е. если обрабатывать недостаточное для восстановления порядка количество малых простых чисел $l$), а после этого применять подход Шенкса—Местре для завершения вычислений на основании полученной информации о возможных порядках, можно, в свою очередь, еще дальше отодвинуть нашу приближенную границу $10^{80}$. Однако мощь алгоритма Шуфа подтверждает то обстоятельство, что к 80-и знакам, с которыми мы умеем работать, пользуясь одним только алгоритмом Шуфа, «поднятие Шенкса—Местре» позволяет добавить всего лишь несколько — скажем, 10 или 20 — дополнительных десятичных знаков. Поэтому на практике обычно больше оправдывает себя улучшение имеющейся реализации алгоритма Шуфа, а не создание над ним надстройки в виде алгоритма Шенкса—Местре.

[6] Т. е. при $l = 3$. — *Прим. ред.*

Спрашивается, можно ли осуществлять подсчет точек для существенно больших простых чисел? На самом деле, видоизмененный алгоритм Шуфа — вариант «Шуфа—Элкиса—Аткина» (SEA), созданный авторами работ [Atkin 1986, 1988, 1992] и [Elkies 1991, 1997] и улучшенный в вычислительном отношении в работах [Morain 1995], [Couveignes and Morain 1994], [Couveignes et al. 1996], — достиг беспрецедентной эффективности подсчета точек. Существенное продвижение Элкиса состояло в наблюдении, что для некоторых значений $l$ (каких именно, зависит от $a, b, p$; по существу, примерно для половины возможных значений $l$) можно использовать определенный многочлен $f_l$, который делит $\Psi_l$, но имеет степень $(l-1)/2$, и в том, что соотношение Шуфа (7.10) можно упростить. Подход Элкиса состоит в нахождении собственного значения $\lambda$, такого, что

$$(X^p, Y^p) = [\lambda](X, Y),$$

где все вычисления выполняются по mod $(f_l, p)$, откуда $\#E = p + 1 - t$ и известно, что

$$t \equiv \lambda + p/\lambda \pmod{l}.$$

Поскольку степени многочленов $f_l$ столь малы, это важное открытие эффективно снижает оценку сложности на степень $\ln p$ и дает оценку $O(\ln^6 p)$, что улучшает первоначальную оценку сложности Шуфа $O(\ln^8 p)$ (см. [Schoof 1995]). (Заметим, однако, что эти оценки предполагают прямое «школьное» умножение целых чисел и допускают дальнейшее понижение степени ln.) Безусловно, идеи варианта SEA поразительно эффективны. Аткин, например, использовал эти оптимизации при нахождении в 1992 г. для наименьшего простого числа, состоящего из 200 десятичных знаков, а именно,

$$\begin{aligned} p \;=\; & 10000000000000000000000000000000000000000000000000\backslash \\ & 00000000000000000000000000000000000000000000000000\backslash \\ & 00000000000000000000000000000000000000000000000000\backslash \\ & 00000000000000000000000000000000000000000000000153, \end{aligned}$$

и кривой над полем $\mathbf{F}_p$, заданной уравнением

$$Y^2 = X^3 + 105X + 78153,$$

порядка группы точек

$$\begin{aligned} \#E \;=\; & 10000000000000000000000000000000000000000000000000\backslash \\ & 00000000000000000000000000000000000000000000000000\backslash \\ & 06789750288004224118080314365460277641928049641888\backslash \\ & 39991591392960032210630561760029050858613689631753. \end{aligned}$$

Интересно, что нетрудно согласиться с тем, что этот выбор кривой «случаен» (пусть даже простое число $p$ и не является таковым). Параметры $(a, b) = (105, 78153)$ для этой кривой были взяты из почтового адреса во Франции (см. [Schoof 1995]). Впоследствии Морейну удалось провести дополнительные

вычислительные усовершенствования и вычислить в явном виде порядок определенной кривой над полем $\mathbf{F}_p$, где $p$ — 500-значное простое число (см. [Morain 1995]).

В нашем рассмотрении, касающемся мощного алгоритма Шуфа и его оптимизаций, мы, как говорится, затронули лишь вершину айсберга. Существует еще очень многое, о чем можно рассказать. Хорошим современным источником информации о практическом подсчете точек на эллиптических кривых служит книга [Seroussi et al. 1999], кроме того, сообщается о различных реализациях, продолжающих вариант SEA (см. [Izu et al. 1998], [Scott 1999]).

В своей фундаментальной статье [Schoof 1985] Шуф описал приложение своего метода подсчета точек к нахождению квадратных корней по модулю $p$ из целого числа $D$ с (детерминированной, а не вероятностной) полиномиальной сложностью, в предположении, что $D$ фиксировано. Несмотря на то, что обычно используемые вероятностные алгоритмы 2.3.8, 2.3.9 гораздо более эффективны на практике, подход Шуфа к нахождению квадратных корней с помощью подсчета точек устанавливает, по крайней мере для фиксированного $D$, настоящую детерминированную полиномиальную оценку сложности.

Нельзя устоять перед соблазном привести здесь один забавный анекдот. Как упоминает Элкис (см. [Elkies 1997]), статья с грандиозным алгоритмом Шуфа была отклонена в своем первоначальном виде как не представляющая, по мнению рецензентов, особой ценности. Но с измененным названием, которое теперь оканчивается на «... square roots mod $p$»[7], исправленная статья [Schoof 1985] была, к нашему удовольствию, наконец опубликована.

Хотя метод SEA остается на данный момент тем бастионом, на который возлагают надежды при подсчете точек на $E(\mathbf{F}_p)$ при простом $p$, появился ряд очень новых (и интересных) результатов для кривых $E(\mathbf{F}_{p^d})$, где простое число $p$ мало. Оказывается, Р. Харли в 2002 г. показал, что это число точек может быть подсчитано при фиксированной характеристике $p$ за время

$$O(d^2 \ln^2 d \ln\ln d),$$

и ему удалось подсчитать число точек на кривой, определенной над гигантским полем $\mathbf{F}_{2^{130020}}$. Другие направления исследований связаны с именем Т. Сато и относятся к каноническим поднятиям и даже к $p$-адическим формам арифметико-геометрического среднего (arithmetic-geometric mean — AGM). Один из хороших способов составить свое представление об активности в этой новой алгебраической области — внимательно изучить ссылки на сайте Харли [Harley 2002].

### 7.5.3. Метод Аткина—Морейна

Мы задавали вопрос: если дана кривая $E = E_{a,b}(\mathbf{F}_p)$, как найти ее порядок $\#E$? Сформулируем нечто вроде обратной задачи, которая крайне важна при доказательстве простоты и в криптографии: можем ли мы сначала задать подходящий порядок $\#E$, а *после этого* указать кривую, имеющую данный по-

[7] «... квадратные корни по модулю $p$» (*англ.*). — *Прим. перев.*

рядок? Например, нам хотелось бы иметь простой порядок, или порядок $2q$ при простом $q$, или же порядок, который делится на большую степень числа 2. Можно было бы назвать это изучением «явно выражаемых» порядков кривых, имея в виду следующее. Для определенных представлений вида $4p = u^2 + |D|v^2$, с которыми мы ранее сталкивались в алгоритме 2.3.13, можно немедленно выписать определенные порядки кривых, а также — обычно затратив больше усилий — параметры $a, b$ соответствующих уравнений кривых. Эти идеи явились результатом плодотворной работы А. О. Л. Аткина в конце 80-х годов XX века и его более поздней работы в сотрудничестве с Ф. Морейном.

Для того, чтобы осознать эти идеи, необходимо ненадолго погрузиться в некоторые дополнительные теоретические рассмотрения, связанные с эллиптическими кривыми. Более основательное изложение см. в статье [Atkin and Morain 1993b] и книгах [Cohen 2000], [Silverman 1986].

Для эллиптической кривой $E$, определенной над полем комплексных чисел $\mathbf{C}$, можно рассмотреть «эндоморфизмы» кривой $E$. Это групповые гомоморфизмы группы $E$ в себя, которые задаются рациональными функциями. Множество таких эндоморфизмов, обозначаемое через $\mathrm{End}(E)$, образует естественно определяемое кольцо, в котором сложение получается из эллиптического сложения, а умножение есть композиция. А именно, если $\phi, \sigma$ принадлежат $\mathrm{End}(E)$, то $\phi + \sigma$ есть эндоморфизм кривой $E$, который переводит точку $P$ в точку $\phi(P) + \sigma(P)$, где последний «+» означает эллиптическое сложение, а $\phi \cdot \sigma$ есть эндоморфизм кривой $E$, который переводит $P$ в $\phi(\sigma(P))$.

Если $n$ — целое число, то отображение $[n]$, которое переводит точку $P$ кривой $E$ в точку $[n]P$, есть элемент кольца $\mathrm{End}(E)$, поскольку это гомоморфизм группы, и поскольку теорема 7.5.5 показывает, что $[n]P$ имеет координаты, которые являются рациональными функциями от координат точки $P$. Таким образом, кольцо $\mathrm{End}(E)$ содержит изоморфную копию кольца целых чисел $\mathbf{Z}$. Часто, и даже как правило, полное описание кольца $\mathrm{End}(E)$ дает кольцо целых чисел. Тем не менее, иногда существуют эндоморфизмы кривой $E$, которым не соответствуют целые числа. Оказывается, впрочем, что кольцо $\mathrm{End}(E)$ никогда не бывает гораздо шире кольца $\mathbf{Z}$. Если оно не изоморфно $\mathbf{Z}$, то оно изоморфно порядку во мнимом квадратичном числовом поле. («Порядок» — это подкольцо конечного индекса в кольце целых алгебраических чисел этого поля.) В таком случае говорят, что кривая $E$ обладает комплексным умножением (complex multiplication), или является CM-кривой.

Предположим, что $E$ — эллиптическая кривая, определенная над полем рациональных чисел, которая, будучи рассмотренной над полем комплексных чисел, обладает комплексным умножением на элементы порядка в поле $\mathbf{Q}(\sqrt{D})$, где $D$ — отрицательное целое число. Предположим, что $p > 3$ — простое число, которое не делит дискриминант кривой $E$. Тогда мы можем рассмотреть кривую $E$ над полем $\mathbf{F}_p$, произведя редукцию коэффициентов кривой $E$ по модулю $p$. Предположим, что простое число $p$ является нормой некоторого целого алгебраического числа в поле $\mathbf{Q}(\sqrt{D})$. В этом случае оказывается, что мы можем с легкостью найти порядок группы эллиптической кривой $E(\mathbf{F}_p)$. Работа по вычислению этого порядка не требует даже коэффициентов нашей кривой

$E$, нужны только числа $D$ и $p$. К тому же работа по вычислению порядка проста — надо использовать алгоритм Корнаккьи—Смита 2.3.13. Существует дополнительная, отчасти более трудная, работа по вычислению коэффициентов уравнения, определяющего кривую $E$, но если из каких-то соображений мы делаем вывод, что рассматриваемый порядок не будет нам полезен, то для него эту дополнительную работу можно не проводить. В этом, по существу, и состоит идея Аткина и Морейна.

Сейчас мы рассмотрим некоторые идеи, касающиеся мнимых квадратичных полей, а также двойственную этим полям теорию бинарных квадратичных форм отрицательного дискриминанта. Некоторые из этих идей были развиты в разд. 5.6. Определим те (отрицательные) дискриминанты $D$, которые имеют отношение к установлению порядка кривой.

**Определение 7.5.7.** Отрицательное целое число $D$ есть фундаментальный дискриминант, если нечетная часть разложения числа $D$ свободна от квадратов и $|D| \equiv 3, 4, 7, 8, 11, 15 \pmod{16}$.

Если быть более кратким, то это дискриминанты мнимых квадратичных полей. Так вот, с каждым фундаментальным дискриминантом связано число классов $h(D)$. Как мы видели в разд. 5.6.3, $h(D)$ есть порядок группы $\mathcal{C}(D)$ приведенных бинарных квадратичных форм дискриминанта $D$. В разд. 5.6.4 мы упоминали, как для подсчета $h(D)$ может применяться метод Шенкса «детские шаги, гигантские шаги». Нижеследующий алгоритм способен также производить этот подсчет, при необходимости генерировать соответствующие приведенные формы, а также вычислять гильбертов многочлен классов, отвечающий дискриминанту $D$. Это многочлен степени $h(D)$ с коэффициентами в $\mathbf{Z}$, такой что поле разложения этого многочлена над $\mathbf{Q}(\sqrt{D})$ имеет группу Галуа, изоморфную группе классов $\mathcal{C}(D)$. Это поле разложения называется гильбертовым полем классов для $\mathbf{Q}(\sqrt{D})$ и является максимальным абелевым неразветвленным расширением поля $\mathbf{Q}(\sqrt{D})$. Гильбертово поле классов обладает тем свойством, что простое число $p$ полностью разлагается в этом поле тогда и только тогда, когда существуют целые числа $u, v$, такие, что $4p = u^2 + |D|v^2$. В частности, поскольку гильбертово поле классов имеет степень $2h(D)$ над полем рациональных чисел $\mathbf{Q}$, простые числа $p$, для которых $4p$ представимо в таком виде, имеют плотность $1/2h(D)$ среди всех простых чисел, см. [Cox 1989].

Нам потребуется функция (вновь мы пропускаем красивые и нетривиальные основы этой теории ради немедленной разработки алгоритма)

$$\Delta(q) = q\left(1 + \sum_{n=1}^{\infty}(-1)^n\left(q^{n(3n-1)/2} + q^{n(3n+1)/2}\right)\right)^{24},$$

возникающая в теории инвариантов и модулярных форм (см. [Cohen 2000], [Atkin and Morain 1993b]). (Интересно, что $\Delta(q)$ имеет другое красивое представление $q\prod_{n\geqslant 1}(1-q^n)^{24}$, но мы не будем пользоваться им в дальнейшем. Первоначально выписанное выражение для $\Delta(q)$ легче поддается вычислению, поскольку показатели в нем растут квадратичным образом.)

**Алгоритм 7.5.8** (число классов и гильбертов многочлен классов). При вводе (отрицательного) фундаментального дискриминанта $D$ этот алгоритм возвращает любую желаемую комбинацию элементов множества, состоящего из числа классов $h(D)$, гильбертова многочлена классов $T \in \mathbf{Z}[X]$ (степень которого равна $h(D)$) и множества приведенных форм $(a, b, c)$ дискриминанта $D$ (число элементов в нем равно $h(D)$).

1. [Инициализация]
   $T = 1$;
   $b = D \bmod 2$;
   $r = \lfloor \sqrt{|D|/3} \rfloor$;
   $h = 0$; // Присвоить нулевое значение счетчику классов.
   $red = \{\ \}$; // Пустое множество примитивных приведенных форм.
2. [Внешний цикл по $b$]
   while($b \leqslant r$) {
   $m = (b^2 - D)/4$;
   for($1 \leqslant a$ and $a^2 \leqslant m$) {
   if($m \bmod a \neq 0$) continue; // Перейти к следующему шагу цикла «for», стремясь обеспечить условие $a|m$.
   $c = m/a$;
   if($b > a$) continue; // Перейти к следующему шагу цикла «for».
3. [Установка значений для варианта с многочленом]
   $\tau = (-b + i\sqrt{|D|})/(2a)$; // Обратить внимание на точность (см. последующий текст).
   $f = \Delta(e^{4\pi i\tau})/\Delta(e^{2\pi i\tau})$; // Обратить внимание на точность.
   $j = (256f + 1)^3/f$; // Обратить внимание на точность.
4. [Начать проверку делителей]
   if($b == a$ or $c == a$ or $b == 0$) {
   $T = T * (X - j)$;
   $h = h + 1$; // Счетчик классов.
   $red = red \cup (a,b,c)$; // Новая форма.
   } else {
   $T = T * (X^2 - 2\,\mathrm{Re}(j)X + |j|^2)$;
   $h = h + 2$; // Счетчик классов.
   $red = red \cup (a, \pm b, c)$; // Две новые формы.
   }
   }
   }
5. [Возвратить интересующие значения]
   return (комбинация объектов) $h, round(\mathrm{Re}(T[X])), red$;

Этот алгоритм прямолинеен во всех отношениях, за исключением того, что касается точности вычислений с плавающей точкой. Отметим, что функция $\Delta$ должна вычисляться для комплексных аргументов $q$. Теория показывает, что точностью, достаточной для всего алгоритма в целом, является, в сущности,

точность

$$\delta = \frac{\pi\sqrt{|D|}}{\ln 10}\sum\frac{1}{a}$$

десятичных знаков, где сумма берется по всем примитивным приведенным формам $(a,b,c)$ дискриминанта $D$ (см. [Atkin and Morain 1993b]). Это означает, что для выполнения фазы [Установка значений для варианта с многочленом] следует использовать немногим более, чем $\delta$ знаков (например, $\delta+10$, как в книге [Cohen 2000]), имея в виду, что в итоге многочлен $T[X]$, состоящий, возможно, из некоторого числа линейных множителей и некоторого числа оставшихся квадратичных множителей, должен иметь целые коэффициенты. Таким образом, окончательная выдача многочлена в виде $round(\mathrm{Re}(T[X]))$ означает, что в выражении для $T$ надо раскрыть скобки и округлить коэффициенты, так что мы будем иметь $T\in\mathbf{Z}[X]$. Алгоритм 7.5.8 можно, конечно, использовать и в многопроходном варианте — сначала вычислить одни только приведенные формы, чтобы оценить $\sum 1/a$ и тем самым требуемую точность, а затем запустить его вновь и на этот раз вычислить фактический гильбертов многочлен классов. Так или иначе, величина $\sum 1/a$ всегда есть $O\left(\ln^2|D|\right)$.

Для удобства читателя мы приводим здесь некоторые конкретные примеры многочленов, полученные с помощью этого алгоритма, где обозначение $T_D$ относится к гильбертову многочлену классов для дискриминанта $D$:

$$\begin{aligned}
T_{-3} &= X,\\
T_{-4} &= X-1728,\\
T_{-15} &= X^2+191025X-121287375,\\
T_{-23} &= X^3+3491750X^2-5151296875X+12771880859375.
\end{aligned}$$

Замечаем, что степени этих многочленов согласуются с приведенными ниже числами классов. У этих многочленов существуют и другие интересные аспекты. Один из них состоит в том, что свободный член всегда является кубом. К тому же, коэффициенты многочленов $T_D$ быстро растут по мере продвижения по спискам дискриминантов. Но в подходе Аткина—Морейна можно использовать менее громоздкие многочлены — так называемое веберово многообразие — ценой небольшого усложнения в частных случаях. Эти и многие другие оптимизации обсуждаются в работах [Morain 1990], [Atkin and Morain 1993b].

В схеме нахождения порядков Аткина—Морейна будет полезно представлять дискриминанты в порядке возрастания их чисел классов; это упорядочение является, по существу, упорядочением по возрастанию сложности. Как показали бы простые прогоны алгоритма 7.5.8 (скажем, в варианте без многочлена),

$$\begin{aligned}
h(D)=1 \text{ при } D &= -3,-4,-7,-8,-11,-19,-43,-67,-163;\\
h(D)=2 \text{ при } D &= -15,-20,-24,-35,-40,-51,-52,-88,-91,-115,\\
&\quad -123,-148,-187,-232,-235,-267,-403,-427;\\
h(D)=3 \text{ при } D &= -23,-31,-59,\ldots\ .
\end{aligned}$$

То, что списки дискриминантов для $h(D)=1,2$ в действительности приведены

здесь полностью, составляет глубокий результат теории (см. [Cox 1989]). В настоящий момент имеются полные списки для $h(D) \leqslant 16$, см. [Watkins 2000], и известно, по крайней мере в принципе, как вычислить полный список для любого наперед заданного значения $h$. Эффективное определение таких списков является чрезвычайно интересной вычислительной задачей.

Чтобы применять метод Аткина—Морейна, нам удобно будет рассматривать упорядочение дискриминантов, скажем, такое, как выше, т. е. ближе к началу будут те из них, у которых $h(D)$ меньше. Мы будем искать порядки кривых, связанные со специальными представлениями

$$4p = u^2 + |D|v^2,$$

в результате, как видно из приведенного ниже алгоритма, возможные порядки кривых будут простыми функциями от $p, u, v$. Отметим, что для $D = -3, -4$ возможны, соответственно, 6 порядков и 4 порядка, в то время как для остальных $D$ существуют два возможных порядка. Указанные представления числа $4p$ будем пытаться обнаружить с помощью алгоритма 2.3.13. Если число $p$ простое, «вероятность» того, что $4p$ представимо в таком виде при условии, что $\left(\frac{D}{p}\right) = 1$, равна $1/h(D)$, как упоминалось выше. В следующем алгоритме предполагается, что либо наш список дискриминантов — конечный, либо мы условились выполнять этот алгоритм в течение некоторого заранее заданного промежутка времени.

**Алгоритм 7.5.9** (метод комплексного умножения (CM) для генерации кривых и порядков). Мы предполагаем, что задан список фундаментальных дискриминантов $\{D_j < 0 : j = 1, 2, 3, \ldots\}$, упорядоченный, скажем, по возрастанию числа классов $h(D)$, а при совпадающем числе классов — по возрастанию $|D|$. Дано простое число $p > 3$. Этот алгоритм выдает либо возможные порядки кривых над полем $\mathbf{F}_p$ — эти кривые представлены CM-кривыми, связанными с различными $D_j$, — либо параметры этих CM-кривых, либо и то, и другое.

1. [Получить невычет]
   Найти случайный квадратичный невычет $g \pmod p$;
   if($p \equiv 1 \pmod 3$ and $g^{(p-1)/3} \equiv 1 \pmod p$) goto [Получить невычет];
   // В случае использования $D = -3$ элемент $g$ также не должен быть кубом
   // по модулю $p$.
   $j = 0$;
2. [Цикл для дискриминанта]
   $j = j + 1$;
   $D = D_j$;
   if( $\left(\frac{D}{p}\right) \neq 1$) goto [Цикл для дискриминанта];
3. [Поиск квадратичной формы для числа $4p$]
   Пытаемся обнаружить представление $4p = u^2 + |D|v^2$, используя алгоритм 2.3.13, а если эта попытка не удалась — перейти к шагу [Цикл для дискриминанта];
4. [Вариант: порядки кривых]
   if($D == -4$) report $\{p + 1 \pm u,\ p + 1 \pm 2v\}$; // 4 возможных порядка.

if($D == -3$) report $\{p+1 \pm u,\ p+1 \pm (u \pm 3v)/2\}$; // 6 возможных порядков.

if($D < -4$) report $\{p+1 \pm u\}$; // 2 возможных порядка.

5. [Вариант: параметры кривых]

if($D == -4$) report $\{(a,b)\} = \{(-g^k \bmod p, 0) : k = 0,1,2,3\}$;

if($D == -3$) report $\{(a,b)\} = \{(0, -g^k \bmod p) : k = 0,1,2,3,4,5\}$;

6. [Продолжение для $D < -4$]

Вычислить гильбертов многочлен классов $T \in \mathbf{Z}[X]$ посредством алгоритма 7.5.8;

$S = T \bmod p$; // Редукция к многочлену $\in \mathbf{F}_p[X]$.

Получить какой-либо корень $j \in \mathbf{F}_p$ многочлена $S$ посредством алгоритма 2.3.10;

$c = j(j-1728)^{-1} \bmod p$;

$r = -3c \bmod p$;

$s = 2c \bmod p$;

7. [Возвратить две пары параметров кривых]

return $\{(a,b)\} = \{(r,s), (rg^2 \bmod p, sg^3 \bmod p)\}$;

Метод Аткина—Морейна предписывает, что для $D = -4, -3$ кривые задаются, в терминах квадратичного невычета $g$, который является также кубическим невычетом в случае $D = -3$, уравнениями

$$\begin{aligned} y^2 &= x^3 - g^k x,\ k = 0,1,2,3, \\ y^2 &= x^3 - g^k,\ k = 0,1,2,3,4,5, \end{aligned}$$

соответственно (т. е. существуют, соответственно, 4 класса и 6 классов изоморфизмов кривых для двух данных значений $D$). В то время как для остальных дискриминантов $D$ соответствующая кривая и ее искажение суть

$$y^2 = x^3 - 3cg^{2k}x + 2cg^{3k},\ k = 0,1,$$

где $c$ задано в соответствии с шагом [Продолжение для $D < -4$]. Этот метод предусматривает гораздо большую общность по сравнению с такими алгоритмическими решениями для явного выражения, как приводимый ниже алгоритм 7.5.10, но с точки зрения реализации он сложнее, главным образом из-за вычисления гильбертова многочлена классов.

Отметим следующую важную деталь: еще до фактического вычисления параметров кривых мы уже знаем, какие имеются возможные порядки кривых. Поэтому и при доказательстве простоты и в криптографических приложениях мы можем проанализировать эти возможные порядки перед тем, как перейти к трудоемким вычислениям параметров $(a,b)$, зная при этом, что если порядок кривой по какой-либо причине привлечет наше внимание, то при желании мы сможем получить эти параметры. Мы обсудим этот аспект в разд. 7.6 и 8.1.

Разберем подробно один из примеров работы алгоритма 7.5.9. Возьмем простое число Мерсенна

$$p = 2^{89} - 1,$$

для которого мы хотим получить некоторые возможные кривые $E_{a,b}(\mathbf{F}_p)$ и их порядки. На приведенном выше шаге алгоритма [Поиск квадратичной формы

для числа $4p$] посредством алгоритма 2.3.13 мы находим большое количество представлений числа $4p$, лишь немногими из которых являются

$$\begin{aligned} 4p &= 48215832688019^2 + 3 \cdot 7097266064519^2 \\ &= 37064361490164^2 + 163 \cdot 2600275098586^2 \\ &= 35649086634820^2 + 51 \cdot 4860853432438^2 \\ &= 27347149714756^2 + 187 \cdot 3039854240322^2 \\ &= 28743118396413^2 + 499 \cdot 1818251501825^2. \end{aligned}$$

У этих выбранных представлений интересующими нас дискриминантами являются

$$\begin{aligned} D &= -3, \\ D &= -163, \\ D &= -51, \\ D &= -187,\ D = -499, \end{aligned}$$

соответственно. Повторяем, что существует большое количество других значений $D$, которые можно использовать для данного $p$. Соответствующими порядками кривых главным образом будут числа

$$p + 1 \pm u,$$

где $u$ — первое число, возводимое в квадрат в полученных представлениях. Хотя в случае $D = -3$ возможных порядков будет больше. Для иллюстрации подробностей работы алгоритма рассмотрим приведенный выше случай $D = -499$. Тогда на шаге [Вариант: параметры кривых] мы получаем

$$\begin{aligned} T_{-499} = \quad & 467113318239995478279867315443744131094937 6 \\ & - 60637178254942663947223925600110510 08x \\ & + 30051011080710262007067259699 20x^2 \\ & + x^3. \end{aligned}$$

Отметим, что как и должно быть, свободный член этого многочлена является кубом. А теперь можно немедленно произвести редукцию (mod $p$) этого кубического многочлена и получить

$$\begin{aligned} S = T_{-499} \bmod p = \quad & 489476008241378181249146744 \\ & + 35656028023043361329419482 5x \\ & + 16627057655833891019210 15x^2 \\ & + x^3; \end{aligned}$$

но мы здесь придерживаемся концепции, согласно которой можно, в принципе, предварительно сохранить заданное количество гильбертовых многочленов классов $T_{-D} \in \mathbf{Z}[X]$, быстро производя редукцию к многочлену $S \in \mathbf{F}_p[X]$ всякий раз, когда анализируется новое значение $p$. Далее нам предстоит использовать алгоритм 2.3.10 для нахождения корня $j$ многочлена $S = T \bmod p$.

Находим, что одним из корней будет

$$j = 4313021278160456153394518 68.$$

Именно это значение запускает процесс построения параметров кривых. Мы получаем

$$c = j/(j - 1728) \bmod p = 544175025087910210133176287$$

и по окончании имеем два уравнения (требуемый невычет $g$ для данного $p$ можно взять равным $-1$)

$$\begin{aligned} y^2 &= x^3 + 22438498366433978194915747 2x \pm \\ &\quad \pm 46938003053313028281679046 3 \end{aligned}$$

с соответствующими им порядками кривых

$$\#E = 2^{89} \pm 28743118396413.$$

Кстати, обычно легко выяснить, какой из порядков имеет каждая из кривых. Для заданных параметров $a, b$ надо найти точку $P \in E$ и убедиться в том, что равенство $[\#E]P = O$ выполнено для одного варианта выбора $\#E$ и не выполнено для другого. В действительности, если $p > 475$, из теоремы 7.5.2 следует, что либо существует точка $P$ на кривой $E$ такая, что $[\#E']P \neq O$ (где $E'$ есть искажение кривой $E$), либо существует точка $Q$ на кривой $E'$, такая, что $[\#E]Q \neq O$. Следовательно, случайным образом выбирая точки, сначала на одной из этих кривых, а потом на другой, можно ожидать, что вскоре удастся обнаружить, какой из порядков соответствует каждой из кривых. Во всяком случае, многие из алгоритмов, в основе которых лежит подход Аткина—Морейна, могут воспользоваться точками, которые просто имеют нулевые кратные, так что до конца выяснять порядок кривой необязательно.

Заметим, что вычисления с многочленом для дискриминантов с большой глубиной (т. е. для дискриминантов, обладающих большими числами классов) могут быть трудны. Например, существует фактор точности при использовании арифметики с плавающей точкой в алгоритме 7.5.8. Поэтому целесообразно предусмотреть средства для установления некоторых *явных* параметров кривых для малых $|D|$, тем самым устраняя необходимость вычислений, связанных с многочленом классов. С этой целью мы составили здесь полный список наборов параметров кривых для всех $D$, для которых $h(D) = 1, 2$; он приведен в табл. 7.1.

| $D$ | $r$ | $s$ |
|---|---|---|
| $-7$ | 125 | 189 |
| $-8$ | 125 | 98 |
| $-11$ | 512 | 539 |
| $-19$ | 512 | 512 |
| $-43$ | 512000 | 512001 |
| $-67$ | 85184000 | 85184001 |
| $-163$ | 151931373056000 | 151931373056001 |
| $-15$ | $1225 - 2080\sqrt{5}$ | 5929 |
| $-20$ | $108250 + 29835\sqrt{5}$ | 174724 |
| $-24$ | $1757 - 494\sqrt{2}$ | 1058 |
| $-35$ | $-1126400 - 1589760\sqrt{5}$ | 2428447 |
| $-40$ | $54175 - 1020\sqrt{5}$ | 51894 |
| $-51$ | $75520 - 7936\sqrt{17}$ | 108241 |
| $-52$ | $1778750 + 5125\sqrt{13}$ | 1797228 |
| $-88$ | $181713125 - 44250\sqrt{2}$ | 181650546 |
| $-91$ | $74752 - 36352\sqrt{13}$ | 205821 |
| $-115$ | $269593600 - 89157120\sqrt{5}$ | 468954981 |
| $-123$ | $1025058304000 - 1248832000\sqrt{41}$ | 1033054730449 |
| $-148$ | $499833128054750 + 356500625\sqrt{37}$ | 499835296563372 |
| $-187$ | $91878880000 - 1074017568000\sqrt{17}$ | 4520166756633 |
| $-232$ | $1728371226151263375 - 11276414500\sqrt{29}$ | 1728371165425912854 |
| $-267$ | $3632253349307716000000 - 12320504793376000\sqrt{89}$ | 3632369580717474122449 |
| $-403$ | $16416107434811840000 - 4799513373120384000\sqrt{13}$ | 33720998998872514077 |
| $-427$ | $564510997315289728000 - 578478561110278400\sqrt{61}$ | 609691617259594724421 |

**Таблица 7.1.** Явные параметры СМ-кривых для числа классов 1 и 2

**Алгоритм 7.5.10** (явные параметры СМ-кривых: числа классов 1, 2). При задании простого числа $p > 3$ этот алгоритм выдает в явном виде СМ-кривые $y^2 = x^3 + ax + b$ над полем $\mathbf{F}_p$, которые имеют порядки, указанные на шаге [Вариант: порядки кривых] алгоритма 7.5.9. Здесь производится исчерпывающий поиск по всем дискриминантам $D$ с числом классов $h(D) = 1, 2$: алгоритм выдает каждый набор параметров $(a, b)$ СМ-кривых для предусмотренных здесь чисел классов.

1. [Установить полный список дискриминантов]
$$\Delta = \{-3, -4, -7, -8, -11, -19, -43, -67, -163,$$
$$-15, -20, -24, -35, -40, -51, -52, -88, -91, -115, -123,$$
$$-148, -187, -232, -235, -267, -403, -427\};$$

2. [Цикл по представлениям]
   for($D \in \Delta$) {
   Пытаемся обнаружить представление $4p = u^2 + |D|v^2$, используя алгоритм 2.3.13, а если эта попытка не удалась, перейти к следующему $D$;
   Выбрать подходящий невычет $g$ по модулю $p$ таким же образом, как это делается на шаге [Получить невычет] алгоритма 7.5.9;

3. [Обработать $D = -3, -4$]
   if($D == -3$) return $\{(a,b)\} = \{(0, -g^k) : k = 0, \ldots, 5\}$;
   // Шесть кривых $y^2 = x^3 - g^k$.
   if($D == -4$) return $\{(a,b)\} = \{(-g^k, 0) : k = 0, \ldots, 3\}$;
   // Четыре кривые $y^2 = x^3 - g^k x$.

4. [Параметры для всех остальных $D$, для которых $h(D) = 1, 2$]
   Выбрать пару $(r, s)$ из приведенной таблицы, используя алгоритм 2.3.9, если требуются квадратные корни (mod $p$);

5. [Возвратить параметры кривых]
   report $\{(a,b)\} = \{(-3rs^3g^{2k} \bmod p, 2rs^5g^{3k} \bmod p) : k = 0, 1\}$;
   // Задающим кривую уравнением будет $y^2 = x^3 - 3rs^3g^{2k}x + 2rs^5g^{3k}$.
   }

Существует несколько вопросов, представляющих интерес в связи с этим алгоритмом. Приведенные в явном виде параметризации алгоритма 7.5.10 могут быть вычислены, конечно, с помощью гильбертовых многочленов классов, как это делается в алгоритме 7.5.8. Однако, размещение этих параметров в явном виде означает, что можно перейти к установлению CM-кривых очень быстро, с минимальными программными затратами. Необязательно даже проверять, что $4a^3 + 27b^2 \neq 0$, как того требуют правила для эллиптических кривых над полем $\mathbf{F}_p$. Еще одной интересной особенностью является то, что соответствующие квадратные корни, присутствующие в данном алгоритме, всегда существуют (mod $p$). Кроме того, многие из представленных в таблице параметров $r, s$ обладают интересными разложениями на множители. В частности, многие значения $s$ обладают высокой степенью гладкости (см. упр. 7.15 для ознакомления с дальнейшими подробностями).

Настало время пояснить на примере уже решенной задачи, насколько быстро алгоритм 7.5.10 будет генерировать кривые и порядки. Взяв простое число $p = \left(2^{31} + 1\right)/3$, находим, посредством вызова алгоритма 2.3.13, представления вида $4p = u^2 + |D|v^2$ для десяти дискриминантов $D$ с числом классов, не превосходящим двух, а именно, для $D = -3, -7, -8, -11, -67, -51, -91, -187, -403, -427$. Соответствующие параметры $a, b$ и порядки кривых выдаются посредством алгоритма 7.5.10 и представлены на таблице.

| $D$ | $E$ | $\#E$ |
|---|---|---|
| $-3$ | $y^2 = x^3 + 0x + 715827882$ | 715861972 |
| | $y^2 = x^3 + 0x + 715827878$ | 715880649 |
| | $y^2 = x^3 + 0x + 715827858$ | 715846561 |
| | $y^2 = x^3 + 0x + 715827758$ | *715793796* |
| | $y^2 = x^3 + 0x + 715827258$ | 715775119 |
| | $y^2 = x^3 + 0x + 715824758$ | 715809207 |
| $-7$ | $y^2 = x^3 + 331585657x + 632369458$ | 715788584 |
| | $y^2 = x^3 + 415534712x + 305115120$ | 715867184 |
| $-8$ | $y^2 = x^3 + 362880883x + 649193252$ | 715784194 |
| | $y^2 = x^3 + 482087479x + 260605721$ | 715871574 |
| $-11$ | $y^2 = x^3 + 710498587x + 673622741$ | 715774393 |
| | $y^2 = x^3 + 582595483x + 450980314$ | 715881375 |
| $-67$ | $y^2 = x^3 + 265592125x + 480243852$ | 715785809 |
| | $y^2 = x^3 + 197352178x + 616767211$ | 715869959 |
| $-51$ | $y^2 = x^3 + 602207293x + 487817116$ | 715826683 |
| | $y^2 = x^3 + 22796782x + 131769445$ | 715829085 |
| $-91$ | $y^2 = x^3 + 407640471x + 205746226$ | 715824963 |
| | $y^2 = x^3 + 169421413x + 664302345$ | *715830805* |
| $-187$ | $y^2 = x^3 + 389987874x + 525671592$ | 715817117 |
| | $y^2 = x^3 + 443934371x + 568611647$ | 715838651 |
| $-403$ | $y^2 = x^3 + 644736647x + 438316263$ | 715881357 |
| | $y^2 = x^3 + 370202749x + 386613767$ | 715774411 |
| $-427$ | $y^2 = x^3 + 370428023x + 532016446$ | 715860684 |
| | $y^2 = x^3 + 670765979x + 645890514$ | 715795084 |

Для данного конкретного запуска требуемый квадратичный невычет (являющийся также кубическим невычетом, что необходимо для случая $D = -3$) был выбран равным 5. Отметим, что алгоритм 7.5.10 ничего не говорит нам о том, какая из пар параметров $(a, b)$ этих кривых соответствует каждому из порядков (которые были взяты из шага [Вариант: порядки кривых] алгоритма 7.5.9). Как упоминалось выше, в этом нет серьезной проблемы — надо найти точку $P$ на одной из кривых, такую, что предполагаемый порядок не обнуляет ее, откуда будет следовать, что данное значение порядка отвечает другой кривой.

В приведенной таблице для $p = (2^{31}+1)/3$ соответствие между кривыми и порядками было найдено именно так.

Но можно, в принципе, пойти еще дальше и указать теоретически, какой из порядков будет соответствовать каждой из кривых, по крайней мере для тех дискриминантов $D$, для которых $h(D) = 1$. Явно выписанные кривые и порядки существуют в литературе (см. [Rishi et al. 1984], [Padma and Ventkataraman 1996]). Многие из подобных результатов восходят к работе Старка, который связал точное значение порядка кривой, $p + 1 - u$, где $4p = u^2 + |D|v^2$, а $u$ может быть положительным или отрицательным, с символом Якоби $\left(\frac{u}{|D|}\right)$. Интересные уточнения этой работы можно найти в современном изложении у Морейна (см. [Morain 1998]).

## 7.6. Доказательство простоты при помощи эллиптических кривых

В разд. 4.1 мы увидели, что частичное разложение на множители числа $n-1$ может привести к доказательству простоты числа $n$. Возникает вопрос: могут ли группы эллиптических кривых, порядки которых распределены в соответствии с теоремой Хассе 7.3.1, применяться для доказательства простоты? Да, действительно могут, о чем свидетельствует одна теорема, которая является своего рода аналогом теоремы Поклингтона 4.1.3 в случае эллиптических кривых.

Прежде чем мы приведем эту теорему, нужно вспомнить понятие псевдокривой $E(\mathbf{Z}_n)$ из определения 7.4.1. Вспомнив также предостережение относительно эллиптического умножения на псевдокривой, упомянутое вслед за этим определением, перейдем к следующему центральному результату[8].

**Теорема 7.6.1** (ECPP-теорема Гольдвассер—Килиана). *Пусть $n > 1$ — целое число, взаимно простое с числом 6, $E(\mathbf{Z}_n)$ — псевдокривая, $s, m$ — натуральные числа, такие, что $s|m$. Предположим, что существует точка $P \in E$, такая, что мы можем осуществить операции на кривой, необходимые для вычисления $[m]P$, и обнаружить, что*

$$[m]P = O,$$

*и для каждого простого числа $q$, делящего $s$, мы можем осуществить операции на кривой и получить, что*

$$[m/q]P \neq O.$$

*Тогда для каждого простого числа $p$, делящего $n$, справедливо сравнение*

$$\#E(\mathbf{F}_p) \equiv 0 \pmod{s}.$$

*Если, сверх того, $s > \left(n^{1/4}+1\right)^2$, то число $n$ — простое.*

[8] ECPP — elliptic curve primality proving, доказательство простоты при помощи эллиптических кривых (*англ.*). — *Прим. перев.*

Доказательство. Пусть $p$ — простой делитель числа $n$. Соотношения, полученные в результате вычислений на псевдокривой, если рассмотреть их по модулю $p$, указывают на то, что $s$ делит порядок точки $P$ на кривой $E(\mathbf{F}_p)$. Это доказывает первое утверждение. Кроме того, если $s > \left(n^{1/4}+1\right)^2$, мы можем заключить, что $\#E(\mathbf{F}_p) > \left(n^{1/4}+1\right)^2$. Однако из теоремы Хассе 7.3.1 следует, что $\#E(\mathbf{F}_p) < \left(p^{1/2}+1\right)^2$. Отсюда получаем, что $p^{1/2} > n^{1/4}$, так что $p > n^{1/2}$. Поскольку все простые делители числа $n$ больше, чем корень из $n$, $n$ должно быть простым. □

## 7.6.1. Тест на простоту Гольдвассер—Килиана

На основании теоремы 7.6.1 Гольдвассер и Килиан указали алгоритм проверки на простоту с ожидаемой полиномиальной сложностью, которая предполагается для всех и строго доказана для «большинства» простых чисел $n$. Т. е. ожидаемое число операций на обработку числа $n$ есть $O\left(\ln^k n\right)$ при абсолютной константе $k$. Их идея состоит в нахождении подходящих кривых, порядки которых имеют достаточно большие «вероятные простые» делители, и проведении рекурсии, исходя из того, что необходимо, в свою очередь, строго доказать простоту этих делителей. На каждом уровне рекурсии, кроме последнего, используется теорема 7.6.1, в которой $s$ должно быть равным вероятному простому делителю порядка кривой. В процессе рекурсии вероятные простые числа становятся все меньше и меньше, и так продолжается до тех пор, пока такое число не станет настолько малым, что его простоту можно доказать методом пробных делений. Это доказательство, в свою очередь, обосновывает все предыдущие шаги и устанавливает факт простоты исходного числа $n$.

**Алгоритм 7.6.2** (тест на простоту Гольдвассер—Килиана). Получив на входе не являющееся квадратом целое число $n > 2^{32}$, предположение о простоте которого имеет под собой серьезные основания (в частности, $\text{НОД}(n, 6) = 1$, и, по возможности, $n$ уже прошло тест на вероятную простоту), этот алгоритм пытается свести вопрос о простоте числа $n$ к вопросу о простоте меньшего числа $q$. Данный алгоритм возвращает либо утверждение «$n$ — составное», либо утверждение «если $q$ — простое, то $n$ — простое», где $q$ — целое число, меньшее $n$.

1. [Выбрать псевдокривую над кольцом $\mathbf{Z}_n$]
   Выбрать случайную пару $(a, b) \in [0, n-1]^2$, такую что $\text{НОД}(4a^3 + 27b^2, n) = 1$;
2. [Установить порядок кривой]
   Посредством алгоритма 7.5.6 вычислить целое число $m$, которое было бы порядком $\#E_{a,b}(\mathbf{Z}_n)$, если бы число $n$ было простым (если же алгоритм подсчета точек завершается некорректно, возвратить «$n$ — составное»);
   // Если число $n$ — составное, то алгоритм 7.5.6 завершается некорректно, если каждый вариант значения $t \pmod l$ отвергается,

или если окончательный порядок кривой не лежит в интервале $(n+1-2\sqrt{n}, n+1+2\sqrt{n})$.

3. [Попытка разложения]
Попытаться получить разложение вида $m = kq$, где $k > 1$, и $q$ — вероятное простое число, превосходящее $\left(n^{1/4}+1\right)^2$; если такое разложение не удается получить за время, определяемое некоторым критерием, то перейти к шагу [Выбрать псевдокривую ...];

4. [Выбрать точку на псевдокривой $E_{a,b}(\mathbf{Z}_n)$]
Выбрать случайное $x \in [0, n-1]$, для которого элемент $Q = (x^3 + ax + b) \bmod n$ таков, что $\left(\frac{Q}{n}\right) \neq -1$;
Применить алгоритм 2.3.8 или алгоритм 2.3.9 (в которых $a = Q$ и $p = n$) для нахождения целого числа $y$, которое удовлетворяло бы сравнению $y^2 \equiv Q \pmod{n}$, если бы число $n$ было простым;
if($y^2 \bmod n \neq Q$) return «$n$ — составное»;
$P = (x, y)$;

5. [Операции с точкой]
Вычислить кратное $U = [m/q]P$ (если же мы столкнемся с невыполнимой операцией взятия обратного, то возвратить «$n$ — составное»);
if($U == O$) goto [Выбрать точку ...];
Вычислить $V = [q]U$ (соблюдая при этом указанное выше правило относительно вычисления обратных);
if($V \neq O$) return «$n$ — составное»;
return «если $q$ — простое, то $n$ — простое»;

Корректность алгоритма 7.6.2 сразу следует из теоремы 7.6.1, в которой роль числа $s$ играет число $q$.

На практике нужно проводить итерации этого алгоритма, получая цепочку суждений, в которой последнее число $q$ настолько мало, что его простоту можно доказать методом пробных делений. Если некоторое промежуточное число $q$ — составное, то можно вернуться к предыдущему звену этой цепочки и снова применить этот алгоритм. Итерации схемы Гольдвассер—Килиана не только приводят к строгому тесту на простоту, но и генерируют сертификат простоты. Этот сертификат можно представлять себе в виде цепочки

$$(n = n_0, a_0, b_0, m_0, q_0, P_0),\ (q_0 = n_1, a_1, b_1, m_1, q_1, P_1), \ldots,$$

состоящей из последовательных записей $n, a, b, m, q, P$, отражающих весь ход рекурсии. Основным достоинством этого сертификата является то, что мы можем опубликовать его вместе с исходным числом $n$, простота которого доказана, или можем как-то еще сослаться на него. Эта сжатая информация может затем использоваться всеми, кто желает убедиться в простоте числа $n$; при этом на каждом шаге используется теорема 7.6.1. Такая реконструкция доказательства обычно требует значительно меньше времени, чем первоначальный запуск, находящий данный сертификат. Эта особенность сертификата является нетривиальной, поскольку многие доказательства простоты можно проверить, лишь запустив их вновь с самого начала.

Следует отметить, что эллиптическую арифметику в алгоритме 7.6.2 можно ускорить с помощью координат Монтгомери $[X : Z]$ с опущенным «$Y$», в соответствии со сказанным в разд. 7.2.

Чтобы помочь читателю в тестировании любой из реализаций, подробно разберем один пример. Возьмем простое число $p = 10^{20} + 39$. При первом проходе алгоритма 7.6.2 мы используем $n = p$ и получаем на шаге [Выбрать псевдокривую...] случайные параметры, скажем,

$$a = 69771859804340235254, \quad b = 10558409492409151218,$$

для которых число $4a^3 + 27b^2$ взаимно просто с числом $n$. Посредством алгоритма 7.5.6 находим число, которое было бы порядком кривой $E_{a,b}(\mathbf{Z}_n)$, если бы $n$ действительно было простым:

$$m = \#E = 99999999985875882644 = 2^2 \cdot 59 \cdot 1182449 \cdot q,$$

при этом известно, что числа $2, 59, 1182449$ — простые (не превышающие границы $2^{32}$, предложенной в описании алгоритма), а $q = 358348489871$ — вероятное простое число. Далее, на шаге [Выбрать точку...] полученной нами случайной точкой будет, например,

$$P = [X : Z] = [31689859357184528586 : 1],$$

где с целью упрощения реализации мы приняли параметризацию Монтгомери, намереваясь использовать алгоритм 7.2.7 для вычисления эллиптических кратных. Так было найдено, что

$$\begin{aligned} U &= [m/q]P = [69046631243878263311 : 1] \neq O, \\ V &= [q]U = O. \end{aligned}$$

Следовательно, число $p$ является простым, если таковым является $q$. Так что теперь мы берем $n = 358348489871$ и снова запускаем алгоритм 7.6.2. В процессе работы были получены соответствующие значения

$$\begin{aligned} a &= 34328822753, \quad b = 187921935449, \\ m &= \#E = 358349377736 = 2^3 \cdot 7 \cdot 7949 \cdot 805019, \end{aligned}$$

где на этот раз все делители не превышают нашей границы $2^{32}$. Для случайно выбранной начальной точки

$$P = [X : Z] = [245203089935 : 1]$$

получаем, при $q = 805019$,

$$\begin{aligned} U &= [m/q]P = [260419245130 : 1] \neq O, \\ V &= [q]P = O. \end{aligned}$$

Отсюда следует, что первоначальное число $p = 10^{20} + 39$ является простым. Из возникающих чисел затем составляется сертификат простоты данного простого числа. Следует отметить, что если брать в качестве примера бóльшие числа, то вряд ли нам повезет так, что в результате каждой попытки мы будем получать хорошее разложение числа $m$, однако существует предположение, что данное событие происходит не так уж редко.

Изучение вычислительной сложности алгоритма 7.6.2 весьма интересно. Успех алгоритма зависит от вероятности нахождения такого порядка кривой, который имеет разложение, указанное на шаге [Попытка разложения]. Заметим, что мы будем удовлетворены, даже если найдем лишь порядок вида $m = 2q$, где $q$ — простое. Таким образом, с помощью теоремы 7.3.2 можно показать, что если

$$\pi\left(x+1+2\sqrt{x}\right)-\pi\left(x+1-2\sqrt{x}\right)>A\frac{\sqrt{x}}{\ln^c x}$$

при положительных константах $A$, $c$, то ожидаемая битовая сложность данного алгоритма составляет $O\left(\ln^{9+c} n\right)$ (см. [Goldwasser and Kilian 1986]). Существует гипотеза, что это неравенство выполнено для $A = c = 1$ и всех достаточно больших значений $x$. Кроме того, используя результаты из аналитической теории чисел, говорящие о том, что такие неравенства *обычно* верны, можно показать, что тест Гольдвассер—Килиана (алгоритм 7.6.2) обычно срабатывает, причем за полиномиальное время. Чтобы восполнить возникающий пробел в обосновании, мы могли бы заметить, что у нас *есть* достаточная информация о простых в интервале длины $x^{3/4}$ вблизи $x$. Используя это обстоятельство, Адлеману и Хуангу (см. [Adleman and Huang 1992]) удалось получить гарантированную ожидаемую полиномиальную оценку сложности. В их схеме цепочка-сертификат генерируется сходным образом, хотя, что примечательно, простые числа в этой цепочке сначала растут, но в конечном итоге начинают убывать до приемлемого уровня. Это убывание происходит за счет использования теста Гольдвассер—Килиана, как и выше, а возрастание устраивается с тем, чтобы «добиться случайности». Первоначальный кандидат $n$ может быть таким, для которого тест Гольдвассер—Килиана не работает (по той причине, что нам никогда не удастся разложить порядки кривых, или просто разложение потребует слишком много времени), так что первоначальные шаги, «сводящие» простоту числа $n$ к простоте бóльших чисел, являются одним из способов замены заданного числа $n$ новым числом, достаточно случайным, чтобы мы могли ожидать, что для него сработает тест Гольдвассер—Килиана. Данный «подъем» осуществляется с помощью якобиевых многообразий гиперэллиптических кривых рода 2.

## 7.6.2. Тест на простоту Аткина—Морейна

Алгоритм Гольдвассер—Килиана 7.6.2 на практике заметно замедляется при больших $n$ из-за стадии подсчета числа точек $\#E$. Аткин нашел изящное решение этой проблемы и совместно с Морейном реализовал высокоэффективную схему доказательства простоты при помощи эллиптических кривых (ECPP) (см. [Atkin and Morain 1993b]). В настоящее время данный метод получил широкое распространение. Существуют различные варианты практической реализации ECPP. Здесь мы приводим лишь один из них.

Вновь идея состоит в том, чтобы либо находить «явно выражаемые» порядки кривых, либо по крайней мере иметь возможность указывать поряд-

ки сравнительно быстро. Можно рассмотреть использование явных формул, наподобие указанных в алгоритме 7.5.10, однако вполне можно «исчерпать свой запас топлива», тщетно пытаясь найти порядок с подходящей для теоремы 7.6.1 структурой. Подход Аткина—Морейна состоит в нахождении кривых с комплексным умножением, таких, как в алгоритме 7.5.9. Таким образом, ключевая стадия (называемая в алгоритме 7.6.2 [Установить порядок кривой]) содержит точку входа в алгоритм Аткина—Морейна 7.5.9 для нахождения порядков/кривых. Сразу бросается в глаза поразительное сходство приводимого ниже алгоритма 7.6.3 и алгоритма 7.6.2. Разница между ними в том, что здесь мы *сначала* ищем подходящие порядки кривых, и только потом строим соответствующую эллиптическую кривую, используя на каждом из этапов алгоритм 7.5.9 и обходясь без алгоритма Шуфа 7.5.6.

**Алгоритм 7.6.3** (тест на простоту Аткина—Морейна). Получив на входе не являющееся квадратом целое число $n > 2^{32}$, предположение о простоте которого имеет под собой серьезные основания (в частности, $\text{НОД}(n, 6) = 1$, и, предположительно, $n$ уже прошло тест на вероятную простоту), данный алгоритм пытается свести вопрос о простоте числа $n$ к вопросу о простоте меньшего числа $q$. Алгоритм возвращает либо утверждение «$n$ — составное», либо утверждение «если $q$ — простое, то $n$ — простое», где $q$ — целое число, меньшее $n$. (Обратите внимание на сходную структуру алгоритма 7.6.2.)

1. [Выбрать дискриминант]
   Выбрать фундаментальный дискриминант $D$, начиная с малых значений $h(D)$, для которого $\left(\frac{D}{n}\right) = 1$, и для которого посредством алгоритма 2.3.13 нам удалось найти решение уравнения $u^2 + |D|v^2 = 4n$ и получить возможные порядки кривых $m$:
   $m \in \{n + 1 \pm u,\ n + 1 \pm 2v\}$ для $D = -4$,
   $m \in \{n + 1 \pm u,\ n + 1 \pm (u \pm 3v)/2\}$ для $D = -3$,
   $m \in \{n + 1 \pm u\}$ для $D < -4$;
2. [Разложить порядки]
   Найти возможный порядок $m$, имеющий разложение вида $m = kq$, где $k > 1$ и $q$ — вероятное простое число, большее $(n^{1/4} + 1)^2$ (если же этого не удалось сделать в течение заданного промежутка времени, то перейти к шагу [Выбрать дискриминант]);
3. [Получить параметры кривой]
   Используя вариант алгоритма 7.5.9 с генерацией параметров, установить параметры $a, b$ эллиптической кривой, которая имела бы порядок $m$, если бы число $n$ действительно было простым;
4. [Выбрать точку на кривой $E_{a,b}(\mathbf{Z}_n)$]
   Выбрать случайное $x \in [0, n - 1]$, для которого элемент $Q = (x^3 + ax + b) \bmod n$ таков, что $\left(\frac{Q}{n}\right) \neq -1$;
   Применить алгоритм 2.3.8 или 2.3.9 (в котором $a = Q$ и $p = n$) для нахождения целого числа $y$, которое удовлетворяло бы сравнению $y^2 \equiv Q \pmod{n}$, если бы число $n$ было простым;
   if($y^2 \bmod n \neq Q$) return «$n$ — составное»; $P = (x, y)$;

5. [Операции с точкой]
   Вычислить кратное $U = [m/q]P$ (если же при вычислении этого кратного встретилась невыполнимая операция взятия обратного, то возвратить «$n$ — составное»);
   if($U == O$) goto [Выбрать точку...];
   Вычислить $V = [q]U$ (соблюдая то же самое правило о вычислении обратных);
   if($V \neq O$) return «$n$ — составное»;
   return «если $q$ — простое, то $n$ — простое»;

Заметим, что если $n$ — составное, то нельзя гарантировать, что алгоритму 2.3.13, вызванному на стадии [Выбрать дискриминант] удастся найти $u, v$, даже если они существуют. В таком случае мы переходим к следующему $D$, пока, наконец, не добьемся успеха, или не сдадимся, потеряв терпение.

Разберем конкретный пример. Вспомним простое число Мерсенна $p = 2^{89} - 1$, рассмотренное вслед за алгоритмом 7.5.9. Мы обнаружили для кривых с комплексным умножением дискриминант $D = -3$, и как оказалось, для этого $D$ существуют шесть возможных порядков кривых. В данном случае рекурсивное доказательство простоты будет работать, если в качестве порядка взять $p + 1 + u$; оказывается, этот выбор годится на каждом уровне рекурсии, а именно:

$$\begin{aligned} p &= 2^{89} - 1, \\ D &= -3: \ u = 34753815440788, \ v = 20559283311750, \\ \#E &= p + 1 + u = 2^2 \cdot 3^2 \cdot 5^2 \cdot 7 \cdot 848173 \cdot p_2, \\ p_2 &= 115836285129447871, \\ D &= -3: \ u = 557417116, \ v = 225559526, \\ \#E &= p_2 + 1 + u = 2^2 \cdot 3 \cdot 7 \cdot 37 \cdot 65707 \cdot p_3, \end{aligned}$$

а простоту числа $p_3 = 567220573$ мы устанавливаем методом пробных делений. То, что мы привели, является, по существу, «стержнем» сертификата простоты числа $p = 2^{89} - 1$. Для полного сертификата, разумеется, необходимы фактические параметры кривых (из шага [Получить параметры кривой]) и соответствующие начальные точки (из шага [Выбрать точку...]), полученные в алгоритме 7.6.3.

Если сравнивать с подходом Гольдвассер—Килиана, вопрос о сложности метода Аткина—Морейна представляется более туманным, хотя эвристические оценки и являются полиномиальными, например, $O(\ln^{4+\epsilon} N)$ операций для доказательства простоты числа $N$ (см. разд. 7.6.3). Дополнительную сложность привносит то обстоятельство, что возможные порядки кривых, которые мы пытаемся разложить, имеют неизвестное распределение. Однако на практике этот метод превосходно работает и, как и метод Гольдвассер—Килиана, предоставляет полноценный краткий сертификат простоты. Реализованные Морейном варианты алгоритма 7.6.3 позволили получить доказательства простоты

«случайных» простых чисел, состоящих более чем из двух тысяч десятичных знаков, о чем мы уже упоминали в разд. 1.1.2. Но оказалось, что можно добиться еще большей эффективности; этот вопрос мы обсудим ниже.

### 7.6.3. Быстрое доказательство простоты при помощи эллиптических кривых (fastECPP)

Новое направление исследований в области доказательств простоты позволило получить доказательства для ряда гигантских чисел. Например, в июле 2004 г. была установлена простота числа Лейланда (имеющего вид $x^y + y^x$)

$$N = 4405^{2638} + 2638^{4405},$$

состоящего из 15071 десятичного знака. Этот «fastECPP-метод» основан на асимптотическом улучшении, принадлежащем Шаллиту, которое дает эвристическую оценку битовой сложности доказательства простоты числа $N$ в виде $O(\ln^{4+\epsilon} N)$.

Основная идея заключается в построении базы небольших квадратных корней и построении дискриминантов на ее основе. Пусть $L = \ln N$, где $N$ — рассматриваемое нами возможное простое число. Тогда алгоритм 7.6.3 требует, как мы ожидаем, $O(L^2)$ попыток выбора дискриминантов $D$ перед тем, как будет найдено подходящее $D$. Вместо этого мы можем строить дискриминанты вида $-D = (-p)(q)$, где $p, q$ — простые, взятые из множеств, размер которых есть $O(L)$. Таким способом можно ускорить шаг [Выбрать дискриминант], так что общая сложность алгоритма 7.6.3 в операциях, которая первоначально составляла $O(\ln^{5+\epsilon} N)$, будет снижена, и число 5 в показателе превратится в 4.

Подробности и разнообразные рекорды в области доказательств простоты можно найти в работах [Franke et al. 2004] и (в той части, что касается теории метода fastECPP) [Morain 2004].

## 7.7. Упражнения

**7.1.** Найти билинейное преобразование вида

$$(x, y) \mapsto (\alpha x + \beta y, \gamma x + \delta y),$$

которое приводит кривую

$$y^2 + axy + by = x^3 + cx^2 + dx + e \tag{7.11}$$

к форме Вейерштрасса (7.4). Далее, укажите, где при обосновании законности этого преобразования используется тот факт, что характеристика поля отлична от 2 и 3.

**7.2.** Покажите, что кривая, задаваемая уравнением

$$Y^2 = X^3 + CX^2 + AX + B,$$

имеет аффинное представление

$$y^2 = x^3 + (A - C^2/3)x + (B - AC/3 + 2C^3/27).$$

Отсюда видно, что кривая Монтгомери ($B = 0$) всегда обладает аффинным эквивалентом. Но обратное неверно. Опишите в точности те условия на параметры $a, b$ в уравнении

$$y^2 = x^3 + ax + b,$$

при которых данная аффинная кривая будет обладать эквивалентом Монтгомери с $B = 0$. Опишите приложения этого результата, например, в криптографии или при подсчете точек.

**7.3.** Покажите, что кривая, заданная соотношением (7.4), является неособой над полем $F$ характеристики $\neq 2, 3$ тогда и только тогда, когда $4a^3 + 27b^2 \neq 0$.

**7.4.** В соответствии с упр. 7.3 условие неособости аффинных кривых состоит в том, что дискриминант $4a^3 + 27b^2$ отличен от нуля в поле $\mathbf{F}_p$. Покажите, что для параметризации

$$Y^2 = X^3 + CX^2 + AX + B$$

и характеристики $p > 3$ условие неособости формулируется в терминах дискриминанта $\Delta$,

$$\Delta = 4(A - C^2/3)^3 + 27(B - AC/3 + 2C^3/27)^2 \neq 0.$$

Далее, покажите, что в случае удобной для вычислений параметризации Монтгомери кривая

$$Y^2 = X^3 + CX^2 + X$$

неособа тогда и только тогда, когда $C^2 \neq 4$.

**7.5.** Для эллиптической кривой над полем $\mathbf{F}_p$, $p > 3$ с уравнением

$$Y^2 = X^3 + CX^2 + AX + B$$

определим $j$-инвариант кривой $E$ формулой

$$j(E) = 4096\ \frac{(C^2 - 3A)^3}{\Delta},$$

где дискриминант $\Delta$ приведен в упр. 7.4. Проделайте следующее упражнение вычислительного характера. Выбрав для удобства малое простое число, которое допускает проведение вычислений вручную или легко обрабатывается на компьютере (Вы можете пользоваться простейшей формулой (7.8) для установления порядков кривых), составьте таблицу порядков кривых и их $j$-инвариантов. На основании этих эмпирических данных выявите взаимосвязь между порядками кривых и значениями $j$-инвариантов. Превосходный обзор удивительной теории $j$-инвариантов и изоморфизмов кривых см. в книге [Seroussi et al. 1999]; см. также указанную в ней разнообразную литературу, в особенности книгу [Silverman 1986].

**7.6.** Здесь мы рассмотрим лишь небольшую часть красивой классической теории эллиптических интегралов и функций с намерением проследить связи последней с современной теорией эллиптических кривых. Хорошим введением в этот круг вопросов служат работы [Namba 1984], [Silverman 1986], [Kaliski

1988]. Одна из существенных связей состоит в замечании Вейерштрасса, согласно которому эллиптический интеграл

$$Z(x) = \int_x^\infty \frac{ds}{\sqrt{4s^3 - g_2 s - g_3}}$$

можно рассматривать как функцию, неявно заданную уравнением

$$\wp_{g_2,g_3}(Z) = x,$$

где $\wp$ — функция Вейерштрасса. Выведите, далее, дифференциальные уравнения

$$\wp(z_1 + z_2) = \frac{1}{4}\left(\frac{\wp'(z_1) - \wp'(z_2)}{\wp(z_1) - \wp(z_2)}\right)^2 - \wp(z_1) - \wp(z_2)$$

и

$$\wp'(z)^2 = \wp^3(z) - g_2\wp(z) - g_3$$

и укажите, как необходимо связать параметры $g_2, g_3$ с аффинными параметрами кривой $a, b$, чтобы дифференциальная схема стала эквивалентна аффинной схеме.

**7.7.** Докажите первое утверждение теоремы 7.1.3, состоящее в том, что $E_{a,b}(F)$ вместе с указанными операциями есть абелева группа. Здесь может пригодиться хорошая программа-процессор для символьных вычислений абстрактной алгебры, особенно это касается наиболее трудоемкой части, связанной с доказательством ассоциативности $(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$.

**7.8.** Покажите, что абелева группа, порядок которой свободен от квадратов, является циклической. Сделайте вывод, что если порядок кривой $\#E$ свободен от квадратов, то группа эллиптической кривой — циклическая. Этот аспект играет важную роль в криптографических приложениях (см. [Kaliski 1991], [Morain 1992]).

**7.9.** Сравните количества операций (учитываются только умножения) в алгоритмах 7.2.2 и 7.2.3 с целью сопоставления эффективности этих алгоритмов при удвоении и сложении (неравных точек). При этом определите границу $k$, такую, что если операция взятия обратного будет быстрее, чем $k$ умножений, то первый алгоритм будет лучше. В этой связи см. упр. 7.25.

**7.10.** Покажите, что если мы договорились о том, что в поле параметр $a = -3$, то число арифметических действий в операции удвоения из алгоритма 7.2.3 можно сократить еще больше. Исследуйте утверждение Солинаса (см. [Solinas 1998]), которое гласит, что «доля эллиптических кривых по модулю $p$, которые допускают приведение к виду с $a = p - 3$ составляет примерно $1/4$, если $p \equiv 1 \pmod 4$, и примерно $1/2$, если $p \equiv 3 \pmod 4$». Попутно заметим, что хотя это незначительное ускорение операции удвоения может показаться тривиальным, на практике его всегда следует иметь в виду, поскольку операции удвоения составляют значительную долю в типичной схеме умножения точек.

**7.11.** Докажите, что тест на эллиптическую сумму, алгоритм 7.2.8, работает корректно. Во-первых, установите для координат $x_\pm$ точек $P_1 \pm P_2$, соответственно, алгебраические соотношения, которым удовлетворяют их сумма

$x_+ + x_-$ и произведение $x_+x_-$, используя для этого определение 7.1.2 и теорему 7.2.6. В получившихся соотношениях зависимость от $y$ должна исчезнуть. А теперь, исходя из полученных соотношений для суммы и произведения, выпишите требуемое квадратное уравнение.

**7.12.** Проведите доказательство ожидаемой эвристической оценки сложности алгоритма ECM по схеме, которая обсуждалась вслед за алгоритмом 7.4.2.

**7.13.** Вспомните метод, относящийся ко второй стадии ECM и затронутый нами в тексте, позволяющий находить соответствия между двумя списками без помощи алгоритма 7.5.1. Его идея состоит в том, чтобы сначала сформировать многочлен

$$f(x) = \prod_{i=0}^{m-1} (x - A_i),$$

а затем вычислить его значения в $n$ точках из $B$, т. е. вычислить его значения при $x = B_j, j = 0, \ldots, n-1$. Суть в том, что если при этом будет найден нуль многочлена $f$, то у нас имеется совпадение (некоторое $B_j$ равно $A_i$). Укажите вычислительную сложность этого метода нахождения $A \cap B$ при помощи многочленов. Как в этой ситуации с многочленами обрабатывать повторяющиеся совпадения? Обратите внимание на связанный с этим упражнением материал разделов 5.5 и 9.6.3.

**7.14.** Проанализировав динамику «рекордных» разложений, полученных с помощью ECM, спрогнозируйте, в каком календарном году для нас станет возможным открытие 70-значных делителей посредством ECM. (По прогнозам Циммермана (см. [Zimmermann 2000]), например, это произойдет в 2010 г.)

**7.15.** Убедитесь в справедливости утверждений, высказанных относительно алгоритма 7.5.10, рассуждая следующим образом. Во-первых, покажите, как были получены представленные в таблице параметры $r, s$. Для этого используйте тот факт, что многочлен классов имеет степень не выше второй, и заметьте, что в задающем кривую уравнении $y^2 = x^3 + Rx/S + T/S$ можно избавиться от знаменателя $S$, умножив обе части на $S^6$. Во-вторых, используйте квадратичный закон взаимности для доказательства того, что каждый квадратный корень, явно присутствующий в табличных параметрах, в самом деле существует. Для этого предполагайте, что для $p$ найдено представление $4p = u^2 + |D|v^2$. В-третьих, покажите, что $4a^3 + 27b^2$ не может обращаться в нуль $(\mathrm{mod}\ p)$. Это можно сделать, разбирая все случаи один за другим, но проще вернуться к алгоритму 7.5.9 и понять, как на самом деле возникают окончательные параметры $a, b$. Наконец, разложите на множители приведенные в таблице значения $s$ и убедитесь в том, что им свойственна высокая степень гладкости. Чем можно объяснить эту гладкость?

**7.16.** Вспомним, что для эллиптической кривой $E_{a,b}(\mathbf{F}_p)$ искажение $E'$ кривой $E$ задается уравнением

$$y^2 = x^3 + g^2ax + g^3b,$$

где $\left(\frac{a}{p}\right) = -1$. Покажите, что порядки кривых связаны соотношением

$$\#E + \#E' = 2p + 2.$$

**7.17.** Предположим, что наибольший порядок элемента конечной абелевой группы $G$ равен $m$. Покажите, что существует абсолютная константа $c > 0$ (т. е. $c$ не зависит от $m$ и $G$), такая, что доля элементов группы $G$, имеющих порядок $m$, не меньше, чем $c/\ln\ln(3m)$. (Здесь множитель 3 присутствует лишь для того, чтобы обеспечить положительность повторного логарифма.) Этот результат имеет отношение к замечаниям, сопровождающим теорему 7.5.2, а также к некоторым результатам гл. 3.

**7.18.** Рассмотрите при $p = 229$ кривые $E, E'$ над полем $\mathbf{F}_p$, заданные, соответственно, уравнениями

$$y^2 = x^3 - 1,$$

$$y^2 = x^3 - 8,$$

где последняя кривая является искажением первой. Покажите, что $\#E = 252, \#E' = 208$, и соответствующие группы имеют строение

$$E \cong \mathbf{Z}_{42} \times \mathbf{Z}_6,$$

$$E' \cong \mathbf{Z}_{52} \times \mathbf{Z}_4.$$

Таким образом, покажите, что каждая точка $P \in E$ такова, что $[252]P = [210]P = O$, и точно так же каждая точка $P \in E'$ такова, что $[208]P = [260]P = O$, вследствие чего для любой точки на любой из этих кривых не существует единственного числа $m$ в интервале Хассе, для которого $[m]P = O$. См. работу [Schoof 1995], в которой описываются этот и другие частные случаи, относящиеся к теоремам Местре.

**7.19.** Здесь мы исследуем сложность алгоритма Шуфа 7.5.6 в операциях. Выведите границу $O\left(\ln^8 p\right)$ для сложности в операциях первоначального метода Шуфа, предполагая, что используется школьное умножение многочленов (которое, в свою очередь, имеет сложность $O(de)$ операций в поле при степенях операндов $d, e$). Объясните, почему продолжение в виде метода Шуфа—Элкиса—Аткина (SEA) снижает эту границу до $O\left(\ln^6 p\right)$. (Чтобы получить это понижение, необходимо знать лишь степень SEA-многочленов, которая при простом $l$ есть $O(l)$, а не $O(l^2)$.) Опишите, что произойдет затем с этой оценкой сложности, если привлечь еще и метод быстрого умножения, причем не только для целых чисел, но и для многочленов (см. текст, сопровождающий алгоритм 7.5.6), а возможно еще и поднятие Шенкса—Местре. Наконец, что можно сказать о *битовой* сложности отыскания порядка кривой при простом $p$, имеющем $n$ битов?

**7.20.** Теория эллиптических кривых может быть использована для установления определенных результатов о суммах кубов в кольцах. С помощью теоремы Хассе 7.3.1 докажите, что если $p > 7$ — простое число, то каждый элемент поля $\mathbf{F}_p$ является суммой двух кубов. Проанализировав, далее, степени простых

чисел, докажите следующее предположение (которое было выдвинуто на основании численных данных и сообщено Д. Коуплендом): пусть $d_N$ — плотность элементов, представимых (в виде (куб+куб)) в кольце $\mathbf{Z}_N$. Тогда

если $63|N$, то $d_N = 25/63$, иначе
если $7|N$, то $d_N = 5/7$, или
если $9|N$, то $d_N = 5/9$,
и во всех остальных случаях $d_N = 1$.

Одно из продолжений таково: изучите суммы более высоких степеней (см. упр. 9.80).

**7.21.** Здесь приводится один из примеров того, как упражнение на символьные вычисления может способствовать пониманию работы специфического, трудного алгоритма. По существу, иногда можно проделать то, что мы могли бы назвать «символьным алгоритмом Шуфа», и получить точные результаты о порядках кривых, рассуждая по следующей схеме. Рассмотрим эллиптическую кривую $E_{0,b}(\mathbf{F}_p)$ при $p > 3$, так что она задается уравнением

$$y^2 = x^3 + b.$$

Мы определим порядок (mod 3) любой такой кривой, причем сделаем это при помощи одних только символьных манипуляций, т. е. без использования обычных численных операций, присущих реализациям алгоритма Шуфа. Проведите доказательства следующих фактов, *не прибегая* к помощи вычислительной техники (хотя программа для автоматических символьных вычислений может пригодиться для проверки наших алгебраических выкладок):

(1) Покажите, что по модулю полинома деления $\Psi_3$ справедливо сравнение

$$x^4 \equiv -4bx \pmod{\Psi_3}.$$

(2) Докажите, что при $k > 0$

$$x^{3k} \equiv (-4b)^{k-1} x^3 \pmod{\Psi_3}.$$

Эта редукция инициирует цепочку точных результатов для соотношения Фробениуса, в чем мы далее убедимся.

(3) Покажите, что выражению $x^p$ теперь можно придать явный вид

$$x^p \equiv (-4b)^{\lfloor p/3 \rfloor} x^{p \bmod 3} \pmod{\Psi_3},$$

в котором использовано наше обычное обозначение mod, так что $p \bmod 3 = 1$ или 2.

(4) Покажите, что $x^{p^2}$ также может быть записано в явном виде как

$$x^{p^2} \equiv (-4b)^{(p^2-1)/3} x \pmod{\Psi_3},$$

и получите, что при $p \equiv 2 \pmod 3$ данное сравнение сводится к сравнению $x^{p^2} \equiv x$, которое не зависит от $b$.

(5) С помощью биномиального разложения и редуцирующего соотношения из пункта (2) установите следующее общее тождество для натурального

числа $d$ и $\gamma \not\equiv 0 \pmod{p}$:

$$(x^3+\gamma)^d \equiv \gamma^d \left(1 - \frac{x^3}{4b}\left((1-4b/\gamma)^d - 1\right)\right) \pmod{\Psi_3}.$$

(6) Исходя из того, что $y^p \equiv y(x^3+b)^{(p-1)/2}$, выразите степень $y^p$ в виде

$$y^p \equiv y b^{(p-1)/2} q(x) \pmod{\Psi_3},$$

где $q(x) = 1$ или $(1 + x^3/(2b))$ при $p \equiv 1, 2 \pmod 3$, соответственно.

(7) Покажите, далее, что мы всегда имеем

$$y^{p^2} \equiv y \pmod{\Psi_3}.$$

А теперь, имея эти подготовительные утверждения, выведите из теоремы 7.5.5, что при $p \equiv 2 \pmod 3$ мы имеем, независимо от $b$, что

$$\#E \equiv p + 1 \equiv 0 \pmod 3.$$

Наконец, при $p \equiv 1 \pmod 3$ выведите на основе оставшихся возможностей для соотношения Фробениуса

$$(c_1 x, y) + [1](x, y) = t(c_2 x, y c_3),$$

в котором параметры $c_i$ зависят от $b$, что порядок кривой $\pmod 3$ зависит от квадратичного характера вычета $b \pmod p$ следующим образом:

$$\#E \equiv p + 1 + \left(\frac{b}{p}\right) \equiv 2 + \left(\frac{b}{p}\right) \pmod 3.$$

Интересно исследовать такой вопрос: как далеко можно продвинуть этот «символьный алгоритм Шуфа» (см. упр. 7.30)?

**7.22.** Для приведенного в качестве примера простого числа $p = \left(2^{31}+1\right)/3$ и его порядков кривых, выписанных вслед за алгоритмом 7.5.10, укажите, какой из этих порядков лучше всего использовать для получения ECPP-доказательства того, что $p$ — простое?

**7.23.** Используйте какой-либо вариант ECPP для доказательства простоты каждого из десяти последовательных простых чисел в упр. 1.87.

**7.24.** Здесь мы применяем идеи ECPP к проверке простоты чисел Ферма $F_m = 2^{2^m}+1$. Рассматривая представления

$$4F_m = u^2 + 4v^2,$$

докажите, что если $F_m$ — простое, то существуют четыре кривые $\pmod{F_m}$

$$y^2 = x^3 - 3^k x, \quad k = 0, 1, 2, 3,$$

имеющие при некотором упорядочении порядки кривых

$$2^{2^m} + 2^{m/2+1} + 1,$$
$$2^{2^m} - 2^{m/2+1} + 1,$$
$$2^{2^m} - 1,$$
$$2^{2^m} + 3.$$

Докажите с помощью компьютера, что $F_7$ (или некоторое еще большее число Ферма) — составное, предъявив на одной из этих четырех кривых точку $P$, которая *не* аннулируется никаким из этих четырех порядков. Следует, вероятно, использовать представление Монтгомери из алгоритма 7.2.7, так что для начальных точек необходимо иметь только их $x$-координаты, проверенные на допустимость (см. объяснение после алгоритма 7.2.1). Если же не использовать это представление, то упражнение обречено на неудачу, поскольку при составных $F_m$ обычно просто нельзя извлечь квадратные корни для получения $y$-координат.

Конечно, знаменитый тест на простоту Пепена (теорема 4.1.2) гораздо более эффективен в вопросе выявления простых чисел, однако наше понятие СМ-кривых играет здесь поучительную роль. На самом деле, когда указанная выше процедура вызывается для $F_4 = 65537$, обнаруживается, что каждая из этих четырех кривых теперь уже имеет начальную точку, которая аннулируется одним из этих четырех порядков. Так что в этом смысле мы можем рассматривать 65537 как «вероятное» простое число. Лишь небольшая дополнительная работа по принципу ECPP-алгоритма Аткина завершает доказательство простоты этого наибольшего из известных простых чисел Ферма.

## 7.8. Проблемы для исследования

**7.25.** С целью выяснения соотношений между оценками сложности алгоритмов 7.2.2, 7.2.3 и 7.2.7 проанализируйте сложность вычисления обратного элемента в поле. Внимательно смотрим на выражения $x_3 = m^2 - x_1 - x_2$, $y_3 = m(x_1 - x_3) - y_1$ и понимаем, что если бы вычисление обратного «ничего не стоило», то аффинный подход был бы лучше. Однако известные методы вычисления обратных весьма трудоемки. На практике обнаруживается, что время вычисления обратного имеет тенденцию на один или два порядка превышать время умножения по модулю. Де Вин и др. (см. [De Win et al. 1998]) предлагают объяснение того, что даже снизить затраты на вычисление обратного (по модулю типичного для криптографии простого числа $p \approx 2^{200}$) до 20 умножений очень трудно. Но существуют открытые вопросы. Что если рассматривать простые специального вида или использовать поисковые таблицы? Идея поисковых таблиц проистекает из простого факта: если можно найти $y$, такое, что $xy \equiv z \pmod p$ при некотором $z$, обратное к которому уже известно, то $x^{-1} \bmod p = yz^{-1} \bmod p$. В связи с вопросом о сложности вычисления обратного см. алгоритм 9.4.5 и упр. 2.11.

Другое направление исследований состоит в том, чтобы попытаться реализовать интересные методы классов Сёренсона для $k$-ичных (в противоположность двоичным) вариантов НОД (см. [Sorenson 1994]); такие методы допускают расширенный вариант, вычисляющий обратные по модулю.

**7.26.** Для эллиптической кривой $E(\mathbf{F}_p)$, $p$ — простое, заданной уравнением

$$y^2 = x(x+1)(x+c)$$

(и $c \not\equiv 0, 1 \pmod p$), покажите прямым применением соотношения (7.8) для

порядка кривой, что $\#E = p + 1 - T$, где

$$T = \sum_{n=0}^{Q} c^n \binom{Q}{n}^2,$$

причем $Q = (p-1)/2$, и мы понимаем под суммой тот ее вычет по модулю $p$, который лежит в интервале $(-2\sqrt{p}, 2\sqrt{p})$. (Один из подходов состоит в том, чтобы записать символ Лежандра в соотношении (7.8) в виде $(p-1)/2$-й степени и затем формально просуммировать по $x$.) Далее, покажите, что

$$T \equiv F(1/2, 1/2, 1; c)|_Q \pmod{p},$$

где $F$ — стандартная гипергеометрическая функция Гаусса, а обозначение с горизонтальной чертой указывает на то, что мы должны взять лишь отрезок гипергеометрического ряда $F(A, B, C; z)$ до члена с $z^Q$ включительно. Также выведите формальное соотношение

$$T = (1-c)^{Q/2} P_Q\left(\frac{1-c/2}{\sqrt{1-c}}\right),$$

где $P_Q$ — классический полином Лежандра порядка $Q$. Используя известные свойства, связанные с преобразованием таких специальных рядов, найдите некоторые явно выражаемые порядки кривых. Например, взяв $p \equiv 1 \pmod 4$ и известное значение в нуле

$$P_Q(0) = \binom{Q}{Q/2},$$

можно вывести, что порядок нашей кривой равен $\#E = p + 1 \pm 2a$, где простое число $p$ представлено в виде $p = a^2 + b^2$. На самом деле, подобное исследование связано с алгебраической теорией чисел. Например, в контексте этой задачи оказывается полезным изучение биномиальных коэффициентов $\pmod p$ (см. [Crandall et al. 1997]).

Отметим, что значение такого гипергеометрического ряда можно вычислить за $O\left(\sqrt{p}\ln^2 p\right)$ операций в поле, если применить быстрые методы вычисления значений рядов (см. [Borwein and Borwein 1987], а также алгоритм 9.6.7). Это означает, что по крайней мере для эллиптических кривых указанного вида у нас имеется еще один алгоритм подсчета точек; по сложности он располагается, по существу, между простым суммированием квадратичных вычетов и алгоритмом Шенкса—Местре. Существует еще одно возможное направление исследований: понятие DAGM из упр. 2.42 можно, в некотором смысле, применить к отрезкам гипергеометрического ряда $\pmod p$. Мы упомянули об этом в связи с тем, что *классическое* понятие AGM для вещественных аргументов служит мощным средством вычисления значений таких функций, как возникшая выше гипергеометрическая форма (см. [Borwein and Borwein 1987]).

**7.27.** По принципу упражнения 7.26 покажите, что для простого числа $p \equiv 1 \pmod 8$ эллиптическая кривая $E$, заданная уравнением

$$y^2 = x^3 + \frac{3}{\sqrt{2}}x^2 + x,$$

имеет порядок

$$\#E = p + 1 - \left(2^{(p-1)/4} \binom{\frac{p-1}{2}}{\frac{p-1}{4}} \bmod \pm\, p\right),$$

где обозначение $\bmod_{\pm}$ указывает на то, что мы берем вычет со знаком, ближайший к 0. Имеет ли это замечание какое-либо значение для разложения на множители чисел Ферма? Вот некоторые наблюдения. Нам известно, что любой простой делитель составного числа $F_n$ сравним с 1 (mod 8); кроме того, $3/\sqrt{2}$ может быть записано по модулю любого числа Ферма $F_n > 5$ в виде $3(2^{3m/4} - 2^{m/4})^{-1}$, где $m = 2^n$. Более того, эта формула работает по модулю любого простого делителя числа $F_n$. В этой связи см. работу [Atkin and Morain 1993a], в которой авторы показывают, как получать удобные для вычислений кривые в тех случаях, когда известно, что потенциальные делители $p$ удовлетворяют определенным сравнениям.

**7.28.** Реализуйте вариант алгоритма ECM из работы [Peralta and Okamoto 1996], который позволяет эффективно «атаковать» составные числа вида $n = pq^2$, где $p$ — простое, а $q$ — нечетное. Их результат основан на интересном вероятностном способе проверки сравнения $x_1 \equiv x_2 \pmod{p}$. Он состоит в выборе случайного числа $r$ и проверке равенства символов Якоби

$$\left(\frac{x_1 + r}{n}\right) = \left(\frac{x_2 + r}{n}\right);$$

примечательно, что эту проверку можно осуществить при неизвестном $p$.

**7.29.** Здесь мы приводим интересные направления исследований, связанные с методом подсчета точек Шуфа — алгоритмом 7.5.6. Во-первых, изучите соотношения между памятью и временем работы алгоритма в зависимости от того, какой из перечисленных ниже вариантов представления мы выбираем: (a) рациональные представления точек $(N(x)/D(x), yM(x)/C(x))$, которые мы выписывали; (b) проективное описание $(X(x,y), Y(x,y), Z(x,y))$ по принципу алгоритма 7.2.3; и наконец, (c) аффинное представление. Заметим, что эти варианты имеют асимптотику сложности, *одинаковую* по порядку, однако речь здесь идет о преимуществах реализаций, например, о постоянных в «$O$».

Подобные рассмотрения привели к созданию действующих программных пакетов, причем не только для «рафинированного» алгоритма Шуфа 7.5.6, но и для изощренных SEA-вариантов. Некоторые из таких пакетов весьма эффективны и способны находить порядок кривой при 200-битовом значении $p$ в течение нескольких минут. Например, существует реализация Скотта (см. [Scott 1999]), которая использует проективные координаты и метод Шупа (см. упр. 9.70) для умножения многочленов; в SEA-расширении используется предварительное вычисление многочленов.

Но существует другая заманчивая возможность: использовать представление Монтгомери из алгоритма 7.2.7, при котором в соотношении Шуфа

$$\left(x^{p^2}, y^{p^2}\right) + [k](x,y) = [t](x^p, y^p)$$

можно анализировать только $x$-координату. Мы вычисляем $x^{p^2}$ (не вычисляя

степеней $y$), используем полиномы деления для нахождения $x$-координаты точки $[k](x,y)$ (а возможно и $[t]$-кратной точки) и применяем алгоритм 7.2.8 для нахождения двух вариантов значения $t$. Сделав это, мы можем действовать по сценарию «неполной китайской теоремы об остатках», который сам по себе интересен для изучения. Такой сценарий предполагает, что мы знаем не конкретное значение $t \bmod l$ для каждого малого простого числа $l$, а *пару* вариантов значения $t$ для каждого $l$. На первый взгляд может показаться, что нам потребуется вдвое больше малых простых чисел, но на самом деле это не так. Если у нас есть, скажем, $n$ меньших простых чисел $l_1, \ldots, l_n$ , то можно выполнить, самое большее, $2^n$ эллиптических умножений для выяснения истинного порядка кривой, который аннулирует случайную точку. Мы могли бы сказать, что при большом $n$ объем работы слишком велик, однако мы можем использовать арифметику $x$-координат лишь для некоторого большего значения $l$. Итак, проблема для исследования состоит в следующем: зная, что арифметика $x$-координат (Монтгомери) менее трудоемка, чем полные $(x,y)$-варианты, опишите, как наилучшим образом управлять возникающей неоднозначностью при определении $t$? Существует ли кроме $2^n$-продолжения продолжение Шенкса—Местре, которое отталкивается от декомпозиции этой неполной китайской теоремы об остатках? Заметим, что весь этот анализ должен учитывать, что иногда возможна благоприятная ситуация $t = 0$; в этом случае вычет $(p + 1 \pm t) \bmod l$ определяется однозначно.

**7.30.** В упр. 7.21 были намечены контуры «символьного» способа выполнения вычислений Шуфа для нахождения порядка эллиптической кривой. Исследуйте возможность проведения аналогичных рассуждений по (mod 3) для кривых, задаваемых уравнением

$$y^2 = x^3 + ax,$$

и раз уж мы заговорили об этом, для кривых, у которых *каждый* из параметров $a, b$ отличен от нуля — такие случаи, вероятно, будут трудными. Исследуйте возможность применения любой из возникших идей в случае малых простых чисел $l > 3$.

**7.31.** Опишите, как можно использовать алгоритм 7.5.10 для создания сравнительно простой программы доказательства простоты, в которой производился бы поиск кривых только для тех дискриминантов $D$, для которых $h(D) = 1, 2$. Достоинство такой схемы очевидно: генерирование эллиптических кривых для таких дискриминантов происходит практически мгновенно. Основным недостатком, конечно, является то, что при исследовании больших вероятных простых чисел огромных затрат потребует разложение на множители порядков кривых из жестко ограниченного множества (можно даже предусмотреть в программе процедуру разложения с помощью ECM, чтобы сделать больший упор на ту часть ECPP, которая связана с разложением). Тем не менее, это неплохой подход для простых чисел, имеющих до нескольких сотен двоичных знаков. Сразу получаем, что тогда не потребуется ни вычислений с плавающей точкой для многочлена классов, ни большого объема памяти для хранения многочленов, ни изощренных процедур для нахождения корней.

**7.32.** Существует способ несколько упростить вычисления с эллиптическими кривыми в алгоритме ECPP. Покажите, что параметризация Монтгомери (см. алгоритм 7.2.7) вполне может быть использована для доказательства простоты некоторого заданного числа $n$ в ECPP-алгоритмах 7.6.2 и 7.5.9 *при условии*, что наряду с требованием необращения в нуль кратных $(X', Z') = [m/q](X, Z)$ мы каждый раз проверяем НОД$(Z', n)$ на предмет возможных делителей $n$.

Опишите, далее, некоторые улучшения в работе ECPP-алгоритмов, которые наблюдаются при действии параметризации Монтгомери. Например, проще найти точку на кривой, поскольку нам нужна лишь допустимая $x$-координата, и т. д.

**7.33.** Здесь мы приводим своеобразную форму «быстрого ECPP», которая при благоприятных условиях позволяет получать практически мгновенные доказательства простоты. Вспомним (см. следствие 4.1.4), что если вероятное простое число $n$ таково, что $n - 1 = FR$, где разложенная на множители часть $F$ превосходит $\sqrt{n}$ (а в разнообразных улучшениях превосходит даже еще меньшую границу), то можно быстро получить доказательство простоты. Рассмотрим теперь ситуацию, когда получено такое же «FR-разложение», и кроме этого нам удалось записать

$$R = \alpha F + \beta,$$

причем для фундаментального дискриминанта $-|D|$ существует представление $4\alpha = \beta^2 + |D|\gamma^2$. Покажите, что если при данных условиях число $n$ — простое, то существует CM-кривая $E$ дискриминанта $-|D|$, для которой порядок задается удобным соотношением

$$\#E = \alpha F^2.$$

Таким образом, $F$ может обеспечивать ECPP-результат о числе $n$ и быть примерно таким же малым, как $n^{1/4}$.

Далее, покажите, что вероятное простое число Макинтоша—Вагштафа, которое имеет вид $n = (2^q + 1)/3$, всегда обладает представлением с дискриминантом $D = -8$, и выпишите соответствующий порядок кривой. При помощи этих идей докажите, что число $(2^{313} + 1)/3$ — простое, если принять во внимание тот факт, что соответствующий порядок кривой $\#E = (2/3)h^2$, где $h$ равно

$$3^2 \cdot 5 \cdot 7 \cdot 13^2 \cdot 53 \cdot 79 \cdot 157 \cdot 313 \cdot 1259 \cdot 1613 \cdot 2731 \cdot 3121 \cdot 8191 \cdot 21841 \cdot 121369 \cdot 22366891.$$

Затем докажите другое интересное следствие: если число

$$n = 2^{2r+2m} + 2^{r+m+1} + 2^{2r} + 1$$

является простым, то соответствующая кривая $E$ такова, что

$$\#E = 2^{2r}(2^{2m} + 1).$$

Пользуясь этим и проанализировав известные алгебраические сомножители числа $2^{2m} + 1$ при нечетном $m$, докажите, что число

$$n = 2^{576} + 2^{289} + 2^2 + 1$$

является простым.

Дополнительную информацию о «быстрых» доказательствах простоты см. в работе [Pomerance 1987a]; см. также обсуждение в книге [Williams 1998, p. 366], касающееся чисел определенного тернарного вида.

**7.34.** После того как с использованием схемы ECM, такой, как в алгоритме 7.4.4, был обнаружен делитель, можно обратиться к следующей интересной задаче: каков был фактический порядок группы, приведший к открытию этого делителя?

Один из подходов, который использовался Брентом и др. (см. [Brent et al. 2000]), состоит в том, чтобы просто «отступать» от пределов стадий, пока не будут найдены точные значения наибольшего и второго по величине простого делителя и так далее, до тех пор, пока порядок группы не будет полностью разложен.

Ну а другой подход состоит в том, чтобы просто вычислить, скажем, посредством алгоритма 7.5.6, этот фактический порядок. С этой целью разработайте подготовительную алгебру кривых следующим образом. Во-первых, покажите, что если кривая выбрана в соответствии с теоремой 7.4.3, то рациональная начальная точка $x/z = u^3/v^3$ удовлетворяет в нашем кольце уравнению

$$x^3 + Cx^2z + xz^2 = \left(\sigma^2 - 5\right)^3 \left(125 - 105\sigma^2 - 21\sigma^4 + \sigma^6\right)^2.$$

Затем сделайте вывод, что порядок выбранной кривой есть либо порядок кривой

$$y^2 = x^3 + ax + b,$$

либо порядок искажения последней, в зависимости от того, какое равенство выполнено: $(\frac{\sigma^3-5\sigma}{p}) = 1$ или $-1$, причем аффинные параметры $a, b$ получаются из соотношений

$$\gamma = \frac{(v-u)^3(3u+v)}{4u^3v} - 2,$$

$$a = 1 - \frac{1}{3}\gamma^2,$$

$$b = \frac{2}{27}\gamma^2 - \frac{1}{3}\gamma.$$

На основании этих манипуляций можно предложить прямой алгоритм нахождения порядка кривой, которая привела к открытию делителя $p$. А именно, пользуясь известным инициализатором $\sigma$, вновь вычисляем, если это необходимо, параметры поля $u, v$, затем применяем приведенные выше формулы для нахождения пары $(a, b)$ параметров аффинной кривой; эту пару, в свою очередь, можно напрямую использовать в алгоритме Шуфа.

Приведем конкретный пример работы этого метода. Делитель Макинтоша—Тардифа

$$p = 81274690703860512587777$$

числа Ферма $F_{18}$ был найден с помощью параметра инициализации $\sigma = 16500076$. С помощью приведенных выше формул находим, что

$$a = 26882295688729303004012,$$

$$b = 10541033639146374421403,$$

и алгоритм 7.5.6 определяет порядок кривой как

$$\begin{aligned} \#E &= 81274690703989163570820 \\ &= 2^2 \cdot 3 \cdot 5 \cdot 23 \cdot 43 \cdot 67 \cdot 149 \cdot 2011 \cdot 2341 \cdot 3571 \cdot 8161. \end{aligned}$$

На самом деле, если взглянуть здесь на два наибольших простых делителя, мы видим, что указанный делитель можно было бы обнаружить с помощью таких небольших пределов стадий, как $B_1 = 4000$, $B_2 = 10000$, соответственно. Р. Макинтош и К. Тардиф в действительности использовали 100000, 4000000, соответственно, но, как всегда бывает при работе с ECM, то, что можно назвать «пост-факторной» ретроспективной информацией, не представляет особой ценности. Укажем еще на практическое подтверждение одного факта — как и следовало ожидать, метод параметризации Брента дает кривую, порядок которой делится на двенадцать.

Если Вы обладаете программным обеспечением для работы с достаточно большим количеством разрядов, Вам предлагается еще один полезный тест, использующий вышеупомянутые идеи. Возьмите известный простой делитель $p = 4485296422913$ числа $F_{21}$ и для соответствующего параметра Брента $\sigma = 1536151048$ найдите порядок группы эллиптической кривой $(\bmod p)$, затем покажите, что пределы стадий $B_1 = 60000$, $B_2 = 3000000$ (реальная пара, первоначально использованная на практике и служащая отправной точкой этого примера на получение ретроспективной информации) достаточны для обнаружения делителя $p$.

**7.35.** (С. Н. Преображенский[9].) Исследуйте приводимый ниже подход к разложению целых чисел на множители, использующий эллиптические кривые и связанные с ними модулярные формы.

Известно, что некоторые модулярные формы имеют целые коэффициенты Фурье $c_n$, и эти коэффициенты удовлетворяют следующим соотношениям:

$$\begin{aligned} &c_{p^r} c_p = c_{p^{r+1}} + p c_{p^{r-1}}, && p \text{ — простое, } p \nmid N; \\ &c_{p^r} = (c_p)^r, && p \text{ — простое, } p \mid N; \\ &c_m c_n = c_{mn}, && \text{НОД}\,(m, n) = 1. \end{aligned}$$

Здесь $c_1 = 1$, $r \geqslant 1$ — целое число, $N$ — так называемый уровень (или ступень) модулярной формы. Кроме того, знаменитая гипотеза Шимуры—Таниямы (доказанная Уайлсом для случая полуустойчивых эллиптических кривых) утверждает, что при определенных условиях числа $a_p$, через которые определяются порядки над полями $\mathbb{Z}_p$ фиксированной эллиптической кривой $E$ над $\mathbb{Q}$, равны коэффициентам Фурье $c_p$ некоторой модулярной формы указанного типа. Числа $a_p$ определяются так:

$$\#E(\mathbb{Z}_p) = p + 1 - a_p.$$

[9] Добавлено при переводе. См. *Преображенский С. Н.* Восстановление коэффициентов Фурье некоторых функций и разложение целых чисел на множители //Вестн. Моск. ун-та. Сер. 1, Математика. Механика. В печати. — *Прим. ред.*

Эти свойства можно использовать для разложения на множители числа $n = pq$. Будем считать, что $p$ и $q$ — нечетные простые примерно одного порядка (такие $n$ называют RSA-числами, см. п. 8.1.2). Выберем произвольную эллиптическую кривую $E$ над $\mathbb{Q}$. Если бы для соответствующей модулярной формы мы могли найти коэффициенты $c_n$ и $c_{n^2}$, разложить их на множители и тем самым найти, скажем, $c_p$ и $c_{p^2}$, то тогда из первого равенства мы нашли бы

$$p = (c_p)^2 - c_{p^2}.$$

Ключевой момент состоит в том, чтобы найти $c_n$ и $c_{n^2}$, когда разложение $n = pq$ неизвестно. (Если бы это разложение было известно, то мы нашли бы $c_n$ и $c_{n^2}$, вычислив $c_p = a_p$ и $c_q = a_q$ с помощью алгоритма Шуфа 7.5.6 для кривой $E$.) Таким образом, мы хотим свести задачу о разложении числа $n$ к задаче о разложении чисел $c_n$ и $c_{n^2}$, зависящих от случайно выбранной кривой $E$.

Можно попытаться найти $c_n$ и $c_{n^2}$, когда разложение $n$ на множители неизвестно, используя метод, напоминающий интерполяцию — мы используем интегральное представление для коэффициентов Фурье модулярной формы и метод Шуфа для вычисления определенного количества коэффициентов $c_{p'}$ и $c_{p''}$ для простых чисел $\{p'\}$ и $\{p''\}$, близких к $n$ и $n^2$ соответственно. Значения коэффициентов в «узлах», которыми служат множества $\{p'\}$ и $\{p''\}$, позволяют «интерполировать» значения коэффициентов в «точках» $n$ и $n^2$. Чтобы понять, как можно вычислить, например, $c_n$, зная определенное количество $c_{p'}$, рассмотрим следующий «наивный» метод. Пусть

$$n < p'_1 = n + \Delta_1 < p'_2 = n + \Delta_2 < \ldots < p'_K = n + \Delta_K.$$

Если бы можно было записать приближенные равенства

$$\int\limits_{-\frac{1}{2}}^{\frac{1}{2}} f(\tau) e^{-2\pi i n \tau} e^{-2\pi i \Delta_1 \tau}\, d\rho \approx$$

$$\approx \int\limits_{-\frac{1}{2}}^{\frac{1}{2}} f(\tau) e^{-2\pi i n \tau} \left(1 + (-2\pi i \Delta_1 \tau) + \ldots + \frac{(-2\pi i \Delta_1 \tau)^{K-1}}{(K-1)!}\right) d\rho =$$

$$= 1 \cdot x_1 + \Delta_1 \cdot x_2 + \ldots + {\Delta_1}^{K-1} \cdot x_K \approx c_{p'_1},$$

$$\ldots,$$

$$1 \cdot x_1 + \Delta_K \cdot x_2 + \ldots + {\Delta_K}^{K-1} \cdot x_K \approx c_{p'_K},$$

где

$$\tau = \rho + i\sigma,\ \sigma > 0,$$

то, решив эту систему линейных уравнений, мы нашли бы $c_n \approx x_1$. Однако эти приближенные равенства, вообще говоря, не имеют места. Чтобы все остаточные члены соответствующих рядов Тейлора для экспонент были малы,

необходимо взять большее количество членов этих рядов, а это увеличит число неизвестных.

Другая мысль состоит в том, чтобы раскладывать не экспоненты $e^{-2\pi i\Delta_k\tau}$ в ряд Тейлора по степеням $\tau$, а раскладывать саму модулярную форму $f(\tau)$ в ряд по многочленам Лежандра, а экспоненты оставлять без изменения. Оказывается, что если форма $f(\tau)$ достаточно хорошо приближается частичными суммами своего разложения по многочленам Лежандра, то с помощью метода, аналогичного описанному выше, можно приближенно найти необходимое количество неизвестных коэффициентов Лежандра и с их помощью приближенно вычислить $c_n$. Однако вопрос о существовании модулярных форм, «достаточно хорошо» приближающихся частичными суммами своих рядов Лежандра, и об их нахождении остается открытым. Открытым остается и вопрос о том, существует ли более быстрый способ вычисления коэффициентов Фурье $c_n$, чем тот, что использует разложение на множители. См. также добавленные при переводе работы [Kilford 2008] (p. 189), [Bach and Charles 2005], [Charles 2005] и [Edixhoven et al. 2006].

Глава 8

# ЭТИ ВЕЗДЕСУЩИЕ ПРОСТЫЕ ЧИСЛА

Многие отмечают, что простые числа в итоге нашли достойное практическое применение в области криптографии. Их криптографическая ценность не вызывает сомнений, однако существуют и многие другие приложения этих великолепных простых чисел. Некоторые из приложений носят сугубо практический характер — например, приложения в численном анализе, прикладной математике и других прикладных науках — в то время как другие относятся к сфере «связей между понятиями», в которой простые числа и связанные с ними понятия используются в теоретической работе вне, скажем так, чистой теории чисел. Эта продуктивная сфера исследований использует простые числа в таких алгоритмах, которые *a priori* могли бы выступать независимо от простых чисел и т. п. По-видимому, справедливо считать, что понятие простого числа по своей природе является вездесущим, поскольку простые числа возникают в столь многочисленных и далеких друг от друга областях знания.

## 8.1. Криптография

Можно сказать, что простые числа применяются в криптографии по причине чрезвычайной сложности определенных вычислений. Двумя такими трудными задачами являются разложение на множители и дискретное логарифмирование. Мы обсудим примеры практического применения этих задач в области криптографии, а также их обобщения на случай эллиптических кривых.

### 8.1.1. Обмен ключом по Диффи—Хеллману

В основополагающей статье [Diffie and Hellman 1976] ее авторы обращают внимание на то, что некоторые групповые операции ведут себя как «односторонние функции» в следующем смысле. Для заданного целого числа $x \geqslant 0$ и элемента $g$ из мультипликативной группы $\mathbf{F}_p^*$ вычисление элемента поля

$$h = g^x$$

(включающее последовательную редукцию по модулю $p$) в общем случае имеет сложность $O(\ln x)$ операций в поле. С другой стороны, решение указанного уравнения относительно $x$ при заданных $g, h, p$, очевидно, является гораздо более трудной задачей. Поскольку $x$ — показатель, и прослеживается явная аналогия с логарифмированием, проблема отыскания значения $x$ называется задачей дискретного логарифмирования (DL). Несмотря на то, что прямая операция (возведение в степень) имеет полиномиальную сложность, до сих пор не

известно ни одного метода, который решал бы задачу дискретного логарифмирования примерно за то же время. Некоторые DL-алгоритмы рассмотрены в гл. 5 и в работе [Schirokauer et al. 1996].

Непосредственным приложением этой «односторонней» природы возведения в степень служит следующий криптографический алгоритм, настолько простой, что мы приводим его здесь на естественном языке, не пользуясь формальным описанием. Предположим, мы желаем, чтобы каждый пользователь имел свой пароль доступа к вычислительной системе или информационному каналу. Выберем общие для всех пользователей простое число $p$ и первообразный корень $g$. Затем каждый участник выбирает свой тайный пароль $x$ (целое число), вычисляет $h = g^x \bmod p$ и, наконец, сохраняет полученное значение $h$ в самой системе. Таким образом, для всего множества пользователей в системе хранится массив чисел $h$. Теперь для получения доступа к системе пользователь должен при входе в нее ввести пароль $x$, система вычисляет значение $g^x \bmod p$ и сравнивает результат с сохраненным числом $h$, отвечающим данному посетителю. Эта система очень проста и основана на том, что по известному значению $h$ трудно определить, каков был пароль $x$ для этого $h$.

Не столь очевидной, но столь же элегантной является схема обмена ключом по Диффи—Хеллману, которая позволяет двум пользователям создать общий ключ шифрования. Хотя мы описываем обмен ключом по Диффи—Хеллману в контексте группы $\mathbf{F}_p^*$, можно (и так часто делают) использовать другие циклические группы.

**Алгоритм 8.1.1** (обмен ключом по Диффи—Хеллману)**.** **Алиса и Боб согласованно выбирают простое число $p$ и порождающий элемент $g \in \mathbf{F}_p^*$. Этот алгоритм позволяет найти общий ключ (mod $p$), такой, что ни один из них не сможет (при условии сложности дискретного логарифмирования) определить индивидуальный ключ другого.**

1. [Алиса создает открытый ключ]
   Алиса выбирает случайное число $a \in [2, p-2]$; // Секретный ключ Алисы.
   $x = g^a \bmod p$; // $x$ — открытый ключ Алисы.
2. [Боб создает открытый ключ]
   Боб выбирает случайное число $b \in [2, p-2]$; // Секретный ключ Боба.
   $y = g^b \bmod p$; // $y$ — открытый ключ Боба.
3. [Вычисление общего ключа]
   Боб вычисляет $k = x^b \bmod p$;
   Алиса вычисляет $k = y^a \bmod p$; // Полученные значения $k$ совпадают.

Разумеется, создание общего ключа основано на равенствах

$$(g^a)^b = (g^b)^a = g^{ab},$$

кроме того, все это проделывается в сочетании с обычными редукциями (mod $p$). Отметим ряд важных особенностей этой базовой концепции обмена ключом по Диффи—Хеллману. Во-первых, Алиса и Боб в принципе могли бы обойтись без выбора случайных чисел, например, выбрав вместо них запоминающиеся фразы, девизы и т. д. и сформировать из них секретные значения $a, b$. Во-

вторых, заметим, что открытые ключи $g^a, g^b \bmod p$ могут быть открыты в том смысле, что (при условии сложности дискретного логарифмирования) можно без опасений сделать их общедоступными. В-третьих, коснемся вопроса о том, что делать с общим ключом, который вырабатывается в этом алгоритме; современные приложения зачастую связаны с использованием общего ключа для шифрования или дешифровки длинных сообщений, скажем, посредством стандартного блочного шифрования, такого, как DES [Schneier 1996]. Несмотря на то, что систему Диффи—Хеллмана можно легко взломать, имея быстрый DL-метод, остается неясным, являются ли эти две задачи равносильными, т. е. если бы предсказатель мог по заданным значениям $g^a$ и $g^b$ выдавать $g^{ab}$, то можно ли было воспользоваться этим предсказателем для быстрого вычисления дискретных логарифмов?

## 8.1.2. Криптосистема RSA

Вскоре после того как стали известны идеи Диффи—Хеллмана, Райвест, Шамир и Адлеман изобрели криптосистему RSA[1] [Rivest et al. 1978], получившую в наши дни широкое распространение.

**Алгоритм 8.1.2** (генерация закрытого/открытого ключа RSA). **Этот алгоритм вырабатывает закрытый ключ и связанный с ним открытый ключ для криптосистемы RSA.**

1. [Выбор простых чисел]
   Выбираем два различных простых числа $p, q$ в соответствии с общепризнанными нормами безопасности (см. текст);
2. [Генерация открытого ключа]
   $N = pq$;
   $\varphi = (p-1)(q-1)$; // Функция Эйлера от числа $N$.
   Выбираем случайное число $E \in [3, N-2]$, взаимно простое с $\varphi$;
   Пара $(N, E)$ образует открытый ключ; // Этот ключ доступен всем.
3. [Генерация секретного ключа]
   $D = E^{-1} \bmod \varphi$;
   Секретный ключ равен $D$; // Ключ $D$ знает только сам пользователь.

Важно отметить, что благодаря сложности разложения на множители числа $N = pq$, доступное всем число $N$ не позволяет с легкостью найти секретные простые числа $p, q$. Более того, известно, что знание целых чисел $D, E$ из отрезка $[1, n-1]$, таких, что $DE \equiv 1 \pmod{\varphi}$, позволяет разложить число $N$ на множители с (вероятностной) полиномиальной сложностью (см. [Long 1981] и ср. упр. 5.27). В рассмотренном алгоритме принято выбирать приблизительно равные по порядку закрытые простые числа $p, q$, однако некоторые криптографы предлагают дополнительные тесты на безопасность. Существует множество публикаций по вопросу возможных недостатков при выборе тех или

[1] Криптосистема RSA названа по первым буквам фамилий ее создателей: Rivest, Shamir, Adleman. — *Прим. перев.*

иных значений $p, q$. В частности, короткий, но достаточно поучительный список возможных слабых мест в защите, которые зависят от величины, а также некоторых теоретико-числовых свойств чисел $p$, $q$, приведен в работе [Williams 1998, p. 391]. Источник [Bressoud and Wagon 2000, p. 249] также содержит список ловушек, связанных с RSA (по различным вопросам безопасности системы RSA см. также упр. 8.2).

Поняв, что открытый ключ представляет собой, в сущности, трудное для взлома (т. е. трудноразложимое) составное число $N = pq$, можно непосредственно перейти к шифрованию сообщений следующим образом.

**Алгоритм 8.1.3** (шифрование/дешифрование в RSA). Предположим, что у Алисы имеются секретный ключ $D_{\mathrm{A}}$ и открытый ключ $(N_{\mathrm{A}}, E_{\mathrm{A}})$, выработанные алгоритмом 8.1.2. При помощи этого алгоритма другой пользователь (Боб) может зашифровать сообщение $x$ (которое мы будем считать целым числом из промежутка $[0, N_{\mathrm{A}})$) и переслать Алисе, которая сможет это сообщение расшифровать.

1. [Боб шифрует]
   $y = x^{E_{\mathrm{A}}} \bmod N_{\mathrm{A}}$; // Боб использует открытый ключ Алисы.
   Боб отсылает число $y$ Алисе;
2. [Алиса дешифрует]
   Алиса получает зашифрованное сообщение $y$;
   $x = y^{D_{\mathrm{A}}} \bmod N_{\mathrm{A}}$; // Алиса восстанавливает исходное число $x$.

Нетрудно видеть, что алгоритм 8.1.3 основан на сравнении

$$x^{DE} \equiv x \pmod{N},$$

вытекающем из равенства $DE = 1 + k\varphi$, которое выполняется по построению числа $D$, так что $x^{DE} = x(x^{\varphi})^k \equiv x \cdot 1^k = x \pmod{N}$ при НОД$(x, N) = 1$. Кроме того, легко заметить, что сравнение $x^{DE} \equiv x \pmod{N}$ остается в силе даже тогда, когда НОД$(x, N) > 1$.

Теперь мы воспользуемся криптосистемой RSA и рассмотрим следующую ситуацию. Пусть каждый из огромного множества пользователей опубликовал свой открытый ключ $(N_i, E_i)$ (при этом получилось нечто, напоминающее телефонную книгу). Тогда кто угодно может отослать зашифрованное сообщение пользователю $j$, воспользовавшись доступным ключом $(N_j, E_j)$ и проведя немного вычислений. Но можно ли сделать так, чтобы получатель $j$ знал, от кого пришло это зашифрованное сообщение? Оказывается, это вполне можно выяснить при помощи хитроумного метода цифровой подписи.

**Алгоритм 8.1.4** (подпись в RSA: простой вариант). Предположим, что у Алисы имеются личный ключ $D_{\mathrm{A}}$ и открытый ключ $(N_{\mathrm{A}}, E_{\mathrm{A}})$, выработанные алгоритмом 8.1.2. Этот алгоритм позволяет другому пользователю (Бобу), обладающему личным ключом $D_{\mathrm{B}}$ и открытым ключом $(N_{\mathrm{B}}, E_{\mathrm{B}})$, «подписать» сообщение $x$ (которое мы будем считать целым числом из промежутка $[0, \min\{N_{\mathrm{A}}, N_{\mathrm{B}}\})$).

1. [Боб шифрует и подписывает]
   $s = x^{D_{\mathrm{B}}} \bmod N_{\mathrm{B}}$; // Боб создает подпись из сообщения.
   $y = s^{E_{\mathrm{A}}} \bmod N_{\mathrm{A}}$; // Здесь Боб пользуется открытым ключом Алисы.
   Боб отсылает число $y$ Алисе;

2. [Алиса дешифрует]
   Алиса получает зашифрованное и подписанное сообщение $y$;
   $s = y^{D_A} \bmod N_A$; // Алиса использует свой закрытый ключ.
   $x = s^{E_B} \bmod N_B$; // Алиса восстанавливает сообщение
   // при помощи открытого ключа Боба.

Отметим, что на завершающем этапе Алиса использует открытый ключ Боба; идея такого использования в том, что (при обычных оговорках о сложности и надежности системы) лишь Боб мог создать исходное сообщение, поскольку его личный ключ $D_B$ известен только ему. Но этот кажущийся элегантным метод цифровой подписи имеет свои недостатки, один из которых таков: если фальсификатор каким-то образом подготовит разложенное на множители сообщение $x = x_1x_2$ и заставит Боба отослать Алисе подписи $y_1, y_2$, соответствующие множителям $x_1, x_2$, то затем злоумышленник сможет выдать себя за Боба, отослав Алисе подпись $y = y_1y_2 \bmod n$ составного сообщения $x$. В этом смысле алгоритм 8.1.4 обладает слишком большой симметрией. Подобные недостатки можно с легкостью устранить на этапе подписания при помощи «дайджеста сообщения», или хэш-функции [Schneier 1996], [Menezes et al. 1997]. Такие стандарты как SHA-1 предоставляют эту хэш-функцию $H$, так что если $x$ — открытый (незашифрованный) текст, то $H(x)$ — целое число (обычно гораздо меньшее, т. е. имеющее намного меньше битов, чем $x$). Этот подход позволяет противостоять некоторым методам раскрытия подписи (или ложной подписи). Рассмотрим метод шифрования с подписью, включающий вычисление хэш-функции.

**Алгоритм 8.1.5** (шифрование с подписью в RSA: более практический вариант). Предположим, что у Боба имеются личный ключ $D_B$ и открытый ключ $(N_B, E_B)$, выработанные алгоритмом 8.1.2. Этот алгоритм позволяет Алисе восстановить исходное сообщение Боба $x$ (которое мы будем считать целым числом из некоторого подходящего промежутка), а также проверить подпись Боба. Будем считать доступной хэш-функцию $H$, соответствующую стандарту SHA-1.

1. [Боб шифрует и подписывает]
   $y = x^{E_A} \bmod N_A$; // Боб шифрует, используя открытый ключ Алисы.
   $y_1 = H(x)$; // $y_1$ равно значению хэш-функции от $x$.
   $s = y_1^{D_B} \bmod N_B$; // Боб создает подпись $s$.
   Боб отсылает составное сообщение $(y, s)$ Алисе;
2. [Алиса дешифрует]
   Алиса получает пару $(y, s)$;
   $x = y^{D_A} \bmod N_A$; // Алиса расшифровывает исходное сообщение $x$.
3. [Алиса обрабатывает подпись]
   $y_2 = s^{E_B} \bmod N_B$;
   if($y_2 == H(x)$) Алиса принимает подпись;
   else Алиса отклоняет подпись;

Отметим, что существует ряд действующих вариантов этого алгоритма, которые не содержат непосредственного шифрования. Например, если безопасность сообщения не важна, а важна лишь подлинность его автора, то можно

просто объединить открытый текст и подпись в пару $(x, s)$ для передачи Алисе. Кроме того, разработаны альтернативные и вместе с тем практические схемы подписи, которые основаны на другой функции — так называемой функции избыточности (см. например, работу [Menezes et al. 1997]).

### 8.1.3. Криптосистемы на эллиптических кривых (криптосистемы ECC)

Середина 80-х годов XX столетия ознаменовалась появлением еще одной удивительной идеи в криптографии — идеи использования эллиптических кривых в криптографических системах [Miller 1987], [Koblitz 1987]. По существу, этот подход связан с выбором открытой кривой $E_{a,b}(F)$, где $F$ — конечное поле. Обычно рассматривают поле $F = \mathbf{F}_p$, где $p$ — простое число, или $F = \mathbf{F}_{2^k}$, где $k$ — специально подобранное целое число. Мы сосредоточимся преимущественно на полях $\mathbf{F}_p$, хотя значительная часть из рассмотренного ниже применима и к конечным полям общего вида. Основное соображение заключается в том, что если точки $P, Q \in E$ удовлетворяют соотношению

$$Q = [k]P$$

для некоторого целого числа $k$, то, вообще говоря, довольно трудно найти дискретный логарифм на эллиптической кривой (EDL), т. е. найти значение множителя $k$. В наши дни доступна масса литературы по вопросам дискретного логарифмирования на эллиптических кривых. В частности, в работе [Lim and Lee 1997] описана роль теоретико-числовых свойств порядка группы (простота, особенности разложения на множители) для обеспечения надежности системы.

В протоколе обмена ключом по Диффи—Хеллману (см. алгоритм 8.1.1) может быть использована циклическая подгруппа любой группы. Следующий алгоритм является модификацией алгоритма Диффи—Хеллмана для групп эллиптических кривых.

**Алгоритм 8.1.6** (обмен ключом в ECC). **Два пользователя, Алиса и Боб, договорились об открытой эллиптической кривой $E$ и об открытой точке $P \in E$, порядок которой равен $n$. (В большинстве сценариев число $n$ простое или почти простое.) Этот алгоритм вырабатывает общий ключ.**

1. [Алиса генерирует открытый ключ]
   Алиса выбирает случайное $K_{\mathrm{A}} \in [2, n-2]$; // Секретный ключ Алисы.
   $Q = [K_{\mathrm{A}}]P$; // Точка $Q$ — открытый ключ Алисы.
2. [Боб генерирует открытый ключ]
   Боб выбирает случайное $K_{\mathrm{B}} \in [2, n-2]$; // Секретный ключ Боба.
   $R = [K_{\mathrm{B}}]P$; // Точка $R$ — открытый ключ Боба.
3. [Каждый из пользователей создает однозначно определенный общий ключ]
   Боб вычисляет точку $K = [K_{\mathrm{B}}]Q$;
   Алиса вычисляет точку $K = [K_{\mathrm{A}}]R$. // Результаты совпадают.

То, что общий ключ определен однозначно, является прямым следствием

группового закона, так как

$$[K_B]([K_A]P) = [K_B K_A]P = [K_A K_B]P = [K_A]([K_B]P).$$

Вновь предполагается, что задача раскрытия, скажем, Бобом закрытого ключа Алисы $K_A$ так же трудна, как задача EDL. Т. е. если задачу EDL решить легко, то обмен ключом в ECC небезопасен; предполагается, что обратное также верно. Заметим, что в реализациях ECC закрытые ключи — целые числа, обычно имеющие порядок $p$ (они могут быть и больше $p$ — вспомним, что сам порядок группы $\#E$ может немного превосходить $p$), в то время как открытые ключи и общий ключ, которым обмениваются, суть *точки*. В типичной ситуации некоторые биты общего ключа могут использоваться, скажем, в блочном шифре. Например, для этой цели можно взять биты $x$-координаты.

Основным результатом о задаче EDL является так называемая «MOV-теорема», которая, по-существу, утверждает, что задача EDL над полем $\mathbf{F}_p$ эквивалентна обычной задаче DL в мультипликативной группе $\mathbf{F}^*_{p^B}$ при некотором значении $B$ (см. [Menezes et al. 1993]). Существует тест, позволяющий на практике оценивать это значение, выражающее степень защищенности в ECC-системе — назовем эту степень MOV-порогом (см. [Solinas 1998]). Используемый на практике MOV-порог $B$ «лежит в окрестности 10», но его примерная величина зависит, конечно, от имеющейся оценки сложности для задачи DL в конечных полях. Добавим, однако, что «суперсингулярные» кривые, имеющие порядок $\#E = p + 1$, особенно уязвимы — известно, что задача EDL для них не труднее, чем задача DL в поле $\mathbf{F}_{p^k}$ при некотором $k \leqslant 6$ (см. [Menezes et al. 1993]). Такие кривые можно исключить *a priori* в силу вышеназванных причин.

Кроме того, в случае когда порядок $\#E = p$, существует так называемая атака Семаева—Смарта—Сато—Араки, основанная на $p$-адической арифметике. (Ее анонсирование Смартом (см. [Smart 1999]) в 1998 г. вызвало бурный резонанс в криптографической среде, хотя ее теория была известна и до этого сообщения (см. [Semaev 1998], [Satoh and Araki 1998]).) Имеются многочисленные ссылки (см., например, сайт Мюллера [Müller 2004]) на более современные атаки; некоторые из них используют информацию о работе схем эллиптического умножения в режиме реального времени.

Между прочим, вопрос о том, как находить эллиптические кривые простого порядка (состоящие, следовательно, из элементов простого порядка) интересен сам по себе. Один из подходов заключается в том, чтобы просто генерировать случайные кривые и устанавливать их порядки посредством алгоритма 7.5.6. Другой состоит в том, чтобы использовать алгоритм 7.5.9 или 7.5.10 для генерации возможных порядков; когда же будет найден простой порядок — двигаться дальше и указать кривую, имеющую этот порядок. Но существуют и более тонкие варианты этих базовых подходов (см. упр. 8.27). Следует отметить, что некоторые криптографы допускают использование кривых порядка $\#E = fr$, где $f$ может состоять из малых простых делителей, а $r$ — большое простое число. Для таких кривых все равно предпочитают находить точки простого порядка $r$, а это можно сделать очень просто.

**Алгоритм 8.1.7** (нахождение точки простого порядка). При задании эллиптической кривой $E_{a,b}(\mathbf{F}_p)$ порядка $\#E = fr$, где $r$ — простое число, этот алгоритм предпринимает попытку нахождения точки $P \in E$ порядка $r$.

1. [Найти начальную точку]
   Выбрать случайную точку $P \in E$ посредством алгоритма 7.2.1;
2. [Проверить кратное]
   $Q = [f]P$;
   if($Q == O$) goto [Найти начальную точку];
   return $Q$; // Точка простого порядка $r$.

Пусть этот алгоритм почти тривиален, но он важен в криптографических приложениях. Одним из таких приложений является эллиптическая подпись. Существует стандартная схема цифровой подписи на эллиптических кривых, которая работает следующим образом и, как ясно с самого начала, использует точку простого порядка.

**Алгоритм 8.1.8** (алгоритм цифровой подписи на эллиптических кривых (ECDSA)). Этот алгоритм предоставляет функции для генерации ключей, формирования подписи и верификации сообщений. Для сообщения используется общее обозначение $M$, $M$ — целое, и предполагается наличие подходящей хэш-функции $h$.

1. [Алиса генерирует ключ]
   Алиса выбирает кривую $E$, порядок которой $\#E = fr$, где $r$ — «большое» простое число;
   Алиса находит точку $P \in E$ порядка $r$ посредством алгоритма 8.1.7;
   Алиса выбирает случайное $d \in [2, r-2]$;
   $Q = [d]P$;
   Алиса делает общедоступным открытый ключ $(E, P, r, Q)$; // Закрытым ключом является $d$.
2. [Алиса формирует подпись]
   Алиса выбирает случайное $k \in [2, r-2]$;
   $(x_1, y_1) = [k]P$;
   $R = x_1 \bmod r$; // Заметим, что $R \neq 0$.
   $s = k^{-1}(h(M) + Rd) \bmod r$;
   if($s == 0$) goto [Алиса формирует подпись];
   Подписью Алисы является пара $(R, s)$, передаваемая с сообщением $M$;
3. [Боб проверяет]
   Боб получает открытый ключ Алисы $(E, P, r, Q)$;
   $w = s^{-1} \bmod r$;
   $u_1 = h(M)w \bmod r$;
   $u_2 = Rw \bmod r$;
   $(x_0, y_0) = [u_1]P + [u_2]Q$;
   $v = x_0 \bmod r$;
   if($v == R$) Боб принимает подпись;
   else Боб отвергает подпись;

Этот алгоритм составлен по образцу более раннего стандарта DSA и представляет собой не что иное, как естественный вариант DSA для эллиптических кривых. Современные детали и вопросы освещаются в работе [Johnson et al. 2001]. Значение хэш-функции $h(M)$, выражаясь техническим языком, предполагается вырабатывать посредством другого стандарта — хэш-функции SHA-1 (см. [Jurišić and Menezes 1997]). Указанные авторы освещают также интересный вопрос безопасности. Они делают вывод, что 1024-битовая система DSA примерно так же безопасна, как 160-битовая система ECDSA. Если так, то это наблюдение вновь подтверждает на современном уровне представлений, что задача EDL трудна почти настолько, насколько вообще может быть трудна вычислительная теоретико-числовая задача.

Текущий рекорд в задаче вычисления EDL был установлен в рамках программы «Certicom Challenge», для которой в 2002 г. Монико и др. была решена задача EDL для эллиптической кривой над полем $\mathbf{F}_p$, где $p$ — 109-битовое простое число. Следующее число, предлагаемое в качестве вызова в этом списке, является 131-битовым простым числом, но на современном уровне представлений о сложности задачи EDL этот 131-битовый случай, вероятно, *в две тысячи раз* сложнее, чем упомянутый выше 109-битовый случай.

Мимоходом отметим, что существует другой путь, ведущий к получению схемы подписи на эллиптических кривых, которая называется схемой Эль Гамаля. Мы не выписываем этот алгоритм — он не является таким общепринятым стандартом, как приведенная схема ECDSA (но он не менее интересен) — однако, его основные моменты содержатся в алгоритме 8.1.10. Кроме того, стоящие за ним теоретические идеи можно найти в статье [Koblitz 1987].

В связи с обменом ключом по Диффи—Хеллману мы упоминали практический прием, состоящий в использовании изощренных методов (к ним относятся также RSA, ECC) для обмена ключом и в последующем использовании известного обеим сторонам ключа в быстром блочном шифре, таком, как DES, например. Но существует и другой удивительный путь, ведущий к своего рода «прямой» ECC-схеме, основанной на понятии встраивания открытого текста в точки на эллиптических кривых. На этом пути все стадии процесса шифрования/дешифрования не требуют ничего, кроме эллиптической алгебры.

**Теорема 8.1.9** (теорема о встраивании открытого текста). *Пусть для простого числа $p > 3$ через $E$ обозначена эллиптическая кривая над $\mathbf{F}_p$, заданная уравнением*

$$y^2 = x^3 + ax + b.$$

*Пусть $X$ — любое целое число из промежутка $[0, p-1]$. Тогда $X$ есть либо $x$-координата некоторой точки на кривой $E$, либо $x$-координата на ее искажении $E'$, которое задается уравнением $gy^2 = x^3 + ax + b$ при некотором $g$ с условием $\left(\frac{g}{p}\right) = -1$. Более того, если $p \equiv 3 \pmod 4$, и мы положим*

$$s = X^3 + aX + b \bmod p,$$

$$Y = s^{(p+1)/4} \bmod p,$$

*то $(X, Y)$ есть точка на $E$ или $E'$ соответственно при*

$$Y^2 \equiv s,\ -s \pmod{p},$$

*где в последнем случае в качестве уравнения кривой $E'$ мы берем уравнение $-y^2 = x^3 + ax + b$.*

Эта теорема без труда доказывается с помощью тех же самых алгебраических свойств искажения, с которыми мы встречались в теореме 7.5.2 и упр. 7.18; она ведет к следующему алгоритму шифрования с прямым встраиванием.

**Алгоритм 8.1.10** (шифрование с прямым встраиванием в ECC). Данный алгоритм позволяет проводить шифрование/дешифрование, пользуясь исключительно эллиптической алгеброй, т. е. без использования промежуточного шифра, посредством прямого встраивания открытого текста в кривые. Предположим, что Алиса и Боб договорились об открытой кривой $E_{a,b}(\mathbf{F}_p)$ с искажением $E'$, на которых лежат, соответственно, открытые точки $P, P'$. В дополнение к этому предполагается, что Боб сгенерировал соответствующие открытые ключи $P_{\text{B}} = [K_{\text{B}}]P, P'_{\text{B}} = [K_{\text{B}}]P'$, как это делается в алгоритме 8.1.6. Через $X$ мы обозначаем блок открытого текста (целое число из промежутка $[0, \ldots, p-1]$), который Алиса желает зашифровать для Боба.

1. [Алиса встраивает открытый текст $X$]
   Алиса выбирает из кривых $E$ и $E'$ ту, на которой $X$ является допустимой $x$-координатой (а если нужны $y$-координаты, вычисляет соответствующее число $Y$), пользуясь теоремой 8.1.9, и берет кривую $E$, если $X$ лежит на обеих кривых;
   // См. упр. 8.5.
   В зависимости от того, какая из кривых $E, E'$ работает, Алиса, соответственно, устанавливает:
   $d = 0$ или $1$; // Бит выбора кривой.
   $Q = P$ или $P'$;
   $Q_{\text{B}} = P_{\text{B}}$ или $P'_{\text{B}}$.
   Алиса выбирает случайное $r \in [2, p-2]$;
   $U = [r]Q_{\text{B}} + (X, Y)$; // Эллиптическое сложение,
   // скрывающее открытый текст.
   $C = [r]Q$; // «Ключ» для возврата к открытому тексту.
   Алиса передает блок (зашифрованное сообщение, ключ, бит), который есть $(U, C, d)$;
2. [Боб дешифрует с целью получения открытого текста $X$]
   Боб анализирует $d$, чтобы определить кривую, на которой будет осуществляться эллиптическая алгебра;
   $(X, Y) = U - [K_{\text{B}}]C$; // Закрытый ключ,
   // примененный вместе с эллиптическим вычитанием.
   Боб теперь восстанавливает открытый текст, который есть $x$-координата $X$;

В этом методе можно распознать схему встраивания Эль Гамаля, в которую мы внесли некоторые усовершенствования по сравнению с предыдущими изло-

жениями ([Koblitz 1987], [Kaliski 1988]). Заметим, что последняя часть теоремы 8.1.9 позволяет алгоритму 8.1.10 работать эффективно, когда для характеристики основного поля справедливо сравнение $p \equiv 3 \pmod 4$. В практических реализациях алгоритма 8.1.10 можно применить два других существенных усовершенствования. Во-первых, в $y$-координатах нет необходимости, если везде в этом алгоритме использовать координаты Монтгомери (алгоритм 7.2.7) и аккуратно применять алгоритм 7.2.8 в соответствующих местах. Во-вторых, в этом алгоритме из-за «точки-ключа» $C$ происходит удваивание передаваемого объема данных. Этого также можно избежать, соответствующим образом настроив протокол обмена случайным числом, так что случайное число $r$ будет детерминированно синхронизироваться двумя сторонами. (Авторы выражают признательность Б. Гарсту за это замечание, которое, фактически, привело к регистрации патента США (U. S. Patent) [Crandall and Garst 2001].) См. упр. 8.3 для выяснения дальнейших подробностей об этих усовершенствованиях. Если все правильно сделать, то получится достаточно эффективная, изящная схема с прямым встраиванием, асимптотически не увеличивающая объем данных.

### 8.1.4. Протокол подбрасывания монеты

В криптографии протоколом является, по существу, алгоритм, указывающий в определенном порядке те шаги, которые должны предпринять участвующие в нем стороны. Мы уже встречались с протоколом обмена ключом и связанными с ним протоколами. Здесь мы исследуем занимательное приложение теоретико-числовых протоколов в культурной сфере. Как можно честно бросить жребий по телефону? Или организовать игру в покер между $n$ пользователями, играющими «втемную» по сети? Мы предполагаем самое худшее: ни одна из сторон не хочет уступать, но необходимо принять решение — в такой ситуации его можно принять, бросив жребий, при этом одна из сторон объявляет исход: «орел» или «решка». Оказывается, такая удаленная жеребьевка в самом деле возможна, если воспользоваться свойствами определенных сравнений.

Кстати, сама потребность в протоколе подбрасывания монеты становится очевидной, если представить себе телефонные переговоры — скажем, между двумя враждующими сторонами, участвующими в судебном процессе — в которых некий важный результат будет являться итогом подбрасывания монеты; под ним понимается случайный бит, на статистику которого не может оказывать влияние ни одна из сторон. Вариант с заявлением одной из сторон о том, что у них только что выпал орел и потому они выиграли, очевидно, не подходит. Честность каждого участника необходимо контролировать, а это можно сделать с помощью искусного применения сравнений, в которых участвуют простые или определенного вида составные числа. Далее приведен один из подходов, в котором мы следуем некоторым идеям из книги [Bressoud and Wagon 2000], касающимся простых протоколов.

**Алгоритм 8.1.11** (протокол подбрасывания монеты). **Алиса и Боб намереваются «подбросить симметричную монету», используя для общения лишь канал**

связи. Они договорились, что если Боб в описанной ниже ситуации угадывает верно, то он выигрывает, иначе выигрывает Алиса.

1. [Алиса выбирает простые числа]
   Алиса выбирает два больших простых числа так, что $p < q$, образует число $n = pq$ и выбирает случайное простое число $r$, такое, что $\left(\frac{n}{r}\right) = -1$;
2. [Алиса отправляет Бобу частичную информацию]
   Алиса отправляет Бобу $n$ и $r$;
3. [Боб выбирает]
   Боб делает выбор между утверждением «наименьший простой делитель числа $n$ есть квадратичный вычет по модулю $r$» и утверждением «наибольший простой делитель числа $n$ есть квадратичный вычет по модулю $r$» и отправляет свой выбор Алисе;
4. [Алиса объявляет победителя]
   Алиса объявляет, прав Боб или не прав, и отправляет ему простые числа $p, q$, так что Боб может сам убедиться в том, что она не обманывает;

Интересно исследовать криптографическую цельность этого алгоритма (см. упр. 8.8). Хотя мы описали этот алгоритм в терминах победителя и побежденного, ясно, что Алиса и Боб могли бы использовать тот же самый метод, чтобы просто выработать случайный бит, скажем, «0», если выиграет Алиса, и «1», если выиграет Боб. Указанный протокол подбрасывания монеты имеет множество вариантов. Например, в книге [Schneier 1996] приводится протокол, в котором Алиса генерирует четыре квадратных корня из числа по модулю $n = pq$ и отправляет их Бобу, а Боб генерирует случайный квадратичный вычет по модулю $n$. Этот сценарий не обладает простотой алгоритма 8.1.11, но имеет много интересных особенностей. Например, с помощью его модификации можно обойти специфический сценарий Микали, в котором Боб намеренно проигрывает [Schroeder 1999]. Также имеются алгоритмы, основанные на числах Блюма и, в целом, на том факте, что произведение $pq$ допускает существование многих корней (см. упр. 8.7). Эти идеи естественным образом обобщаются на протокол игры в покер, когда несколько игроков объявляют, что у них на руках, и на тому подобные протоколы [Goldwasser and Micali 1982].

## 8.2. Генерирование случайных чисел

Как известно, задача генерирования случайных чисел возникает еще на заре компьютерного века (не позднее, чем в 40-х годах XX столетия). И тогда уже становится ясно, что формирование случайных чисел при помощи машинной арифметики в принципе невозможно, говоря словами Дж. фон Неймана, «порочно по сути»[2]. Несмотря на то, что компьютер в некотором смысле может гарантировать почти случайное распределение, в действительности известные способы компьютерных вычислений обладают детерминированностью, так что

[2] В оригинале «in a state of sin». — *Прим. перев.*

само понятие случайности приходится переводить на язык машин Тьюринга и последовательных алгоритмов. Читателю, сомневающемуся в существовании более «случайной» технологии (к слову, «абсолютная» случайность невозможна в смысле теории вероятностей), мы предложим следующий экзотический способ: представим, что с помощью приемника волн высокой частоты мы можем «прослушивать» удаленные галактики. Посредством оцифровки реликтового излучения черных тел молодого космоса, сгенерируем случайный поток битов. Мы не утверждаем, что вселенная действительно случайна, но мы можем предполагать, что сигнал, излучаемый отдаленными галактиками настолько непредсказуем, насколько это вообще возможно.

Сегодня моделирование реальной случайности важно как никогда. В частности, многие криптографические системы используют сгенерированные случайным или по возможности близким к случайному образом числа. Детерминированное устройство, генерирующее числа, случайные, как может показаться постороннему наблюдателю, можно использовать, чтобы построить простую криптосистему. Сгенерируйте случайный поток битов. Чтобы зашифровать сообщение, побитно вычислите суммы по модулю два соответствующих битов сообщения и сгенерированного потока. Для расшифровки опять проделайте ту же операцию, с тем же самым случайным потоком. Такая криптосистема в принципе защищена от взлома, если только не вмешиваются обычные в таких случаях обстоятельства: сообщение оказывается длиннее, чем случайный поток, или тот же случайный поток используется для шифровки других сообщений, либо злоумышленник сумел завладеть какой-либо информацией о генераторе и т.д. Невзирая на подобные практические трудности, эта схема служит иллюстрацией фундаментального кредо криптографии: как-нибудь используй что-то, что не может знать посторонний.

Кажется, что не реже, чем разрабатывается новый способ генерации случайных чисел, обнаруживается, что некоторая более старая схема оказывается неслучайной настолько, что, скажем, ставит под угрозу защищенность криптосистемы или же влияет на корректность испытаний по методу Монте-Карло. Мы сделаем краткий обзор способов генерации случайных чисел, естественно, с позиции применения простых чисел.

### 8.2.1. Модулярные методы

Линейный конгруэнтный метод стал основным рабочим инструментом в теории генераторов случайных чисел. Он использует целочисленные итерации

$$x_{n+1} = (ax_n + b) \bmod m,$$

где $a, b, m$ — целые константы, причем $m > 1$. Задано также исходное состояние, скажем, $x_0$. По сей день появляются все новые и новые результаты исследований эффективности этого и подобных ему генераторов. Одна из разновидностей называется мультипликативным конгруэнтным генератором с шагом рекурсии

$$x_{n+1} = (cx_n) \bmod m,$$

здесь стартовое значение $x_0$ предполагается взаимно простым с $m$. Для прикладных задач, где возникает необходимость реализовать функцию $random()$, принимающую значения из интервала $[0, 1)$, стандартным приемом является использование $x_n/m$.

Рекуррентные последовательности, такие как две приведенных выше, часто оказываются периодическими. Для генерирования случайных чисел желательно использовать рекурсию с большим периодом. Легко заметить, что период линейного конгруэнтного генератора не превосходит $m$, а мультипликативного конгруэнтного генератора — $m - 1$. При некоторых ограничениях на параметры конгруэнтный генератор будет генерировать последовательность $(x_n)$ с полным периодом $m$ в линейном варианте, и $m - 1$ — в мультипликативном. В следующей теореме перечислены основные свойства этих генераторов:

**Теорема 8.2.1** (Лемер). *Линейный конгруэнтный генератор, определяемый рекуррентным соотношением*

$$x_{n+1} = (ax_n + b) \bmod m$$

*обладает периодом $m$ тогда и только тогда, когда*

*(1)* $\text{НОД}(b, m) = 1$,
*(2)* $p|a - 1$, *если* $p$ — *простое и* $p|m$,
*(3)* $4|a - 1$, *если* $4|m$.

*Далее, мультипликативный конгруэнтный генератор, определяемый рекуррентным соотношением*

$$x_{n+1} = (cx_n) \bmod m$$

*имеет период $m - 1$ тогда и только тогда, когда*

*(1)* $m$ — *простое,*
*(2)* $c$ *является примитивным корнем по модулю* $m$,
*(3)* $x_0 \not\equiv 0 \pmod m$.

Во многих компьютерных системах до сих пор применяется линейная схема, хотя у нее имеются известные недостатки, о которых мы еще поговорим.

Прежде мы рассмотрим конкретный пример стандартного линейного конгруэнтного генератора:

**Алгоритм 8.2.2** (32-разрядный генератор случайных чисел (Кнут, Левис)). Этот генератор обладает сравнительно неплохими статистическими характеристиками. Алгоритм состоит из функций инициализации и генерации псевдослучайного числа. В качестве модуля генератора выберем $M = 2^{32}$. Операции по модулю $M$ производятся быстро, достаточно взять логическое поразрядное «И» (&) с числом $M - 1$. Случайная последовательность генерируется при последовательных вызовах функции $random()$, первоначальная инициализация производится процедурой $seed()$.

1. [Процедура *seed*]
   *seed*() {

```
        Выбор начального значения x;        // x — целое число из [0, M − 1].
        return;
    }
2. [Функция random]
    random() {
        x = (1664525x + 1013904223) & (M − 1);
        return x;                                   // Новое случайное число.
    }
```

Заметим, что операция поразрядного логического «И» с числом $M-1$ состоит в выборе 32 младших разрядов выбранного числа. Устроенный аналогично известный генератор

$$x_{n+1} = (16807x_n) \bmod M_{31},$$

где $M_{31} = 2^{31} - 1$ — число Мерсенна, показал неплохие результаты в ряде экспериментальных тестов (но не во всех), см. работы [Park and Miller 1988], [Press et al. 1996].

Интересное усовершенствование стандартного конгруэнтного генератора было предложено в работе [Wu 1997]. Рекурсивный шаг алгоритма

$$x_{n+1} = \left((2^{30} - 2^{19})x_n\right) \bmod M_{61}.$$

Тот факт, что $M_{61}$ число Мерсенна, позволяет сравнительно быстро проводить вычисления.

**Алгоритм 8.2.3** (быстрый 61-разрядный генератор случайных чисел). Алгоритм состоит из функций инициализации и генерации случайных чисел по методу By с модулем генератора $M = 2^{61} - 1$ и множителем $c = 2^{30} - 2^{19}$. Несмотря на то, что алгоритм включает операцию умножения по модулю, в данном случае все необходимые операции — это сложения/вычитания, левые/правые сдвиги (<< / >>, соответственно), операция логического поразрядного «И» (&), которая играет роль приведения по модулю, mod.

```
1. [Процедура seed]
    seed() {
        Выбор начального значения x;        // x целое число из [1, M − 1].
        return;
    }
2. [Функция random]
    random() {
        x = (x >> 31) + ((x << 30)&M) − (x >> 42) − ((x << 19)&M);
        if(x < 0) x = x + M;
        return x;                                   // Новое случайное число.
    }
```

Благодаря сдвигам и операциям логического «И», данный алгоритм не содержит явно умножений и делений. Более того, как отмечено в работе [Wu 1997], генератор отлично зарекомендовал себя при проверке на нескольких основных статистических критериях. Конечно, как и любой машинный

генератор, он также допускает обобщение, надо только правильно выбрать параметры; например, чтобы получить достаточно длинную случайную последовательность, параметры $c, M$ должны быть подобраны так, чтобы $c$ было примитивным корнем по модулю простого числа $M$. Необходимо сделать предостережение: самые последние исследования показывают, что у генератора, построенного по алгоритму 8.2.3, есть недостатки. Такие генераторы действительно успешно проходят проверку спектральными критериями, но на ряде битовых статистик они дают отрицательный результат, что показано в работе [L'Ecuyer and Simard 1999]. Даже при этом есть весьма веские основания, чтобы использовать этот генератор за его быстродействие, простоту в реализации и удовлетворительное соответствие многим, пусть и не всем, статистическим критериям.

Известно огромное множество модификаций конгруэнтных генераторов. Интересная идея позволяет строить генераторы с чрезвычайно большим периодом. В генераторе случайных чисел, который мы собираемся рассмотреть, используются умножения матрицы на вектор. Пусть $\mathbf{T}$ — матрица размера $k \times k$ и $\vec{x}$ — вектор длины $k$. Определим правило вычисления следующего вектора в последовательности как $\vec{x} = \mathbf{T}\vec{x}$, пусть также у нас имеется правило извлечения битов или значений из заданного вектора.

**Теорема 8.2.4** (Голомб). *Для заданного простого числа $p$ через $\mathbf{M}_k(p)$ обозначим группу несингулярных матриц размера $k \times k$ с коэффициентами* $(\mathrm{mod}\ p)$. *Пусть $\vec{x}$ — отличный от нуля вектор из $\mathbf{Z}_p^k$. Тогда для того чтобы период последовательности*

$$\vec{x},\ \mathbf{T}\vec{x},\ \mathbf{T}^2\vec{x},\ \ldots$$

*был равен $p^k - 1$, необходимо и достаточно, чтобы порядок $\mathbf{T} \in \mathbf{M}_k(p)$ был равен $p^k - 1$.*

С помощью этой изящной теоремы строится итерационный генератор таким же образом, как и в предыдущих примерах. Правда, в работах [Golomb 1982] и [Marsaglia 1991] отмечается, что есть куда более эффективные способы генерирования последовательностей со сверхбольшими периодами. Так как в данном случае генератор определяется посредством итераций, будет уместно помимо основной теоремы привести и описание алгоритма.

**Алгоритм 8.2.5** (генератор случайных чисел с большим периодом). Получая в качестве входов целые числа $b \geqslant 2$ и $r > s > 0$, алгоритм генерирует последовательность псевдослучайных чисел, каждое вычисляется по $r$ предыдущим членам и текущему биту переноса $c$. Исходной точкой служит вектор $\vec{v}$, первые $r$ компонент которого принадлежат $[0, b-1]$, а последняя компонента $c = 0$ или 1.

1. [Процедура *seed*]
   *seed*() {
   Выбор параметров $b \geqslant 2$ и $r > s > 0$;
   Инициализация вектора значений и переноса: $\vec{v} = (v_1, \ldots, v_r, c)$;
   return;
   }

2. [Функция $random$]
   $random()$ {
      $x = v_s - v_r - c$, // Новое значение $x$ вычисляется как функция от предыдущих значений.
      if($x < 0$) {
         $x = x + b$;
         $c = 1$; // Производится «вычитание с заимствованием».
      } else $c = 0$;
      $\vec{v} = (x, v_1, \ldots, v_{r-1}, c)$, // Сдвиг выбрасывает прежнее значение $v_r$
      return $x$; // Новое случайное число
   }

Практический эффект от применения алгоритма, прямо скажем, впечатляет. Например, выбрав параметры $b = 2^{64}, r = 30, s = 6$ и выполняя итерации

$$x_0 = v_6 - v_{30} - c,$$

с учетом приведения по модулю, переноса и сдвига, как это описано в алгоритме 8.2.5, получаем последовательность с периодом

$$P \approx 10^{578}.$$

Это лишь одно из многих замечательных следствий следующей теоремы:

**Теорема 8.2.6** (Марсалья). *Период генератора случайных чисел, построенного с помощью алгоритма* 8.2.5, *равен*

$$P = \varphi(b^r - b^s + 1).$$

Таким образом, точное значение периода для последнего примера составляет

$$\varphi\left(2^{64\,30} - 2^{64\,6} + 1\right) = 2^{64\,30} - 2^{64\,6} \approx 10^{578},$$

так как аргумент функции $\varphi$ — простое число. Отметим, что одно и то же значение может встречаться в периоде генерируемой последовательности несколько раз, что, однако, не приводит к совпадению последующих членов. Длину периода определяет вектор $\vec{v}$, внутренний параметр алгоритма. Поскольку существует порядка $b^r$ возможных векторов $\vec{v}$, теорема Марсальи, в каком-то смысле, соответствует интуитивным представлениям.

Другой пример итерационного генератора — дискретный экспоненциальный генератор с шагом итерации

$$x_{n+1} = g^{x_n} \pmod N,$$

где $g, x_0, N$ заданы. В работах [Blum et al. 1986], [Lagarias 1990], [Friedlander et al. 2000] получено несколько надежных результатов, которые можно использовать для защиты информации. Большое внимание всегда уделяется вопросу генерирования «защищенного» случайного бита с наименьшими вычислительными затратами. Известно, что если выбирается лишь один из разрядов числа $x_n$, то тем самым, в определенном смысле, гарантируется защищенность, но

ценой большого количества вычислительных операций на один сгенерированный бит. В работе [Patel and Sundaram 1998] показано, что можно использовать большую часть разрядов $x_n$ так, что результат остается защищенным в криптографическом смысле. В этом случае на один бит затрачивается значительно меньшее количество операций.

Сегодня используется и множество других генераторов, в частности, сдвиговые, хаотические и клеточные автоматы (CA). Некоторые генераторы к настоящему времени признаны криптографией ненадежными, прежде всего это относится к простейшим конгруэнтным генераторам, даже использующим не линейную, а полиномиальную форму более высокой степени (см. [Lagarias 1990]). Одна из проблем, стоящих перед исследователями, состоит в том, что достаточно надежные генераторы, как, например, дискретный экспоненциальный с различными вариациями, безопасность которого основывается на сложности дискретного логарифмирования, работают медленно. По счастью, разработан ряд стандартных тестов на случайность, особенно для последовательностей двоичных битов, которые всегда могут быть призваны, чтобы развенчать данный метод генерирования или же вселить некоторую уверенность относительно его надежности (см. [Menezes et al. 1997]).

Плодотворная с точки зрения безопасности идея, предложенная Миллером, состоит в использовании линейного конгруэнтного генератора, в котором сложение производится на группе точек эллиптической кривой. Рассмотрим эллиптическую кривую $E$ над конечным полем. Выбрав целое число $a$ и точку $B \in E$, можно сформировать последовательность посредством итераций

$$P_{n+1} = [a]P_n + B, \tag{8.1}$$

где сложение понимается в эллиптическом смысле. Стартовым значением здесь является некоторая точка $P_0 \in E$. В качестве случайного элемента поля может выступать $x$-координата точки $P_n$. Данная схема, очевидно, не может быть взломана так же просто, как и обычная линейная конгруэнтная схема. Важно, что для стандартных значений множителей $a$, скажем, степеней двойки, метод сравнительно эффективен в силу простоты ступенчатого алгоритма эллиптического умножения. Конечно, можно воспользоваться приведенными операциями, следуя алгоритму 7.2.8. Другими словами, рассматривать только $x$-координаты, не заботясь о неопределенности операции $[a]P \pm B$, что фактически освобождает от выполнения групповых операций сложения, но вместо этого приходится извлекать квадратные корни.

Другой подход к построению генераторов случайных чисел с помощью эллиптических кривых был предложен в работе [Gong et al. 1999]. А именно, на эллиптические кривые над $\mathbf{F}_{2^m}$ были распространены ранее известные понятия сдвиговых регистров и кодовых слов (см. упр. 8.29).

А теперь ненадолго остановимся на вопросе генерирования случайной последовательности битов. Конечно, мы всегда можем воспользоваться каким-либо «хорошим» генератором случайных чисел, выбирая некоторый бит — для определенности, младший — из каждого числа. Но читатель, возможно, будет удивлен, например, тому, что в результате вычисления символов Лежандра,

изначально предназначенного для поиска точек на эллиптической кривой,

$$\left(\frac{x^3+ax+b}{p}\right)=\pm 1,$$

где $x$ последовательно пробегает целочисленные значения из некоторого интервала, а (редкие) нулевые результаты отбрасываются, скажем, получается статистически приемлемая случайная последовательность из $\pm 1$. Еще более удивительно, что если аргумент $x$ в таком символе Лежандра генерировать на линейном конгруэнтном или каком-либо ином генераторе, то последовательность станет исключительно случайной в смысле любой статистики.

Указанный способ генерирования случайных потоков необходимо сравнить с обычными генераторами, основанными на применении логической операции «исключающего ИЛИ». Пример такого сравнения приводится в работе [Press et al. 1996], в нем используется примитивный многочлен (mod 2)

$$x^{18}+x^5+x^2+x+1.$$

(Многочлен над конечным полем $F$ называется *примитивным*, если он неприводим и корень его порождает циклическую мультипликативную группу поля.) Пусть $x_{-1}$ — «текущее» значение случайного бита; обозначим предыдущие 17 битов через $x_{-2}, x_{-3}, \ldots, x_{-18}$. Тогда новый бит $x_0$ следующим образом вычисляется с помощью логики сдвигов, соответствующей данному многочлену,

$$\begin{aligned} x_0 &= x_{-18}, \\ x_{-5} &= x_{-5} \wedge x_0, \\ x_{-2} &= x_{-2} \wedge x_0, \\ x_{-1} &= x_{-1} \wedge x_0, \end{aligned}$$

где символом «$\wedge$» обозначен оператор логического «исключающего ИЛИ» (совпадающий с оператором сложения в поле четной характеристики). Затем все индексы должны быть сдвинуты так, чтобы через $x_{-1}$ обозначался новый бит — $x_0$ в приведенной выше записи. Приведем точное описание алгоритма:

**Алгоритм 8.2.7** (быстрый и простой генератор случайных битов). Данный алгоритм состоит из функций инициализации начальных значений и генерации случайного бита, соответствующей многочлену $x^{18}+x^5+x^2+x+1$ над $\mathbf{F}_2$.

1. [Процедура *seed*]
   *seed*() {
   $h = 2^{17}$; // 100000000000000000 в двоичной системе счисления.
   $m = 2^0 + 2^1 + 2^4$; // Двоичная маска 10011.
   Выбор целочисленного значения $x_0$ из $[1, 2^{18}]$;
   return;
   }
2. [Функция *random* возвращает 0 или 1]
   *random*() {

```
        if((x & h) ≠ 0) { // Результат поразрядной конъюнкции чисел x, h
сравнивается с 0.
            x = ((x ∧ m) << 1) | 1;     // Вычисляются логические операции
«ИЛИ» (|) и «исключающего ИЛИ» (∧).
            return 1;
        }
        x = x << 1;
        return 0;
    }
```

В той же работе [Press et al. 1996] рассматривается ряд других многочленов (mod 2) со степенями вплоть до 100 и выше.

Всякий раз при внимательном изучении теории генераторов случайных чисел прослеживается концептуальная обратная связь с теорией простых чисел. Не только многие предложенные генераторы случайных чисел используют простые числа как таковые, но и многие алгоритмы, в том числе некоторые из описанных в настоящей книге, используют подходящие выборки случайных чисел. Но когда необходима проверка случайности на статистических тестах, а так обыкновенно и происходит, то случайные последовательности выступают совсем в другой роли. В данном контексте имеются в виду методы квази-Монте-Карло (quasi-Monte Carlo — qMC), к которым мы и переходим.

## 8.3. Методы квази-Монте-Карло

Кто бы мог подумать во времена Гаусса, Эйлера, Лежандра, скажем, в то, что простые числа будут применяться на практике при анализе финансовых рынков в XX столетии? Мы имеем здесь в виду вовсе не криптографию — она и так возникает всякий раз, когда речь заходит о деньгах, — а методы квази-Монте-Карло, которые, приблизительно говоря, относятся к специальному разделу анализа по методу Монте-Карло (т. е. статистическому анализу). Вычисления по методу Монте-Карло лежат в основе целых областей прикладной науки.

Существенная часть метода Монте-Карло состоит в выборе некоторого большого непрерывного (или даже дискретного, при необходимости) пространства, из которого выбираются случайным образом точки, скажем, для вычисления многомерного интеграла. Следует ожидать, что «в среднем» результат будет близок к истинному значению, которое теоретически можно было бы вычислить, проведя бесконечное число испытаний. Любопытно, что теория чисел — и, в частности, теория простых чисел — может быть изложена на основе методов квази-Монте-Карло (qMC). Методы qMC отличаются от традиционного метода Монте-Карло тем, что не нуждаются в абсолютно случайной последовательности испытаний. Вместо этого рассматриваются псевдослучайные последовательности, которые в действительности не являются случайными в строго статистическом смысле, но обладают некоторыми общими свойствами, практически отвечающими сути решаемой проблемы.

Хоть это и слишком упрощенно, наглядно представить себе разницу между случайными методами и qMC можно так: случайным образом выбранные

точки должны образовывать «скопления» и «пустоты», в то время как точки qMC в принципе *избегают друг друга*, минимизируя скученность, и стремятся *заполнять пустоты*. По этим причинам qMC-испытания могут — в зависимости от размерности пространства и конкретной постановки проблемы — быть предпочтительней для решения таких задач, как численное интегрирование, нахождение минимума/максимума и статистическое оценивание вообще.

### 8.3.1. Теория дискрепанса

Предположим, что необходимо вычислить значение некоторого интеграла по $D$-размерной области $R$, а именно,

$$I = \int \int \ldots \int_R f(\vec{x})\, d^D \vec{x},$$

однако нет никакой надежды получить решение в явном, аналитическом виде. В этом случае можно воспользоваться методом Монте-Карло. Вычисление среднего значения от подстановок в подынтегральное выражение $N$ «случайных» векторов $\vec{x} = (x_1, \ldots, x_D)$ из области интегрирования с последующим домножением его на меру области $R$ дает приближение, обозначим его через $I'$, точного значения интеграла $I$. Из основных дисперсионных законов статистики мы можем ожидать, что погрешность составит не менее

$$|I' - I| = O\left(\frac{1}{\sqrt{N}}\right),$$

где величина мультипликативной константы зависит, естественно, от размерности $D$, подынтегральной функции $f$ и области $R$. Интересно, однако, что асимптотический закон $N^{-1/2}$ инвариантен относительно $D$. Для сравнения, так называемый «сеточный» метод, в котором из области интегрирования $R$ выбираются точки узлов разбиения, приводит к погрешности, оцениваемой не меньше, чем

$$|I' - I| = O\left(\frac{1}{N^{1/D}}\right).$$

Эта величина, особенно при больших $D$, может быть совершенно неприемлемой. В действительности, метод сеток — за редкими исключениями — практически применим только для вычисления одно- или двумерных интегралов, если только подынтегральная функция не имеет специальный вид, или есть особые причины использовать сетку и т.п. Понятно, почему метод Монте-Карло со случайным выбором множества точек использовался для численного интегрирования в пространствах размерности 3 и более.

Замечательно однако, что стандартный метод Монте-Карло может быть усовершенствован таким образом, что реальная погрешность будет иметь величину

$$|I' - I| = O\left(\frac{\ln^D N}{N}\right),$$

причем в некоторых случаях порядок понижается до $\ln^{D-1} N/N$, в зависимости от реализации (этот вопрос в свое время будет нами рассмотрен). Идея состоит в использовании последовательностей с малым дискрепансом (см. определение 8.3.1), относящихся к классу квази-Монте-Карло (qMC) последовательностей (некоторые авторы *определяют* последовательности с малым дискрепансом как имеющие оценку для $|I' - I|$ такую, как указано выше; см. упр. 8.32). Прежде всего важно отметить, что qMC-последовательности *не* являются случайными в классическом смысле. Фактически, точки таких последовательностей дистанцируются друг от друга. Предполагается, что точки настоящих случайных последовательностей образуют «скопления» и «пустоты», тогда как точки qMC-последовательностей заполняют области в некотором смысле более равномерно (см. упр. 8.12).

Мы начинаем обзор qMC-методов с определения понятия дискрепанса, считая вещественнозначными компоненты векторов, лежащих в области $R$.

**Определение 8.3.1.** Пусть $P$ — множество, содержащее не менее $N$ точек в области $R = [0,1]^D$ (единичный $D$-размерный куб). Дискрепанс множества $P$ относительно семейства $F$ измеримых по Лебегу областей $R$ определяется (обозначения $D_N$ и $D_N^*$ не связаны с размерностью $D$) формулой

$$D_N(F;P) = \sup_{\phi\in F}\left|\frac{\chi(\phi;P)}{N} - \mu(\phi)\right|,$$

где $\chi(\phi;P)$ — это число точек из $P$, лежащих в $\phi$, а через $\mu$ обозначена лебегова мера. Далее, предельный дискрепанс множества $P$ определим как

$$D_N(P) = D_N(G;P),$$

где $G$ — семейство подобластей вида $\prod_{i=1}^D [u_i, v_i)$. Кроме того, определим $\star$-дискрепанс множества $P$ как

$$D_N^*(P) = D_N(H;P),$$

где $H$ обозначает семейство подобластей вида $\prod_{i=1}^D [0, v_i)$. Наконец, если $S \subset R$ — счетная бесконечная последовательность, $S = (\vec{x}_1, \vec{x}_2, \ldots)$, то понятия дискрепансов $D_N(S)$ вводятся строго в терминах первых $N$ членов последовательности $S$.

Данное определение выглядит терминологически перегруженным, однако несложное рассуждение показывает, что за ним скрывается просто оценка того, насколько точно множество $P$ соответствует области. На первый взгляд может показаться, что оптимальный дискрепанс должен достигаться на обыкновенной равномерной сетке, однако в случае пространства более чем одной размерности это интуитивное представление приводит к заблуждению, как мы увидим. Хороший способ прояснить смысл понятия дискрепанса состоит в изучении теоремы: счетное бесконечное множество $S$ равномерно распределено в $R = [0,1]^D$ тогда и только тогда, когда $\star$-дискрепанс (равносильным образом, предельный дискрепанс) стремится к нулю при $N \to \infty$. Кроме того, понятия предельного и $\star$-дискрепансов, в общем-то, не являются независимыми;

на самом деле можно показать, что для любого множества $P$ в обозначениях введенного выше определения выполняется

$$D_N^*(P) \leqslant D_N(P) \leqslant 2^D D_N^*(P).$$

С этими результатами можно ознакомиться по работам [Niederreiter 1992] и [Tezuka 1995].

Важность понятия дискрепанса — и, в частности, $\star$-дискрепанса $D^*$ — сразу становится понятной из следующего фундаментального результата, который можно положить в основу qMC-теории интегрирования. Здесь следует упомянуть ограниченную вариацию Харди—Краузе $H(f)$, оценивающую отклонения функции $f$. Точное определение $H$ нам не понадобится (но можно посмотреть в работе [Niederreiter 1992]), так как вычислительный аспект qMC-теории опирается преимущественно на понятие полной вариации:

**Теорема 8.3.2** (Коксма—Хлавка). *Пусть ограниченная вариация функции $f$ на $R = [0,1]^D$ равна $H(f)$, а последовательность $S$ такая, как в определении* 8.3.1, *тогда*

$$\left| \frac{1}{N} \sum_{\vec{x} \in S} f(\vec{x}) - \int_{\vec{r} \in R} f(\vec{r})\, d^D \vec{r} \right| \leqslant H(f) D_N^*(S).$$

*Более того, это неравенство точное в том смысле, что для любой $N$-точечной $S \subset R$ и любого $\varepsilon > 0$ найдется функция $f$ с $H(f) = 1$ такая, что левая часть неравенства не меньше, чем $D_N^*(S) - \varepsilon$.*

Этот замечательный результат непосредственно связывает величину погрешности приближенного многомерного интегрирования и $\star$-дискрепанс $D_N^*$. Теперь вопрос об эффективности qMC-испытаний может решаться в терминах дискрепансов. Кстати говоря, одним из многих заслуживающих упоминания результатов после теоремы 8.3.2 является полученная в работе [Wozniakowski 1991] оценка «средней» погрешности интегрирования в единичном кубе. Как показано Возняковским, «среднее по статистическому ансамблю» — в достаточно строгом смысле — погрешности интегрирования тесно связано с дискрепансом, окончательно подтверждая исключительную практическую важность понятия дискрепанса. Более того, ряд последних исследований, как мы увидим, ведет к пониманию того, почему реальные испытания по методу qMC зачастую приводят к лучшим результатам (более точным), чем то следует из оценок с помощью дискрепансов.

В целом, у qMC-последовательности $S$ значение $D^*$ должно быть мало, и именно при построении таких последовательностей начинает привлекаться теория чисел. Первое, что необходимо отметить, есть тонкое различие между дискрепансом множества точек и дискрепансом последовательности. Положим размерность $D = 1$ и рассмотрим множество точек

$$P = \left\{ \frac{1}{2N}, \frac{3}{2N}, \ldots, 1 - \frac{1}{2N} \right\}.$$

Его дискрепанс $D_N^*(P) = 1/(2N)$. С другой стороны, *не существует ни одной счетной бесконечной последовательности $S$*, удовлетворяющей условию

$D_N^*(S) = O(1/N)$. На самом деле, в работе [Schmidt 1972] доказано, что если $S$ — счетная бесконечная последовательность, тогда для бесконечно многих $N$ выполняется

$$D_N^*(S) \geqslant c\frac{\ln N}{N},$$

где $c$ — абсолютная константа (не зависящая от $N$ и $S$). В действительности можно положить $c = 3/50$ (см. [Niederreiter 1992]), но суть в том, что требование бесконечности qMC-последовательности позволяет построить сколь угодно много подобных примеров, что заставляет проявить особую бдительность. В приведенном выше примере множества $P$ с дискрепансом $1/(2N)$ нет противоречия, поскольку его элементы сами зависят от $N$.

## 8.3.2. Специальные qMC-последовательности

Мы готовы теперь перейти к построению последовательностей с малым $\star$-дискрепансом. Основная задача состоит в том, чтобы для заданного простого числа $p$ построить последовательность с малым дискрепансом, используя $p$-ичное представление целых чисел. Но прежде мы рассмотрим конструкцию в несколько более общем случае системы счисления с произвольным основанием $B$. Для пространств более чем одной размерности будет использоваться набор взаимно простых оснований.

**Определение 8.3.3.** Пусть $B \geqslant 2$ есть основание системы счисления. Последовательностью ван дер Корпута для основания $B$ называется последовательность

$$S_B = (\rho_B(n)), \quad n = 0, 1, 2, \ldots,$$

где $\rho_B$ — функция перехода к системе счисления с обратным основанием, которая определяется для неотрицательных $n$ следующим образом. Если в $B$-ичной системе счисления $n = \sum_i n_i B^i$, то

$$\rho_B(n) = \sum_i n_i B^{-i-1}.$$

Такие последовательности несложно представить и так же просто сгенерировать практически — проще, чем можно было бы ожидать. Рассмотрим пример последовательности ван дер Корпута для основания $B = 2$. Начиная с $n = 0$, выписываем в двоичной системе ряд целых чисел

$$n = 0, 01, 10, 11, 100, \ldots$$

и, обращая порядок битов, получаем (также в двоичной системе)

$$S = (0.0, 0.10, 0.01, 0.11, 0.001, \ldots).$$

В общем виде член последовательности $\rho_B(n) \in S$ с индексом

$$n = n_k n_{k-1} \ldots n_1 n_0$$

вычисляется путем изменения направления записи цифр:

$$\rho_B(n) = 0.n_0 n_1 \ldots n_k.$$

Известно, что любая последовательность ван дер Корпута имеет дискрепанс

$$D_N^*(S_B) = O\left(\frac{\ln N}{N}\right);$$

здесь символ $O$-большое заменяет константу, зависящую только от $B$. Оказывается, что эта константа принимает наименьшее значение при $B = 3$, но для приложений существенно то, что она, вообще говоря, растет с увеличением $B$ (см. [Faure 1981]).

Когда размерность $D > 1$, qMC-последовательности, аналогичные последовательностям ван дер Корпута, можно определить следующим образом:

**Определение 8.3.4.** Пусть $\bar{B} = \{B_1, B_2, \ldots, B_D\}$ — набор взаимно простых оснований, любое $B_i > 1$. Последовательностью Холтона по основанию $\bar{B}$ называется последовательность

$$S_{\bar{B}} = (\vec{x}_n), \quad n = 0, 1, 2, \ldots,$$

где

$$\vec{x}_n = (\rho_{B_1}(n), \ldots, \rho_{B_D}(n)).$$

Другими словами, последовательность Холтона подразумевает выбор различных оснований систем счисления для различных координат векторов. Соответствующие основания должны быть попарно взаимно просты. Поэтому для генерирования qMC-последовательности в трехмерном единичном кубе можно выбрать простые основания $\{B_1, B_2, B_3\} = \{2, 3, 5\}$, записывая при этом $n = 0, 1, 2, \ldots$ одновременно во всех трех системах счисления, в результате получается

$$\begin{aligned}
\vec{x}_0 &= (0,\ 0,\ 0), \\
\vec{x}_1 &= (1/2,\ 1/3,\ 1/5), \\
\vec{x}_2 &= (1/4,\ 2/3,\ 2/5), \\
\vec{x}_3 &= (3/4,\ 1/9,\ 3/5),
\end{aligned}$$

и т.д. Примечательна особеность размещения этих точек в единичном кубе. Фундаментальное качественное отличие qMC-последовательностей состоит в том, что каждая следующая точка попадает в область, «в которой еще не было других точек». Сравним с классическим методом Монте-Карло, в котором — так как испытания совершенно случайны — точки не только порой ложатся слишком кучно, но и образуют «пустоты», словно стремясь занять более предпочтительные области.

Последовательностями Холтона не исчерпывается множество qMC-последовательностей, как мы увидим в следующем разделе. А сейчас сформулируем теорему, описывающую типичный характер роста дискрепанса с увеличением размерности:

**Теорема 8.3.5** (дискрепанс Холтона). *Через $S_{\bar{B}}$ обозначим последовательность Холтона для системы оснований $\bar{B}$. Тогда ее $\star$-дискрепанс удовлетво-*

*ряет неравенству:*

$$D_N^*(S_{\bar{B}}) < \frac{D}{N} + \frac{1}{N}\prod_{i=1}^{D}\left(\frac{B_i - 1}{2\ln B_i}\ln N + \frac{B_i + 1}{2}\right).$$

Довольно сложное доказательство было предложено в работе [Niederreiter 1992]. Заметим, что из теоремы следует точная верхняя оценка соответствующей символу «$O$-большое» мультипликативной константы в выражении для ошибки интегрирования

$$D_N^*(S_{\bar{B}}) = O\left(\frac{\ln^D N}{N}\right),$$

упомянутом во вводной части раздела. Кроме того, мы можем наблюдать (нежелательный) эффект предположительного роста дискрепанса с увеличением основания (мы говорим, предположительного, так как это только лишь верхняя оценка). На самом деле этот эффект наблюдается на практике. Отметим, что известно $N$-точечное множество Хеммерсли, в котором вектор $\vec{x}_n$ имеет первую компоненту $x_0 = n/N$, а остальные компоненты $\vec{x}_n$ образуют $(D-1)$-мерный вектор Холтона. Это множество зависит от $N$, и потому не может порождать бесконечную последовательность. Однако для множеств Хеммерсли дискрепанс имеет несколько лучшую оценку

$$D_N^* = O\left(\frac{\ln^{D-1} N}{N}\right),$$

что служит иллюстрацией того, как зависимость множества от $N$ может приводить к уменьшению сложности вычислений.

### 8.3.3. Простые числа на Уолл-стрит?

Интересным упражнением является тестирование эффективной qMC-последовательности, скажем, при вычислении объема единичного $D$-шара. Последовательности Холтона показывают хорошие результаты для небольших размерностей $D$, примерно в пределах 10. Одно из преимуществ последовательности Холтона в том, что она легко просчитывается вперед, и поэтому несколько или более компьютеров могут одновременно производить выборку из непересекающихся частей последовательности. В следующем алгоритме показано, как вычисляются элементы последовательности, начиная с $n$-го члена. Что делает процедуру особенно эффективной, так это то, что запись индекса в рассматриваемых системах счисления постоянно корректируется при переходе от одного индекса к другому.

**Алгоритм 8.3.6** (быстрое генерирование qMC-последовательности). Алгоритм генерирует векторы $D$-мерной последовательности Холтона. Через $p_1, \ldots, p_D$ обозначим первые $D$ простых чисел. Процедура *seed*() для заданного стартового индекса $n$ возвращает вектор $\vec{x}_n$, компоненты которого естественно

обозначить через $\vec{x}_n[1], \ldots, \vec{x}_n[D]$. Затем, вызывая функцию $random()$, последовательно получаем $\vec{x}_{n+1}, \vec{x}_{n+2}, \ldots$, при этом мы предполагаем, что ни один из индексов не превосходит $N$. Для удобства также вычисляются представления стартового индекса $n$, записываемые обобщенными цифрами $(d_{i,j})$. Затем, по мере исполнения функции $random()$, они последовательно увеличиваются, как в курвиметре, образуя представления для индексов, больших $n$.

1. [Процедура *seed*]
   *seed*$(n)$ {  // $n$ — выбранный стартовый индекс.
   for$(1 \leqslant i \leqslant D)$ {
   $K_i = \left\lceil \frac{\ln(N+1)}{\ln p_i} \right\rceil$;  // Параметр, отвечающий за точность.
   $q_{i,0} = 1$;
   $k = n$;
   $x[i] = 0$;  // $\vec{x}$ обозначает $\vec{x}_n$.
   for$(1 \leqslant j \leqslant K_i)$ {
   $q_{i,j} = q_{i,j-1}/p_i$;  // $q_{i,j} = p_i^{-j}$.
   $d_{i,j} = k \bmod p_i$;  // Первоначально $d_{i,j}$ являются цифрами $p_i$-ичного представления числа $n$.
   $k = (k - d_{i,j})/p_i$;
   $x[i] = x[i] + d_{i,j} q_{i,j}$;
   }
   return;  // $\vec{x}_n$ имеет координаты $(x[1], \ldots, x[D])$.
   }
2. [Функция *random*]
   *random*$()$ {
   for$(1 \leqslant i \leqslant D)$ {
   for$(1 \leqslant j \leqslant K_i)$ {
   $d_{i,j} = d_{i,j} + 1$;  // Приращение индекса.
   $x[i] = x[i] + q_{i,j}$;
   if$(d_{i,j} < p_i)$ break;  // Выйти из цикла после вычисления всех переносов.
   $d_{i,j} = 0$;
   $x[i] = x[i] - q_{i,j-1}$;
   }
   }
   return $(x[1], \ldots, x[D])$;  // Новый вектор $\vec{x}$.
   }

Легко убедиться, что этот алгоритм функционирует по принципу «курвиметра» и преобразует запись индекса в $p_m$-ичной системе счисления в соответствии с определением 8.3.4. Обратим внимание на то, что параметр $K_i$ имеет смысл максимально возможного для индекса $j$ числа знаков в представлении по основанию $p_i$. Для определения $K_i$ должно быть задано некоторое число $N$, ограничивающее сверху диапазон значений индексов. Такое или эквивалентное требование необходимо, чтобы ограничить точность операций в системе счисления с обратным основанием.

В алгоритме 8.3.6 обычно используется формат чисел с плавающей точкой, т.е. хранятся сами степени $q_{i,j}$, а не целочисленные показатели $n_{i,j}$. Хотя в принципе нет ничего плохого в том, чтобы конкретный генератор использовал вместо $q_{i,j}$ показатели степеней. Более того, целочисленный формат позволяет тестировать алгоритм следующим интересным способом. Пусть, к примеру, $N = 1000$, так что можно вычислять $\vec{x}_0, \ldots, \vec{x}_{999}$; выберем размерность $D = 2$, что означает выбор чисел 2,3 в качестве оснований. Затем выполним процедуру $seed(701)$, в результате получим

$$\vec{x}_{701} = (757/1024, 719/729).$$

Далее 9 раз подряд вызываем функцию $random()$, в результате имеем

$$\vec{x}_{710} = (397/1024, 674/729).$$

И конечно, мы можем проверить правильность работы алгоритма, вернувшись в начало и вызвав $seed(710)$, и убедиться в том, что стартуя с индекса $701 + 9$, мы получим в точности вектор $\vec{x}_{710}$, указанный выше.

Интересно, что алгоритм 8.3.6 на самом деле быстрый, хотя бы в следующем смысле. Практически он способен работать быстрее, даже чем встроенные системные генераторы случайных чисел. Благодаря этому преимуществу, алгоритм приобретает значение даже вне связи с задачей численного интегрирования. Когда требуется получить случайное число, равномерно распределенное в $[0, 1)$, естественно обратиться к встроенной системной случайной функции, особенно если свойственная случайным выборкам тенденция образовывать скопления и пустоты не вызывает беспокойства. Но в статистике часто возникает потребность иметь нерегулярное «покрытие» интервала $[0, 1)$, можно сказать, «справедливое» покрытие, когда не пропущен никакой подынтервал. И это тот самый случай, когда следует обратиться к qMC-алгоритму.

Теперь мы можем без труда взять многомерный интеграл с помощью процедур алгоритма 8.3.6:

**Алгоритм 8.3.7** (многомерное qMC-интегрирование). Пусть заданы размерность $D$ и интегрируемая функция $f : R \to R$, где $R = [0, 1]^D$. Данный алгоритм вычисляет многомерный интеграл

$$I = \int_{\vec{x} \in R} f(\vec{x})\, d^D\vec{x},$$

генерируя $N_0$ векторов qMC-последовательности $(\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_n, \ldots, \vec{x}_{n+N_0-1}, \ldots)$, начиная с $n$-го. Для инициализации алгоритма 8.3.6 предполагается известной константа $N \geqslant n + N_0$, ограничивающая индексы.

1. [Процедура инициализации — как в алгоритме 8.3.6]
   $seed(n)$; // Обобщенному qMC-вектору $\vec{x}$ присваивается значение $\vec{x}_n$.
   $I = 0$;
2. [qMC-интегрирование]
   // Функция $random()$ обновляет qMC-вектор (см. алгоритм 8.3.6).
   for($0 \leqslant j < N_0$) $I = I + f(random())$;
   return $I/N_0$; // Приближенное значение интеграла.

Рассмотрим пример работы данного алгоритма. Вычислим объем единичного $D$-шара, т. е. шара радиуса 1. Для этого выразим подынтегральную функцию $f$ через функцию Хевисайда $\theta$ (которая равна 1, если аргумент положителен, 0 — если отрицателен, и $1/2$ — в 0),

$$f(\vec{x}) = \theta(1/4 - (\vec{x} - \vec{y}) \cdot (\vec{x} - \vec{y})),$$

где $\vec{y} = (1/2, 1/2, \ldots, 1/2)$, так что $f$ равна нулю всюду за пределами шара радиуса $1/2$. (Это максимальный шар, который можно поместить в куб $R$.) Таким образом, за оценку объема $D$-мерного шара следует принять $2^D I$, где $I$ есть результат работы алгоритма 8.3.7 для заданной характеристической функции сферы $f$.

Мы уже упоминали, насколько поразительное впечатление производит qMC-алгоритм такого типа в сравнении с классическими испытаниями по методу Монте-Карло. Замечательной особенностью концепции qMC является то, что вычисления легко распараллеливаются. Алгоритм 8.3.7 может выполняться, скажем, $M$ машинами, стартующими с разных индексов. Идеальный случай — когда в итоге реализуется некоторая непрерывная последовательность из $NM$ векторов. Естественно, все зависит от вида процедуры инициализации $seed()$. Более точно, чтобы вычислить интеграл на миллиарде точек, каждая из 100 машин реализует вышеописанный алгоритм с $N = 10^7$, за той лишь разницей, что 0-я машина должна стартовать с $n = 0$ (т. е. с вызова процедуры $seed(0)$), следующая — с $n = 1$, и т.д. вплоть до машины с номером 99, которая начинает с $n = 99$. В качестве ответа следует взять среднее по 100 испытаниям.

Ниже приводится типичный пример численного сравнения. Вычисляется число $\pi$ с помощью qMC-методов и сравнивается с результатом стандартного метода Монте-Карло. Напомним, что объем единичного $D$-шара в точности равен

$$V_D = \frac{\pi^{D/2}}{\Gamma(1 + D/2)}.$$

Через $V_D(N)$ обозначим объем шара, вычисляемый на $N$ сгенерированных векторах, а через $\pi_N$ обозначим «экспериментальное» значение числа $\pi$, которое выражается из формулы объема через $V_D$. Мы здесь преследуем две цели: показать типичную скорость сходимости алгоритма и дать представление о присущем методу параллелизме. Результат работы алгоритма 8.3.7 на примере измерения объема 3-мерного шара с простыми основаниями $p = 2, 3, 5$ приведен в табл. 8.1.

В левом столбце указано количество «случайных» точек, «брошенных» в единичный $D$-куб, в другом столбце — соответствующее *суммарное* приближение числа $\pi$. Мы говорим, «суммарное», поскольку каждая последовательность из $10^6$ испытаний может проводиться на отдельной машине. Тогда ответ в правом столбце получается путем комбинирования результатов работы машин вплоть до $N$-й включительно. Например, $\pi_5$ может быть вычислено с помощью одной серии из $5 \cdot 10^6$ испытаний, или, что эквивалентно, на 5 различных машинах, каждая использует по $10^6$ точек. В последнем случае процедура $seed(n)$ должна быть выполнена для 5 различных значений аргумента, в за-

| $N/10^6$ | $\pi_N$ |
|---|---|
| 1 | 3.14158 |
| 2 | 3.14154 |
| 3 | 3.14157 |
| 4 | 3.14157 |
| 5 | 3.14158 |
| 6 | 3.14158 |
| 7 | 3.14158 |
| 8 | 3.141590 |
| 9 | 3.14158 |
| 10 | 3.1415929 |

**Таблица 8.1.** Приближения числа $\pi$ получены с помощью qMC-последовательности Холтона с параметрами $p = 2, 3, 5$. Объем единичного 3-мерного шара вычисляется с различными длинами выборок из qMC-последовательностей, от $N = 10^6$ до $N = 10^7$. Результаты выписаны до первого неправильного десятичного знака.

висимости от конкретной машины. Как полученные результаты соотносятся с методом Монте-Карло? Рассуждая грубо, мы можем ожидать, что в нижней строке ($N = 10^7$) аналогичной таблицы для метода Монте-Карло ошибка будет приблизительно в третьем знаке после запятой (в силу того, что в данном случае $\log_{10}\sqrt{N}$ примерно равно 3.5). Такое преимущество qMC-методов над классическими — выражаемое несколькими порядками величины — является типичным для небольших размерностей и количества точек порядка нескольких миллионов.

А теперь к вопросу об Уолл-стрит[3], под чем подразумевается такой феномен, как вычислительная финансовая теория. Читатель, которому вопрос интегрирования в пространствах очень больших размерностей $D$ кажется неактуальным, может избавиться от своего скептического отношения, познакомившись с родом вычислений, предпринимавшихся в связи с теорией управления рисками и другими вычислительными аспектами финансовой теории. Так, применительно к финансовой теории, Папагеоргиу и Траубом в 1997 г. изучались 25-мерные интегралы вида

$$I = \int \dots \int_{\vec{x}\in R} \cos|\vec{x}|\; e^{-\vec{x}\,\vec{x}}\, d^D\vec{x}.$$

Авторы пришли к довольно неожиданному выводу, что qMC-методы (в данном случае использовались последовательности Форе) могут превосходить стандартный метод Монте-Карло, невзирая на асимптотику $O((\ln^D N)/N)$, которая с практической точки зрения не смотрится выигрышно в сравнении с $O(1/\sqrt{N})$, когда $D = 25$. В других работах (например, в работе [Paskov and

[3] Уолл-стрит (Wall Street) — улица в Нью-Йорке, где находится Нью-Йорская фондовая биржа. — *Прим. перев.*

Traub 1995]) исследовались интегралы такой размерности как $D = 360$. Как отмечают авторы, рассматриваемые интегралы (имеющие отношение, говоря языком экономистов, к «облигациям, обеспеченным пулом ипотек» (CMO)) удобны для тестирования, поскольку подынтегральная функция имеет известную вычислительную сложность, что — по словам авторов — «позволяет ограничиться возможно меньшим количеством испытаний». Как отмечалось в работе [Boyle et al. 1995], а также и многими другими исследователями, будет qMC-метод превосходить стандартный метод Монте-Карло при интегрировании в пространстве большой размерности $D$ или нет, в значительной степени зависит от конкретной решаемой задачи. В целом, среди специалистов по численным методам, не имеющих непосредственного отношения к финансовому миру, просматривается тенденция к работе с интегралами, которые представляют все большую проблему для применения qMC-методов. То есть, «финансовые» подынтегральные функции на практике оказываются более «гладкими».

Не менее интересной, чем qMC-теория сама по себе, является непрекращающаяся дискуссия в литературе по qMC-методам. Некоторые авторы считают, что последовательность Холтона (которую мы подробно рассматривали как пример qMC-последовательности, опирающейся на теорию простых чисел) уступает в эффективности последовательностям Соболя (см. [Bratley and Fox 1988]) или Форе ([Niederreiter 1992]). Как мы уже отмечали выше, такая оценка, по видимости, сильно зависит от сферы приложения. Однако подобные обвинения имеют под собой теоретическое основание, а именно, теорему ([Faure 1982]) о $\star$-дискрепансе последовательности Форе, которая гласит:

$$D_N^* \leqslant \frac{1}{D!}\left(\frac{p-1}{2\ln p}\right)^D \frac{\ln^D N}{N},$$

где $p$ — минимальное простое число, не меньшее чем $D$. Хотя $D$-мерную последовательность Холтона можно построить, используя $D$ первых простых чисел, а оценка Форе зависит от следующего простого числа, все же оценка теоремы 8.3.5 существенно проигрывает. Но вероятно, оценки обеих теорем не являются теоретически наилучшими. Как бы то ни было, простые числа в очередной раз появляются в теории дискрепанса и ее qMC-приложениях.

Как отмечалось в публикациях, погрешность qMC-метода $O\left((\ln^D N)/N\right)$ для достаточно больших $D$, достаточно малых $N$ и ряда комбинаций $D$ и $N$ хуже, чем у стандартного метода Монте-Карло, $O\left(1/\sqrt{N}\right)$. По этой причине некоторые исследователи советуют применять qMC-методы с осторожностью. Одна из проблем в том, что многочисленные теоремы, такие как теорема 8.3.5 и приведенная выше оценка Форе не дают информации о реальном поведении констант в символе «$O$-большое». По этому вопросу в недавнее время достигнуты некоторые продвижения. Одно из них состоит в открытии так называемых «разреженных» последовательностей Холтона. Данный метод позволяет устранить нежелательную зависимость между координатами $D$-мерных векторов последовательности Холтона. Это достигается в два приема. Во-первых, осуществляется перестановка цифр в записи числа в системе

счисления с обратным основанием. И затем, пусть простые основания обозначены через $p_0, \ldots, p_{D-1}$, выбирается отличное от них простое число $p_D$, и используется только каждый $p_D$-й вектор обычной последовательности Холтона. В работе [Kocis and Whiten 1997] утверждается, что решающим образом повышается эффективность последовательностей Холтона для больших размерностей $D$, скажем, от 40 до 400. Любопытно, что авторы обнаружили исключительно удачный вариант выбора дополнительного простого $p_D = 409$, чему пока не найдено объяснения. Другая идея, предложенная в работе [Crandall 1999], состоит в использовании небольшого подмножества простых чисел, даже при больших $D$, и соответствующей последовательности Холтона малой размерности как векторного параметра для заполняющей $D$-мерное пространство кривой. Так как оценка теоремы 8.3.5 жестко привязана к величине основания, есть повод надеяться, что данный метод, использующий небольшие простые основания, дает определенное статистическое преимущество в многомерных пространствах.

В то время как понятие дискрепанса возникло сравнительно давно, постоянно возникают все новые идеи относительно генерирования qMC-множеств. Один многообещающий подход, связанный с так называемыми $(t, m, s)$-сетями, развит в работах [Owen 1995, 1997a, 1997b], [Tezuka 1995], [Veach 1997]. Под сетями здесь понимаются множества точек, обладающие свойством «минимального заполнения». Например, множество из $N = b^m$ точек в $s$-мерном пространстве называется $(t, m, s)$-сетью, если любой допустимый параллелепипед объемом $b^{t-m}$ содержит в точности $b^t$ точек. Еще одна любопытная связь между простыми числами и дискрепансом проводится в работе [Joe 1999] (см. также ссылки в ней). В данной «теоретико-числовой» концепции используются приближения вида

$$\int_{[0,1]^D} f(\vec{x})\, d^D\vec{x} \;\approx\; \frac{1}{p}\sum_{j=0}^{p-1} f\left(\left\{\frac{j\vec{K}}{p}\right\}\right).$$

Здесь $\{\vec{y}\}$ обозначает вектор, состоящий из дробных частей координат вектора $\vec{y}$, а $\vec{K}$ — некоторый постоянный вектор, все компоненты которого взаимно просты с $p$. На самом деле вместо простого $p$ можно использовать составные числа, однако вычисление так называемого $L_2$-дискрепанса и связанной с ним ожидаемой погрешности интегрирования особенно просто, когда $p$ — простое. Мы упомянули об этих двух методах, чтобы подчеркнуть, что qMC-теория находится в состоянии непрерывного развития. И кто знает, когда и как теория чисел, в частности, теория простых чисел, будет применяться в qMC-методах будущего?

В завершение этого раздела мы отметим новый результат, который может объяснить, почему qMC-испытания в некоторых случаях так эффективны. В работе [Sloan and Wozniakowski 1998] отмечается, что погрешность для некоторых последовательностей (как, например, у 360-мерной qMC-последовательности Трауба из финансовых приложений) ведет себя асимптотически как $O(1/N)$, т. е. независимо от размерности $D$. На самом деле авто-

рами доказано, что существуют классы интегрируемых функций, для которых подходящие последовательности с малым дискрепансом имеют абсолютную погрешность интегрирования $O(1/N^{\rho})$ для некоторого действительного $\rho \in [1,2]$.

## 8.4. Диофантов анализ

Здесь пойдет речь о диофантовом анализе, который, грубо говоря, занимается поиском целочисленных решений различных уравнений. Нами уже упоминалась последняя теорема Ферма (FLT) о решениях уравнения

$$x^p + y^p = z^p$$

и говорилось о том, как одни лишь попытки решить ее численно подняли нижнюю грань для $p$ до нескольких миллионов (разд. 1.3.3, упр. 9.68). Это замечательная вычислительная задача — не имея в виду, конечно, изумительного доказательства FLT Уайлса — однако есть множество других подобных исследований. В таких приключениях часто образуется здоровое соединение теории и вычислений.

Например, для уравнения Каталана

$$x^p - y^q = 1,$$

где $p, q$ — простые, а $x, y$ — натуральные числа, известно *только одно* решение, тривиальное, но изящное,

$$3^2 - 2^3 = 1.$$

Обратим внимание на то, что задачу поиска диофантовых решений мы здесь просто сводим к вопросу существования еще больших пар следующих друг за другом степеней натуральных чисел. Неплохой исторический обзор проблемы Каталана приводится в работе [Ribenboim 1994], а в статье [Mignotte 2000] рассказывается о последних исследованиях. Используя теорию линейных форм логарифмов алгебраических чисел, Тайдеман в 1976 г. показал, что уравнение Каталана имеет конечное число решений. Конкретнее,

$$y^q < e^{e^{e^{e^{730}}}},$$

(см. [Guy 1994]). Таким образом, полное решение проблемы Каталана сводится к (гигантским!) расчетам. Вскоре после появления выдающейся теоремы Тайдемана, Ланжевен показал, что для любого решения показатели $p, q < 10^{110}$. За прошедшие годы этот предел для показателей продолжил движение вниз, и одни результаты вытеснялись другими. Например, на момент выхода в свет первого издания этой книги было известно, что $\min\{p,q\} > 10^7$ и $\max\{p,q\} < 7.78 \times 10^{16}$. Кроме того, были известны конкретные легко проверяемые условия на допустимые пары показателей, например, двойное условие Вифериха, принадлежащее Михайлеску: если $p, q$ — степени Каталана, отличные от пары 2, 3, то

$$p^{q-1} \equiv 1 \pmod{q^2} \quad \text{и} \quad q^{p-1} \equiv 1 \pmod{p^2}.$$

Можно было надеяться, что эти продвижения наряду с достаточно серьезными вычислениями окончательно решат проблему Каталана. Эту проблему действительно удалось решить окончательно, но с помощью гораздо более хитроумных средств, чем простое вычисление.

В работе [Mihăilescu 2004] представлено окончательное решение проблемы Каталана, и действительно, числа 8 и 9 являются единственной парой соседних нетривиальных степеней натуральных чисел. Интересно отметить, что нам до сих пор не известно, существует ли бесконечно много пар соседних степеней, разность между которыми равна 2 или любому другому фиксированному числу, большему 1, но предполагается, что это множество конечно. В этой связи см. упр. 8.20.

Следующее диофантово уравнение связано как с последней теоремой Ферма, так и с проблемой Каталана

$$x^p + y^q = z^r, \tag{8.2}$$

где $x, y, z$ — натуральные взаимно простые числа, а показатели степеней $p, q, r$ — простые числа с условием $1/p + 1/q + 1/r \leqslant 1$. Гипотеза Ферма—Каталана утверждает, что существует не более чем конечное число решений $x^p, y^q, z^r$ уравнения (8.2). Ниже приведены все известные решения:

$$\begin{aligned}
1^p + 2^3 &= 3^2 \quad (p \geqslant 7)\\
2^5 + 7^2 &= 3^4\\
13^2 + 7^3 &= 2^9\\
2^7 + 17^3 &= 71^2\\
3^5 + 11^4 &= 122^2\\
33^8 + 1549034^2 &= 15613^3\\
1414^3 + 2213459^2 &= 65^7\\
9262^3 + 15312283^2 &= 113^7\\
17^7 + 76271^3 &= 21063928^2\\
43^8 + 96222^3 &= 30042907^2.
\end{aligned}$$

(Последние пять решений были найдены Бейкерсом и Цагиром.) Гипотеза Тайдемана и Цагира состоит в том, что уравнение (8.2) вовсе не имеет решений, если $p, q, r \geqslant 3$. За доказательство этой гипотезы назначена денежная премия (премия Била), см. работы [Bruin 2003] и [Mauldin 2000]. Из работы [Darmon and Granville 1995] известно, что если $p, q, r$ *фиксированы* и выполняется условие $1/p + 1/q + 1/r \leqslant 1$, то уравнение (8.2) имеет не более чем конечное число решений $x, y, z$. Нам также известно, что при некоторых значениях $p, q, r$ единственными решениями являются те, которые присутствуют в нашей небольшой таблице. В частности, все возможные тройки с показателями $\{2, 3, 7\}$, $\{2, 3, 8\}$, $\{2, 3, 9\}$ и $\{2, 4, 5\}$ присутствуют в вышеприведенном списке. Кроме того, существуют и многие другие тройки показателей, для которых доказано, что нетривиальных решений не существует. Эти результаты связаны с именами многих людей, среди которых Беннетт, Бейкерс, Брёйн, Дармон, Элленберг,

Краус, Мерель, Поонен, Шефер, Скиннер, Столл, Тейлор и Уайлс. Информацию о некоторых недавно вышедших статьях, из которых можно извлечь и другие имена, см. в работах [Bruin 2003] и [Beukers 2004].

Гипотеза Ферма—Каталана является частным случаем известной ABC-гипотезы. Через $\gamma(n)$ обозначим наибольший свободный от квадратов делитель числа $n$. Тогда, согласно ABC-гипотезе, для любого $\varepsilon > 0$ существует не более чем конечное число троек взаимно простых натуральных чисел $a, b, c$ таких, что

$$a + b = c, \quad c < \gamma(abc)^{1-\varepsilon}.$$

Современный обзор ABC-гипотезы, включающий множество удивительных следствий, можно найти в статье [Granville and Tucker 2002].

Хотя как правило, диофантовы уравнения решаются исключительно сложно, во многих случаях, прибегая к таким инстументам как квадратичная взаимность, удается сузить диапазон возможных решений. Например, можно доказать, что уравнение

$$y^2 = x^3 + k \tag{8.3}$$

не имеет целочисленных решений, если $k = (4n-1)^3 - 4m^2$, $m \neq 0$, и никакой простой делитель $m$ не равен 3 (mod 4) (см. упр. 8.13).

Помимо занимательных примеров решений отдельных уравнений, разработана фундаментальная общая теория диофантовых уравнений. История этого проекта, растянувшегося на десятилетия, завораживает. Центральная проблема, сформулированная на рубеже столетия[4], известная как «10-я проблема Гильберта», ставит вопрос о существовании *универсального* алгоритма для решения произвольного диофантова уравнения. За все время попыток решения этой проблемы в центре внимания находилось понятие диофантова множества. Диофантовым множеством $S$ называется множество натуральных чисел, для которого существует многочлен нескольких переменных $P(X, Y_1, \ldots, Y_l)$ с коэффициентами из $\mathbf{Z}$ такой, что $x \in S$ тогда и только тогда, когда уравнение $P(x, y_1, \ldots, y_l) = 0$ имеет натуральное решение $y_j$. Несложно доказывается теорема Путнама 1960 г. (см. [Ribenboim 1996, p. 189]), которая утверждает, что множество $S$ натуральных чисел — диофантово тогда и только тогда, когда существует многочлен нескольких переменных $Q$ с целыми коэффициентами, множество положительных значений которого на натуральных аргументах есть в точности множество $S$.

Используя это определение диофантова множества, группа математиков-теоретиков во главе с Путнамом, Дэвисом, Робинсоном и Матиясевичем получила следующий поразительный результат: *множество простых чисел — диофантово*. А именно, они показали, что существует многочлен $P$ от некоторого количества переменных с целыми коэффициентами такой, что когда его переменные принимают всевозможные натуральные значения, множество положительных значений многочлена $P$ является *в точности* множеством простых чисел.

[4] Имеется в виду двадцатое столетие. — *Прим. перев.*

Один такой многочлен был найден в явном виде Джонсом, Сато, Вада и Винсом в 1976 г. (см. [Ribenboim 1996]). Он таков:

$$
\begin{aligned}
&(k+2)\Big(1-(wz+h+j-q)^2-((gk+2g+k+1)(h+j)+h-z)^2\\
&-(2n+p+q+z-e)^2-\big(16(k+1)^3(k+2)(n+1)^2+1-f^2\big)^2\\
&-\big(e^3(e+2)(a+1)^2+1-o^2\big)^2-\big(a^2y^2-y^2+1-x^2\big)^2\\
&-\big(16r^2y^4(a^2-1)+1-u^2\big)^2\\
&-\big(((a+u^4-u^2a)^2-1)(n+4dy)^2+1-(x+cu^2)\big)^2\\
&-(n+l+v-y)^2-\big(a^2l^2-l^2+1-m^2\big)^2-(ai+k+1-l-i)^2\\
&-(p+l(a-n-1)+b(2an+2a-n^2-2n-2)-m)^2\\
&-(q+y(a-p-1)+s(2ap+2a-p^2-2p-2)-x)^2\\
&-(z+pl(a-p)+t(2ap-p^2-1)-pm)^2\Big).
\end{aligned}
$$

Многочлен имеет степень 25 и, удобным образом, зависит от 26 переменных, что позволяет задействовать все буквы латинского алфавита! Замечательно, что любое простое число $p$, вычисляемое на основе подобного многочлена, может быть получено вместе с доказательством простоты, которое требует только $O(1)$ *арифметических* операций. Для этого нужно присвоить 26 переменным значения так, чтобы значение многочлена было равно $p$. Однако такая проверка может потребовать огромное количество *битовых* операций.

«10-я проблема Гильберта» в итоге была решена. Ответ состоит в том, что искомого алгоритма не существует. Последнюю точку поставил Матиясевич, доказав, что любое перечислимое множество — диофантово. Но до тех пор, в течение более чем полувека, главным объектом исследований было множество простых чисел (см. [Matijasevič 1971], [Davis 1973]).

Диофантов анализ, от которого исторически произошла вся теория чисел, сегодня — все еще занимательная энергично развивающаяся область, привлекающая ученых-математиков и любителей математических головоломок. С основной частью теории можно ознакомиться, воспользовавшись ресурсами Интернета (см., например, работу [Weisstein 2000]).

## 8.5. Квантовые вычисления

Было бы уместно часть раздела, посвященного приложениям, отвести под то, что может стать ведущим способом вычислений в XXI веке. Речь идет о квантовых вычислениях, которые должны стать достойной сменой компьютерным процессам в обычном смысле, которым они до сих пор наделялись. Первое, что важно уяснить, это принципиальное отличие классической машины Тьюринга (TM) от квантовой машины Тьюринга (QTM). Предшествующая модель TM лежит в основе любого массового современного компьютера, за исключением, быть может, очень немногочисленных пробных экспериментальных QTM, по-

ка в стадии микроскопических опытов на элементарных частицах и т. п. (Хотя кто-нибудь может возразить, что природа уже миллиарды лет управляется глобальной QTM.) Главной особенностью TM является то, что она работает «последовательно», следуя набору инструкций (программе) предопределенным образом. (Есть такое понятие как вероятностная машина Тьюринга, имеющая статистический характер поведения, но чтобы не перегружать данный обзор, мы не будем рассматривать эту концепцию.) Напротив, в QTM должен быть реализован присущий реальному миру «параллелизм» с тем, чтобы вознести эффективность вычислений на поистине небывалую высоту. Имеется в виду, конечно, параллелизм как принцип протекания природных процессов с точки зрения законов квантовой механики. Эти законы опираются на совсем не очевидные представления. Как известно из квантовой теории, явления в микромире происходят совсем не так, как в макромире. Вспомним о корпускулярно-волновом дуализме (является электрон волной или частицей, или и тем, и другим?), об определении радиуса, вероятности, явлении интерференции — не только волн, но и совершенно материальных частиц — и т.д. В следующем разделе мы даем самое общее представление об основных понятиях квантовых вычислений с целью отобразить некоторые качественные особенности этой новой науки.

### 8.5.1. Квантовые машины Тьюринга (QTM) и интуиция

QTM пока находятся в самой начальной экспериментальной стадии и до сих пор не привлекались к решению ни одной «полезной» задачи. И потому будет уместно охарактеризовать в общих чертах, преимущественно пользуясь аналогиями, что должна представлять собой QTM. Обратимся к голографии — науке, с помощью которой получают отображение монолитного трехмерного тела на плоскую «голограмму». В действительности, при этом «вычисляется» 3-мерное преобразование Фурье, локальными флуктуациями мощности в котором определяется, что именно появляется на голограмме. Поскольку скорость света составляет приблизительно один фут в наносекунду ($10^{-9}$ секунды), будет справедливо заметить, что когда лазерный луч падает на какой-то предмет (скажем, на шахматную фигурку), и отраженные лучи смешиваются с опорным пучком, в результате чего образуется голограмма, «за считанные наносекунды осуществляется БПФ огромного порядка». В качественном, пусть и нетрадиционном смысле, стандартный $O(N \ln N)$ алгоритм (где $N$ должно обозначать достаточно большое число дискретно расположенных в пространстве точек, скажем, если строится высокоточная голограмма) превращается просто в алгоритм сложности $O(1)$. Хотя мы и допускаем некоторую вольность, используя нашу символику «$O$-большое» в данном контексте, мы всего-лишь хотели показать, что в модели интерференции световых волн, которая лежит в основе голографии, присутствует параллелизм. Интенсивность света на поверхности пленки голограммы в итоге зависит от *каждой* точки шахматной фигурки. Таков голографический или, можно сказать, «параллельный» аспект. И предсказываемый принцип работы QTM напоминает об этом эффекте.

Мы не утверждаем, что лабораторная голограмма устроена по принципу QTM, так как некоторые детали в приведенной упрощенной схеме были опущены. Что касается современной QTM-теории, то во-первых, она, помимо принципа квантовой интерференции, содержит еще два важных элемента, а именно, вероятностный характер поведения и теоретическую базу, использующую операторы, в частности, унитарные. Во-вторых, для нас представляет интерес практическая модель QTM, ориентированная не только на оптические эксперименты, но также на некоторые из сложнейших задач, с которыми столкнулись стандартные машины Тьюринга — например, факторизацию больших целых чисел. Было предложено много новых идей, однако первым, кто ввел понятие QTM в обиход, был замечательный ученый Фейнман. Он обнаружил, что квантово-механическая модель на условной машине Тьюринга имеет тенденцию работать в экспоненциальное число раз быстрее. В работах [Feynman 1982, 1985] была даже построена точная модель QTM на базе элементарных квантовых регистров. Формальное определение, более или менее соответствующее современным теоретическим представлениям, впервые было предложено в работах [Deutsch 1982, 1985]. Прекрасный обзор — занимающий золотую середину между популярным и математическим — содержится в работе [Williams and Clearwater 1998]. Тем, кто интересуется технической стороной физики, а также некоторыми сопутствующими теоретико-числовыми аспектами, можно посоветовать работу [Ekert and Jozsa 1996]. Совсем популярное изложение теории квантовых вычислений можно найти в работе [Hey 1999], а материалы для лекционного курса — в работе [Preskill 1999][5].

Придадим еще больший «квантовый» колорит идее естественного вычисления БПФ посредством лазерного излучения. В квантовой теории рассматривается такая идеальная система как квантовый осциллятор. Рассмотрим функцию потенциала $V(x) = x^2$. Уравнение Шрёдингера показывает, как волновой пакет $\psi(x,t)$, где $t$ — это время, движется в потенциальном поле. В классической механике аналогом является система с массой на невесомой пружине, совершающая равномерные колебания с периодом, скажем, $\tau$. Колебания есть и у квантовой модели, однако обнаруживается поразительный эффект: за четверть классического периода $\tau$ первоначальный волновой пакет *сворачивается в свое преобразование Фурье*. Это означает, что отправив как-то в QTM данные в виде начальной функции $\psi(x,0)$, то затем, прочитав $\psi(x,\tau/4)$, мы получим БПФ. (Кстати говоря, эта же идея лежит в основе дискуссии вокруг представления дзета-функции Римана (8.5).) Таким образом, в динамике элементарных частиц существуют процессы, аналогичные лазерной голографии. Заметим также, что волновая функция $\psi$ имеет смысл комплекснозначной амплитуды, а $|\psi|^2$ является плотностью вероятности. Это служит иллюстрацией участия статистического аспекта квантовой теории в физической картине.

Чтобы двигаться дальше и подготовиться к оставшейся части раздела, мы сейчас коснемся еще нескольких положений QTM-теории. С самого начала важно отметить, тем более если речь идет о теоретико-числовых алгоритмах,

[5]См. также добавленную при переводе книгу [Нильсон и Чанг 2006]. — *Прим. ред.*

что в QTM для хранения экспоненциального числа данных достаточно «полиномиального» объема памяти. Например, мы можем некоторым образом в так называемом квантовом регистре хранить *все* целые числа $a \in [0, q-1]$, используя только $\lg q$ кубит[6]. На первый взгляд, это кажется невероятным, однако мы уже предупреждали о том, что квантовый мир в известной степени идет вразрез с интуитивными представлениями. Здесь приходит на помощь мысленный эксперимент. Пусть $q = 2^d$, и требуется построить квантовый регистр из $d$ кубитов. Рассмотрим цепочку из $d$ отдельных молекул аммиака, в химии обозначаемых как $NH_3$. Молекулу можно представить в виде тетраэдра с тремя атомами водорода и атомом азота в вершинах. Про атом N будем говорить, что он находится либо «сверху», либо «снизу» (1 или 0) от трех атомов H. Таким образом, любое $d$-разрядное двоичное число может быть задано совокупной ориентацией молекул. Но как реализовать все возможные двоичные строки длины $d$? Ответ лежит на поверхности, стоит лишь вспомнить о замечательном квантовом принципе: молекула аммиака может одновременно находиться в *обоих* состояниях $1, 0$. Можно сказать, что состояния с минимальной энергией — назовем их «нулевыми» состояниями — обладают симметрией, если симметрия есть в геометрическом расположении. Когда «сосуд» с аммиаком находится в нулевом состоянии, каждая молекула некоторым образом находится «наполовину» в каждом из состояний $0, 1$. Формально, нулевое состояние кубита аммиака (в данной модели, молекулы) описывается как

$$\phi = \frac{1}{\sqrt{2}} ( \mid 0 \rangle + \mid 1 \rangle ),$$

где обозначение $\mid \rangle$ является стандартным (см. в указанных выше источниках по квантовой теории). Это обозначение служит напоминанием о том, что состояние принимает значения из абстрактного гильбертова пространства. Для того чтобы работать с ним как с измеримым числом, нужно брать скалярное произведение. Например, для приведенного нулевого состояния $\phi$ вероятность того, что мы *обнаружим* молекулу в состоянии $\mid 0 \rangle$ вычисляется как скалярный квадрат

$$| \langle 0 \mid \phi \rangle |^2 = \left| \frac{1}{\sqrt{2}} \langle 0 \mid 0 \rangle \right|^2 = \frac{1}{2},$$

т. е. с 50-процентной вероятностью атом азота будет обнаружен «снизу». Вернемся к нашему квантовому регистру из $d$ кубит (молекул). Если все молекулы находятся в нулевых состояниях $\phi$, то, в каком-то смысле, любая двоичная строка длины $d$ реализована. На самом деле, состояние регистра в целом задается формулой (см. [Shor 1999])

$$\psi = \frac{1}{2^{d/2}} \sum_{a=0}^{2^d-1} \mid a \rangle,$$

теперь $|a\rangle$ означает совокупное состояние, где ориентация молекул соответствует битам числа $a$. Например, в состоянии с номером $d = 5$, или $|10110\rangle$,

[6] Кубит — единица информации в квантовом компьютере. — *Прим. перев.*

атомы азота расположены «сверху, снизу, сверху, сверху, снизу». Нет ничего сверхъестественного в том, что вероятность обнаружить весь регистр в заданном состоянии $a \in [0, 2^d - 1]$ составляет $1/2^d$. Именно в этом смысле любое значение $a$ содержится в регистре — полный набор значений $a$ хранится в виде «суперпозиции».

Для состояния, включающего все числа $a \in [0, q-1]$, можно определить действие на кубиты унитарными операторами. Например, мы можем изменять 0-й и 7-й кубиты, действуя на их состояния матричным оператором. Здесь уместна физическая аналогия, когда в рамках эксперимента по интерференции два пучка света, каждый поляризован в одном из двух возможных направлений, пропускаются через щель (с расположенным внутри поляризационным светофильтром). Такое унитарное преобразование сохраняет суммарные вероятности, перераспределяя амплитуды между состояниями.

Пусть $q > n$, а $x$ — некоторый вычет $(\mathrm{mod}\ n)$, тогда действуя подходящими унитарными операторами, можно перевести регистр в состояние

$$\psi' = \frac{1}{2^{d/2}} \sum_{a=0}^{2^d-1} \mid x^a \bmod n \rangle,$$

которое опять будет суперпозицией. Отличие возникает при определении вероятности того, что регистр находится в состоянии $\mid b \rangle$. Эта вероятность равна нулю, если только $b$ не является вычетом степени $a$ по модулю $n$.

В завершение нашего беглого обзора отметим, что квинтэссенция программирования по принципу «разделяй и властвуй», а именно, алгоритм БПФ, допускает лаконичную QTM-интерпретацию. Оказывается, что действуя унитарными операторами (как и выше, двухкоординатными) в определенной последовательности, можно привести систему в состояние

$$\psi'' = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} e^{2\pi i ac/q} \mid c \rangle,$$

что, по крайней мере теоретически, дает возможность реализовывать на QTM многие интересные алгоритмы с полиномиальной сложностью. А пока заметим, что операции сложения, умножения, деления, модулярного возведения в степень и БПФ могут выполняться за время $O(d^\alpha)$, где $d$ обозначает количество кубитов в регистре (которых конечное число), а $\alpha$ — некоторая подходящая константа. Подробное изложение реализации этих фундаментальных операций можно найти в любом из перечисленных источников. Хоть до сих пор никто и не производил вычисления на QTM — в лабораториях к настоящему времени создано лишь несколько микроскопических образцов — из литературных трудов предстает достаточно четкая картина. Считается, что параллельная реализация большого числа операций над $d$-разрядными числами, за полиномиальное относительно $d$ время, реальна.

## 8.5.2. Квантовый алгоритм факторизации Шора

Только что мы вкратце познакомились с общими понятиями QTM-теории, теперь также коротко расскажем о некоторых новых квантовых алгоритмах, имеющих отношение к проблемам теории чисел. В работах [Shor 1994, 1999] была предложена остроумная идея факторизации $n$ посредством вычисления мультипликативных порядков случайных чисел $(\mathrm{mod}\ n)$ согласно следующему предложению.

**Предложение 8.5.1.** *Пусть нечетное число $n > 1$ имеет ровно $k$ различных простых делителей. Тогда для произвольного элемента $y$ из $\mathbf{Z}_n^*$ с мультипликативным порядком $r$ вероятность того, что $r$ — четное, и $y^{r/2} \not\equiv -1$ $(\mathrm{mod}\ n)$ составляет не менее $1 - 1/2^{k-1}$.*

(Упр. 8.22 позволяет получить несколько более сильный результат.) Из этого предложения следует, что, по меньшей мере в принципе, число $n$ можно разложить на множители, перебрав «несколько» целых чисел $y$ с соответствующими (четными) порядками $r$. Для этого нужно вычислить

$$\text{НОД}\,(y^{r/2} - 1, n),$$

из чего с неплохой вероятностью определяется нетривиальный множитель $n$, поскольку $y^r - 1 = (y^{r/2} + 1)(y^{r/2} - 1) \equiv 0\ (\mathrm{mod}\ n)$. Точнее, вероятность успеха составляет по крайней мере $1 - 1/2^{k-1}$, что не меньше $1/2$, если только $n$ не является простым или степенью простого числа.

Таким образом, алгоритм Шора сводится к вычислению порядков случайных вычетов по модулю $n$. Для обычной машины Тьюринга это безнадежная задача, связанная с проблемой дискретного логарифмирования (DL). Однако присущий QTM параллелизм позволяет определять мультипликативные порядки вычетов сравнительно несложно. Мы воспроизводим вариант алгоритма Шора по работам [Williams and Clearwater 1998] и [Shor 1999]. Обратим внимание, что несмотря на то, что соответствующая машина не была построена, предлагаемый алгоритм должен работать. И ничто не мешает тот же алгоритм испытать на стандартной машине Тьюринга, ну а потом, конечно, убедиться в том, что QTM, как заявляется, выполняет тот же алгоритм в экспоненциальное число раз быстрее.

**Алгоритм 8.5.2** (квантовый алгоритм факторизации Шора)**.** Пусть дано нечетное число $n$, не являющееся ни простым, ни степенью простого числа. Данный алгоритм при помощи квантовых вычислений пытается возвратить нетривиальный множитель числа $n$.

1. [Инициализация]
   Выбирается $q = 2^d$ такое, что $n^2 \leqslant q < 2n^2$;
   $d$-кубитный квантовый регистр переводится в состояние:
   $$\psi_1 = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |\, a \,\rangle\ ;$$
2. [Выбор основания]

Выбирается произвольное $x \in [2, n-2]$, взаимно простое с $n$;

3. [Вычисление всех степеней]
   Второй $d$-кубитный квантовый регистр переводится в состояние:
$$\psi_2 = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} | \, x^a \bmod n \, \rangle;$$
4. [Осуществление квантового БПФ]
   БПФ применяется к первому квантовому регистру — он переходит в смешанное состояние
$$\psi_1 = \frac{1}{q} \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} e^{2\pi \imath ac/q} \, | \, c \, \rangle \, | \, x^a \bmod n \, \rangle \, ;$$
5. [Определение периода $x^a$]
   Измеряется состояние машины; из измерения и расчетов, как в классической машине Тьюринга, период $r$ определяется как минимальное число, удовлетворяющее $x^r \equiv 1 \pmod{n}$;
6. [Вывод]
   Если $r$ — нечетное, перейти к пункту [Выбор основания];
   Используя предложение 8.5.1, предпринимается попытка найти нетривиальный множитель числа $n$. В случае неудачи перейти к пункту [Выбор основания];

Мы намеренно оставили без подробного разъяснения заключительные шаги алгоритма. Превосходное и детальное изложение этих шагов содержится в работе [Shor 1999]. Главная идея, лежащая в основе шага [Определение периода ...], состоит в следующем. После выполнения БПФ, машина может быть обнаружена в состоянии $| \, c \, \rangle | \, x^k \bmod n \, \rangle$ с вероятностью

$$P_{c,k} = \left| \frac{1}{q} \sum_{\substack{a=0 \\ x^a \equiv x^k \pmod{n}}}^{q-1} e^{2\pi \imath ac/q} \right|^2 = \left| \frac{1}{q} \sum_{b=0}^{\lfloor (q-k-1)/r \rfloor} e^{2\pi \imath (br+k)c/q} \right|^2 \tag{8.4}$$

Далее, можно показать, что значение этого выражения имеет яркие «всплески» при определенных зависящих от $r$ значениях $c$. По этим «всплескам», которые, по предположению, обнаруживаются все одновременно с измерением состояния QTM, можно посредством несложных дополнительных расчетов вычислить период $r$. Некоторые важные моменты освещены в упр. 8.22, 8.23, 8.24, 8.36. Из последнего упражнения следует, что дискретное логарифмирование (DL) выполняется на QTM за полиномиальное время.

Справедливости ради отметим, что квантовые компьютеры — это не только вычислительные машины, о которых ходят разговоры, но которые до сих пор не имеют никакого практического воплощения. В 1999 г. Шамир привел описание «Twinkle»-машины[7] для факторизации чисел (см. [Shamir 1999]). Данное

[7] *«Twinkle»* в переводе означает «огонек». — *Прим. перев.*

устройство представляет собой специальный оптоэлектронный процессор, реализующий либо метод квадратичного решета (QS), либо метод решета числового поля (NFS). Еще одно направление предположительного развития вычислительных машин будущего — это идея «ДНК-компьютера», которая опирается на несомненный вычислительный потенциал самых высокоорганизованных живых систем, совершенствовавшихся на протяжении эпох (см. [Paun et al. 1998]). Если кого-то интересует не столько математическое, сколько популярное изложение, освещающее вопросы компьютерных вычислений будущего, рекомендуем прочитать выпуск журнала *Technology Review* Массачусетского технологического института за май-июнь 2000 г., где содержится популярная подборка сведений о ДНК-, молекулярных и квантовых компьютерах.

## 8.6. Простые числа в смежных областях, забавные и любопытные факты

Подобно тому как простые числа нашли приложения в криптографии, статистике и других областях, связанных с вычислениями, они применяются и в таких далеких от математики дисциплинах, как физика, химия, биология и инженерное дело. И даже более того, о вовлечении простых чисел в более широкий, можно сказать, более жизненный контекст свидетельствуют занимательные истории из фольклора. Помимо научных связей, есть то, что можно назвать «культурными» связями. Фактами применения простых чисел в смежных областях можно наполнить содержание целой книги. Понимая это, мы отобрали несколько ярких примеров таких междисциплинарных связей, краткое знакомство с которыми завершает эту главу.

По праву одним из пионеров изучения междисциплинарного аспекта является Шрёдер. Его работы за последнее десятилетие, посвященные широкому применению простых чисел в технике (см., например, статью [Schroeder 1997]), производят неизгладимое впечатление. Из составляющих содержание подобных работ примеров междисциплинарных связей, назовем лишь несколько. Это конечные поля $\mathbf{F}_q$, принимающие участие в построении кодов с исправлением ошибок, дискретное преобразование Фурье (ДПФ) в акустике, функция Мёбиуса $\mu$ и другие функции в разных науках и т.д. Чтобы показать, насколько глубоким может быть взаимопроникновение наук, скажем (здесь мы ссылаемся на Шрёдера), что в некоторых астрономических опытах, предпринятых для проверки выводов общей теории относительности Эйнштейна, обработка очень слабого сигнала производилась при помощи кодов с исправлением ошибок (а значит, и конечных полей). Аргументы такого типа показывают, что достижения культуры и науки, в какой-то степени, опираются на простые числа. Большое удовольствие может доставить изучение работы [Caldwell 1999], в которой описываются исследования в различных областях с применением простых чисел.

В биологии простые числа появляются, например, в таком контексте. Мы здесь цитируем работу [Yoshimura 1997] с тем, чтобы показать, какую роль

могут играть простые числа в области знания или культуры. Там практически не привлекается привычная теоретико-числовая терминология, а, как правило, используются интуитивные суждения:

> Периодические цикады (Magicicada spp.) замечательны тем, что они появляются на свет удивительно скоординированным по времени образом, сильно привязаны к одному месту и обладают исключительно длинными для насекомых (17- и 13-летними) жизненными циклами. Высказано несколько гипотез, объясняющих целесообразность синхронизации и механизмы, обеспечивающие ее. Но не известно удовлетворительного объяснения жизненным циклам, длящимся простое число лет. Согласно моей эволюционной гипотезе, вынужденная задержка в развитии произошла по причине похолодания климата в ледниковом периоде. В данной концепции исключительно низкая плотность популяции взрослых особей, обусловленная очень длительной стадией созревания, привела к синхронизированному времени вылета и несменяемости местообитания вследствие ограниченных возможностей для спаривания. Простые числа (13 и 17) были выбраны в качестве жизненных циклов за то, что, видимо, эти циклы с наименьшей вероятностью совмещаются, смешиваются или поглощаются другими синхронными циклами.

Интересно, что в довольно обширной литературе, предшествующей работе Йошимуры, содержится по меньшей мере три различных объяснения, почему простые числа 13 и 17 стали длинами жизненных циклов. Понятно, что в каждой теории, старой или новой, должен обыгрываться факт минимального числа делителей у простых чисел. И на самом деле, такие примеры есть в литературе (список соответствующих трудов приведен в той же работе [Yoshimura 1997]). Чтобы иметь представление о том, как объясняют возникновение жизненных циклов с простым периодом, представим себе хищника с жизненным циклом 2 года (четное число), привязанным, естественно, к вызываемому Солнцем чередованию времен года. Эта двухлетняя периодичность проявляется в каждой стороне жизни, будь то воспроизведение потомства или смерть. Так как ни 13, ни 17 не делятся на этот период, хищники время от времени становятся довольно голодными. Это не единственный способ объяснения — угроза со стороны хищников часто вовсе не используется в аргументации, а акцент ставится на внутривидовую конкуренцию и приспособленность видов с простыми жизненными циклами самих по себе. Однако в каждом эволюционном объяснении отмечается недостаток делимости, как то и должно быть. Другими словами, из подобных рассуждений должно стать ясно, помимо всего прочего, почему жизненный цикл с большим числом делителей приводит к вымиранию вида.

Другой пример, в котором встречаются благородные простые числа — теперь в связи с молекулярной биологией — рассматривается в работе [Yan et al. 1991]. Авторы установили, что некоторые аминокислотные последовательности в генетическом коде содержат (двоичную) запись простых чисел. Процитируем один из фрагментов:

> Аддитивно генерируемые числа могут как быть, так и не быть простыми. Мультипликативно генерируемые числа не являются простыми (т. е. они «составные», в терминах теории чисел). Таким образом, простые числа более «созидательны», чем не простые ... . Созидательность и неделимость простых чисел приводит к выводу, что простые числа, меньшие чем 64, являются числовыми эквивалентами аминокислот. Иначе говоря, аминокислоты — это своего рода евклидовы единицы молекул живых организмов.

Далее авторы вводят в своей теории правила диофантова анализа. Авторы далеки от того, чтобы ставить под сомнение распространенное в приложениях представление о том, что составные числа, в каком-то смысле, несут меньше информации (менее совершенны), чем простые. Более того, оно легко просматривается в этом исследовании генетического кода.

Обратимся теперь к некоторым приложениям теории простых чисел в конкретном разделе физики. Выше мы уже касались связи между квантовыми вычислениями и теоретико-числовыми проблемами. Помимо нее, есть гипотеза Гильберта—Пойа с захватывающей воображение историей. По сути, гипотеза утверждает, что поведение дзета-функции Римана на критической прямой $\mathrm{Re}(s) = 1/2$ определенным образом связано с некоторым загадочным (комплексным) эрмитовым оператором: нетривиальные нули дзета-функции являются собственными значениями оператора.[8] Каждый приведенный здесь факт (даже из теории элементарных частиц) имеет непосредственное отношение к простым числам, как мы знаем из главы 1. Вопрос о распределении собственных значений определенных матриц находился в центре внимания физиков-теоретиков на протяжении десятилетий. В начале 1970-х состоялась случайная беседа между одним из ведущих физиков-исследователей в области случайных матриц Дайсоном и Монтгомери, специалистом по теории чисел, изучавшим связь между нетривиальными нулями дзета-функции и простыми числами. Ученые пришли к выводу, что некоторые особенности распределения собственных значений случайных матриц схожи с распределением нетривиальных нулей. В результате появилась широко известная гипотеза о том, что таинственный оператор, открывающий дорогу к изучению свойств дзета-функции, принадлежит классу гауссовых унитарных ансамблей (GUE). В этой теории рассматриваются $n \times n$-матрицы $G$, для которых выполняются условия $G_{aa} = x_{aa}\sqrt{2}$ и, для $a > b$, $G_{ab} = x_{ab} + iy_{ab}$, а также эрмитово тождество $G_{ab} = G^*_{ba}$, где все $x_{ab}, y_{ab}$ являются гауссовыми случайными величинами с нулевым математическим ожиданием и единичной дисперсией. В работах [Odlyzko 1991, 1992, 1994, 2000] показано, что эмпирически статистика упорядоченных нетривиальных нулей дзета-функции во многих отношениях эквивалентна теоретическому распределению собственных значений достаточно большой такой матрицы $G$. Например, обозначим через $\{z_n : n = 1, 2, \ldots\}$ в порядке возрастания множество (положительных) мнимых частей нетривиаль-

[8] В теории дзета-функции Римана принято нули, расположенные в критической полосе $\mathrm{Re}(s) \in (0, 1)$, называть нетривиальными. Согласно гипотезе Римана все нетривиальные нули расположены на критической прямой $\mathrm{Re}(s) = 1/2$. — *Прим. перев.*

ных нулей дзета-функции. Из более развитой теории дзета-функции известно, что величина

$$\delta_n = \frac{z_{n+1} - z_n}{2\pi} \ln \frac{z_n}{2\pi}$$

в среднем равняется 1. Но компьютерный график гистограммы значений $\delta$ обнаруживает поразительное сходство с (рассчитанной теоретически) статистикой собственных значений GUE-матрицы. Такие сопоставления проводились на более чем $10^8$ нулях, соседствующих с $z_N$, где $N \approx 10^{20}$ (а в работе [Odlyzko 2000] даже рассматривалось $10^{10}$ нулей с еще большими порядковыми номерами). Таким образом, однозначно напрашивается вывод: более чем вероятно существование оператора, собственные значения которого есть в точности нетривиальные нули дзета-функции Римана (с поправкой на логарифмический множитель). Однако далеко не все так ясно, как могло бы быть. Так, в частности, график преобразования Фурье

$$\sum_{N+1}^{N+40000} e^{ixz_n},$$

построенный Одлыжко, не отражает монотонное убывание (по $x$), характерное для собственных значений GUE-оператора. В действительности, в точках $x = p^k$, т. е. степенях простых чисел, наблюдаются «всплески». С теоретико-числовой точки зрения этот эффект не является неожиданным. Но мысля физически, можно сказать, что последовательность нетривиальных нулей характеризуется «корреляционной зависимостью на всем протяжении». А еще ранее было отмечено, что подобное поведение возможно в случае, когда нетривиальные нули являются собственными значениями не GUE-оператора как такового, а некоторого неизвестного гамильтониана соотвествующей хаотической динамической системы. Большая и многообещающая работа была проделана в этом направлении (получившем название «квантовой хаологии») Берри и другими учеными (см. [Berry 1987]).

Есть и еще примеры, связывающие понятия из физики с дзета-функцией Римана. Так, о любопытной связи между дзета-функцией и квантовым осциллятором упоминается в работе [Borwein et al. 2000]. Так, в 1991 г. Крэндалл показал, что существует гладкая нигде не обращающаяся в нуль квантовая волновая функция $\psi(x, 0)$ такая, что через некоторое конечное время $T$ эволюции под действием оператора Шрёдингера она превращается в волновую функцию $\psi(x, T)$ с очень замысловатым графиком, имеющую бесконечное число нулей, которые являются в точности нулями $\zeta(1/2 + ix)$ на критической прямой. Более точно, для рассматриваемой волновой функции, отвечающей заданному времени $T$, можно выписать разложение в ряд по собственным функциям

$$\psi(x,T) = f\left(\frac{1}{2} + ix\right)\zeta\left(\frac{1}{2} + ix\right) = e^{-x^2/(2a^2)} \sum_{n=0}^{\infty} c_n(-1)^n H_{2n}(x/a), \qquad (8.5)$$

где $a$ — некоторое положительное вещественное число, $(c_n)$ — определенная последовательность вещественных зависящих от $a$ коэффициентов, а $H_m$ — это

многочлен Эрмита степени $m$. Здесь $f(s)$ — аналитическая функция от $s$, не имеющая нулей. Интересно, что если суммирование по $n$ ограничить, скажем, первыми $N$ слагаемыми, то корни полученного многочлена степени $2N$ будут достаточно точно приближать нули дзета-функции. Пример $N = 27$ (так что многочлен имеет степень 54) приведен в той же работе [Borwein et al. 2000]. Там вычисляются первые семь нетривиальных нулей, из них самый первый — с точностью до 10 десятичных знаков. Подобным образом, в принципе, можно сколь угодно точно вычислять нетривиальные нули дзета-функции как собственные значения матрицы Гессенберга (которые, в свою очередь, являются корнями некоторого многочлена). Любопытное наблюдение можно сделать в отношении гипотезы Римана. А именно, оставив в приведенном выше выражении члены с индексами, скажем, вплоть до $n = N$, мы ожидаем получить $2N$ комплексных корней многочлена степени $2N$ по переменной $x$. Но практически лишь некоторые из этих $2N$ корней — действительные (такие, что точки $\frac{1}{2}+ix$ расположены на критической прямой). Опять же опытным путем установлено, что при больших $N$ оставшиеся корни многочлена находятся на значительном удалении от критической прямой. Гипотеза Римана, сформулированная в терминах разложения в ряд по многочленам Эрмита, должна как-то объяснить наличие таких нулей, не лежащих на вещественной оси. Также, гипотеза Римана может быть каким-то образом интерпретирована с помощью квантовой динамики, и, вне всяких сомнений, подобный междисциплинарный подход обещает быть весьма плодотворным.

Здесь нельзя не рассказать об одном интересном примере, на этот раз из области техники. Сегодня это может показаться необычным, но в 40-х гг. ученый и инженер ван дер Поль продемонстрировал необыкновенно смелое техническое решение, реализовав «аналоговое» разложение в ряд Фурье. Интеграл, который использовался ван дер Полем — это частный случай ($\sigma = 1/2$) следующего выражения, определенного при $s = \sigma + it$, $\sigma \in (0,1)$ (см. [Borwein et al. 2000]):

$$\zeta(s) = s \int_{-\infty}^{\infty} e^{-\sigma\omega} \left( \lfloor e^{\omega} \rfloor - e^{\omega} \right) e^{-i\omega t} \, d\omega.$$

На самом деле, ван дер Поль (см. [van der Pol 1947]) построил и испытал электрическую схему, аналоговым образом осуществляющую требуемое преобразование для $\sigma = 1/2$. В современном преимущественно цифровом мире все еще открытым остается вопрос, позволяет ли идея, предложенная ван дер Полем, эффективно реализовать, скажем, быстрое преобразование Фурье, чтобы вычислить приближенно интересующий нас интеграл. Можно высказать даже более решительное предположение, что, в принципе, подобным образом может быть построено аналоговое устройство — скажем, очень сложная электрическая схема — хотя бы для распознавания простых чисел или выявления каких-либо их свойств.

В этом месте нашего краткого междисциплинарного обзора следует сделать предостережение. Читатель не должен впасть в заблуждение относительно того, что физики-теоретики всегда стремятся найти подтверждение господству-

ющим гипотетическим представлениям в области простых чисел или дзета-функции Римана. Так, в работе Шлезингера 1986 г. утверждается, что *если* поведение дзета-функции в критической полосе имеет характер «фрактального случайного блуждания» (формально говоря, если нетривиальные нули образуют «полет Леви» в точном стохастическом смысле), то фундаментальные законы теории вероятностей должны быть пересмотрены в том случае, когда гипотеза Римана *верна*.

Все же не следует думать, что роль простых чисел в физике сводится к изучению дзета-функции Римана. В 1994 г. В. С. Владимиров с коллегами написали целую книгу о $p$-адических расширениях полей в теоретической физике. Вот что они пишут:

> Разработка формализма математической и теоретической физики над полем $p$-адических чисел представляет значительный интерес даже независимо от возможных приложений, так как способствует более глубокому пониманию формализма стандартной математической физики. Можно высказать следующий принцип. ***Фундаментальные законы физики должны допускать формулировку, инвариантную относительно выбора числового поля.***

(Выделено авторами.) Процитированный фрагмент соответствует кооперативному духу настоящего раздела. По ссылкам из этой увлекательной книги можно перейти дальше к работам в области $p$-адической квантовой гравитации и $p$-адических релятивистских уравнений Эйнштейна.

Время от времени физиками ставились даже «эксперименты с простыми числами». Например, в работе [Wolf 1997] описывается сигнал, обозначим его через $x = (x_0, x_1, \ldots, x_{N-1})$, где каждая из компонент $x_j$ есть количество простых чисел внутри некоторого интервала. А именно,

$$x_j = \pi((j+1)M) - \pi(jM),$$

где $M$ — это некоторая фиксированная длина интервала. Далее рассматривается дискретное преобразование Фурье (ДПФ)

$$X_k = \sum_{j=0}^{N-1} x_j e^{-2\pi i jk/N}$$

с нулевой компонентой ряда Фурье

$$X_0 = \pi(MN).$$

Примечательно, что данный сигнал обладает спектром (имеется в виду зависимость от индекса $k$) так называемого «$1/f$-шума» — фактически, можно назвать его «розовым» шумом. На самом деле, Вольф утверждает, что

$$|X_k|^2 \sim \frac{1}{k^\alpha} \tag{8.6}$$

с показателем степени $\alpha \sim 1.64\ldots$. Это означает, что в частотной области (т. е. для компонент Фурье с индексами $k$) степенная зависимость, очевидно, выра-

жается дробным показателем. Вольф считает, что это может свидетельствовать о том, что множество простых чисел находится в «самоорганизованном критическом состоянии», отмечая, что предположительно встречаются все возможные (четные) длины интервалов между простыми числами, так что естественная шкала не может быть здесь использована. Подобные свойства также выражены у хорошо известных сложных систем, характеризующихся «$1/k^{\alpha}$-шумом». Хотя степенной закон, с точки зрения асимптотики, может быть несовершенным, Вольф обнаружил, что он выполняется на протяжении очень большого интервала значений $M, N$. Так, при $M{=}2^{16}, N{=}2^{38}$ график в координатах $(\ln|X_k|^2, \ln k)$ выглядит как идеально прямая линия с угловым коэффициентом $\approx -1.64$. Появится или нет теория, удовлетворительно объясняющая этот степенной закон (в конце концов, это могло быть случайным совпадением, не распространяющимся на очень большие простые числа), в любом случае представляется интересным найти взаимосвязь между организацией сложных систем и структурой множества простых чисел (см. упр. 8.33).

В общественных отношениях (если угодно, ненаучной сфере) важную роль играют небольшие простые числа, такие как 2, 3, 5, 7; свидетельства тому берут начало в библейских преданиях и встречаются в современном фольклоре. Из многих написанных в последнее время заметок, выделим статью [Paulos 1995] «По мотивам великолепной пятерки»[9], опубликованную в деловом журнале «Форбс», где с юмором обыгрываются предрассудки, связанные с числом 5, происходящие из факта пяти пальцев на руке человека. Числу 7 также досталась изрядная порция «эфирного времени», как всегда и было. В статье [Stuart 1996] «Магия числа семь», взятой из (откуда бы вы думали?) медицинского журнала, в частности, перечисляются: «Семь периодов человеческой жизни, семь морей и океанов, семь смертных грехов, семимильные сапоги-скороходы, семь кругов ада, семь чудес света, семь столбов мудрости, Белоснежка и семь гномов, «7-Up» ...». Автор затем рассказывает, какую важную роль играло число 7 в многовековой традиции исцеления, идущей от Гиппократа, что выражалось, например, в количестве дней для купания в специальной воде для обретения хорошего здоровья. Примечательно, что небольшие простые числа на протяжении тысячелетий с завораживающей и мистической силой притягивали к себе всех людей, независимо от их приверженности к математике. Конечно, многое из сказанного справедливо и в отношении небольших составных чисел, таких как 6 или 12. И все же, интересно было бы раз и навсегда понять, откуда идет неослабевающая веками сила очарования собственно простых чисел — то ли от высокой их плотности в начале натурального ряда, то ли оттого, что большая часть населения имеет интуитивное представление об особом статусе простых чисел, направляющем к ним человеческое воображение.

К простым числам, и иногда к достаточно большим простым числам, часто обращаются музыкальная теория и музыковедение. Так, мы читаем в работе [Warren 1995]:

[9] В оригинале «High 5 Jive». — *Прим. перев.*

Наборы из 12 звуковых тонов порождаются последовательностями из пяти подряд идущих простых чисел, каждое из которых умножается на любое из трех наибольших чисел в последовательности. Так, из простых последовательностей вплоть до (37, 41, 43, 47, 53), получаются двенадцать музыкальных гамм. Благодаря этим гаммам, рождается благозвучный диссонанс, присущий как музыкальным композициям с компьютерным сопровождением, так и импровизациям на клавишных инструментах.

А теперь приведем выдержку из работы [Dudon 1987]. В ней рассказывается о том, как взаимосвязаны в музыке простые числа и числа Фибоначчи (автором используется нетрадиционное, хотя и очень близкое к нему, определение ряда Фибоначчи):

Золотая гамма — это единственный неравноинтервальный музыкальный порядок, основанный на золотом сечении. Чаще используются равноинтервальные порядки $5, 7, 12, 19, 31, 50$ и т. д. с выбором конкретного числа из ряда Фибоначчи, соответствующего одной и той же универсальной золотой гамме, основанной на геометрии интервалов, связанных отношением золотого сечения. Автор устанавливает длины и отношения между предлагаемыми интервалами и объясняет, как особое музыкальное звучание происходит от такого цикла из «золотых пятерок», что выражается в практически полном совпадении тональности с интервалами, относящимися друг к другу как первые пять значащих простых чисел (3:5:7:11:13).

Из приведенных выше и многих других музыковедческих работ видно, что не только самые маленькие, но даже и двузначные простые числа играют заметную роль в теории музыки. Как знать, может наступит час, когда еще большие простые числа появятся в подобных исследованиях, особенно если принять во внимание ту стремительность, с которой новые связи возникают в системе «человек-машина-алгоритм» сегодня.

## 8.7. Упражнения

**8.1.** Поясните количественно, что имел в виду Брент, советуя, чтобы запомнить цифры числа 65537, произносить фразу:

*Данное число Ферма еще простое.*[10]

И в продолжение, к какому числу Ферма и к какому его множителю относится высказывание Полларда?

*Я же вас призываю приняться за весьма нам нужный $\rho$-метод атаки на огромные составные. Поллард.*[11]

[10] В оригинале «Fermat prime, maybe the largest». — *Прим. перев.*

[11] В оригинале «I am now entirely persuaded to employ rho method, a handy trick, on gigantic composite numbers». — *Прим. перев.*

**8.2.** За прошедшее время было высказано множество критических замечаний в адрес RSA-криптосистемы. Некоторые исходят из элементарных положений, другие опираются на нетривиальные теоретико-числовые понятия. Проанализируйте один или несколько из приведенных ниже примеров:

(1) Предположим, что ответственный за безопасность хочет облегчить себе жизнь и назначает для всех $U$ пользователей *один и тот же* модуль $N = pq$. Пусть представитель власти наделяет каждого пользователя $u \in [1, U]$ уникальными секретным ключом $D_u$ и открытым ключом $(N, E_u)$. Докажите строго, что система в целом — уязвима.

(2) Покажите, что Алиса может следующим образом обмануть (ничего не подозревающего) Боба, заставив «подписать» провокационное (опасное для Боба) сообщение $x$. Пусть, следуя алгоритму 8.1.4, Алиса выбирает произвольное число $r$ и уговаривает Боба подписать и отправить ей «случайное» сообщение $x' = r^{E_B} x \bmod N_B$. Покажите, что Алиса затем без труда может подобрать $s$ такое, что $s^{E_B} \bmod N_B = x$, тем самым заполучив подписанную версию вредоносного сообщения $x$.

(3) Здесь рассматривается способ взлома, применимый в случае, когда секретный ключ достаточно мал, следуя результату работы [Wiener 1990]. Пусть RSA-модуль $N = pq$, где $q < p < 2q$. Пусть далее, в естественном предположении $ED \bmod \varphi(N) = 1$, выполнено ограничение на величину секретного ключа, $D < N^{1/4}/3$. Сперва покажите, что

$$|N - \varphi(N)| < 3\sqrt{N}.$$

Затем установите, что существует целое число $k$ такое, что

$$\left| \frac{E}{N} - \frac{k}{D} \right| < \frac{1}{2D^2}.$$

Наконец, докажите, что секретный ключ $D$ находится (а открытая пара $N, E$ известна) за полиномиальное время (т. е. за число операций, ограниченное некоторой степенью $\ln N$).

(4) Также предпринимались так называемые «временные» атаки. При использовании машины, вычисляющей такие выражения, как $x^D$, с помощью схемы возведения в степень, в которой операции возведения в квадрат и умножения выполняются за *различное*, но фиксированное время, может быть получена информация о показателе $D$. С помощью криптосистемы сгенерируем достаточно много цифровых подписей $x_i^D \bmod N$, где $i$ пробегает некоторое множество. Для каждого $i$ запомним время $T_i$, затраченное на выполнение $i$-й подписи. Проделаем такой же «временной» эксперимент, но теперь, скажем, со всеми $x_i^3$ (получим временную последовательность $\{t_i\}$). Покажите, как из корреляции между множествами $\{t_i\}$ и $\{T_i\}$ можно определить разряды секретного ключа $D$.

Мы дали лишь самое поверхностное представление об RSA-атаках. Можно упомянуть также о технологии взлома, основанной на «приведении решетки» (см. [Coppersmith 1997]) и любопытных выводах работы [Boneh and Venkatesan

1998], опирающихся на (частичную) взаимосвязь между разложением на множители и взломом RSA. Обзор результатов в этой области можно найти, например, в работе [Boneh 1999]. Мы выражаем благодарность Kao за некоторые идеи, которые легли в основу данного упражнения.

**8.3.** Мы уже отмечали выше, что для шифрования со встраиванием по алгоритму 8.1.10, в принципе, не обязательно использовать $y$-координаты и связующую «точку-ключ». При помощи алгоритма 7.2.8 и генератора Миллера (см. уравнение (8.1)) постройте точный и подробный алгоритм для «прямого встраивания», не использующий $y$-координаты, без увеличения объема передаваемых данных (не считая неизбежной передачи знакового бита $d$, что асимптотически можно не считать увеличением). Можно дополнительно ввести несколько «битов четности», например, для выбора одного из двух квадратных корней в алгоритме 7.2.8 и т. д.

**8.4.** Опишите способ «встраивания» открытого целого числа $X \in \{0, \ldots, p-1\}$ на *одну* заданную кривую, в котором каким-то образом число $X$ последовательно увеличивается (при необходимости) до тех пор, пока $X^3 + aX + b$ не станет квадратичным вычетом (mod $p$). Подобная схема описана в статье [Koblitz 1987].

**8.5.** В каком случае в алгоритме 8.1.10 $X$ является $x$-координатой точки на обеих кривых $E, E'$?

**8.6.** При любом использовании параметризации Монтгомери (алгоритм 7.2.7) в любом криптографическом режиме нам недоступно точное значение $Y$-координаты. Фактически для пары координат Монтгомери $(X, Z)$ нам известно, что $Y^2 = (X/Z)^3 + c(X/Z)^2 + a(X/Z) + b$, так что могут существовать две возможности для $Y$. Объясните, как Алиса, которая собирается отправить Бобу точку $(X, Y)$ на кривой, может осуществить так называемое «сжатие точек», в том смысле, что она может отправить Бобу $X$-координату и небольшую дополнительную информацию.

Но перед тем как отправить точную информацию, Алисе нужно *самой* знать правильное значение корня $Y$. Разработайте криптографическую схему (например, схему обмена ключом), в которой используется алгебра координат Монтгомери $(X, Z)$, а $Y$ каким-то образом восстанавливается. (Одна из причин, по которой желательно иметь значение $Y$, состоит в том, что этого требуют некоторые современные криптографические стандарты.) Интересное исследование [Okeya and Sakurai 2001] связано с проблемой построения такой схемы. На самом деле подобные аспекты, обычно связанные с переносом эффективных криптосистем ECC на чипы и смарт-карты, широко освещены в современной литературе. Простой поиск сайтов о ECC-оптимизациях в Интернете сегодня выдает большое количество очень свежих ссылок. Лишь одним (из многих) источников, с которого можно начать знакомство с этой темой, является работа [Berta and Mann 2002] и работы из ее списка литературы.

**8.7.** Разработайте протокол подбрасывания монеты, используя тот факт, что если $n$ есть произведение двух различных нечетных простых чисел, то квадратичные вычеты по модулю $n$ имеют 4 квадратных корня вида $\pm a, \pm b$. По-

следующее вычисление этих корней для заданного квадратичного вычета не представляет труда, если известно разложение числа $n$ на множители. И наоборот, факторизация $n$ легко осуществляется, когда известны все 4 квадратных корня. В этой связи обратите внимание на числа Блюма из упр. 2.26, которые нередко используются в протоколах подбрасывания монеты. См. также работы [Schneier 1996] и [Bressoud and Wagon 2000, p. 146].

**8.8.** Проанализируйте потенциальные криптографические изъяны в алгоритме 8.1.11. К примеру, умение быстро раскладывать на множители число $n$ позволяет Бобу провести нечестную махинацию. Таким образом, безопасность данного протокола, как и многих других, априори покоится на предполагаемой трудности факторизации отправляемого Алисой числа $n$. Может ли Алиса обмануть Боба, каким-то образом склонив его к выбору в пользу какого-либо конкретного простого числа? Может ли Боб увеличить свои шансы, если знает или догадывается о том, что Алиса выбирает простые числа $p, q, r$ случайным образом в определенном диапазоне? Способна ли одна из сторон проиграть умышленно?

**8.9.** После описания алгоритма 8.1.11 упоминалось о том, что протокол подбрасывания монеты может быть модифицирован под коллективные игры, такие как покер. Выберите конкретный протокол (из алгоритма в тексте или же из источников, ссылки на которые см. в упр. 8.7) и *подробно* опишите проект «телефонного покера», в котором посредством общедоступного телефонного соединения игрокам раздается, скажем, по 5 карт, в нужное время карты раскрываются и т. п. Интуитивно ясно, что если можно реализовать «подбрасывание монетки», точно так же можно реализовать эту игру в покер. Но цель упражнения в том, чтобы в деталях разработать алгоритм законченной игры.

**8.10.** Докажите, что этап верификации в алгоритме 8.1.8 работает корректно. Подумайте, какова вероятность пропуска ложной цифровой подписи и как сложно ее подделать.

**8.11.** Постройте генератор случайных чисел, используя одностороннюю функцию.[12] Оказывается, что для этой цели годится совершенно *произвольная* односторонняя функция. См. работу [Hastad et al. 1999] или [Lagarias 1990].

**8.12.** Реализуйте быстрый qMC-алгоритм 8.3.6 на базе последовательности Холтона размерности $D = 2$. Изобразите графически «облако» из нескольких тысяч точек в единичном квадрате. Оцените качественное (визуальное) различие между вашим графиком и графиком точек со случайно распределенными координатами.

**8.13.** Докажите утверждение для уравнения (8.3) при оговоренных условиях на параметр $k$. Начните с исследования диофантова уравнения (mod 4) и покажите, что $x \equiv 1 \pmod 4$. Продолжайте анализировать уравнение (mod 4), пока не придете к символу Лежандра $\left(\frac{-4m^2}{p}\right)$ для $p \equiv 3 \pmod 4$. (См., например в книге [Apostol 1976, Section 9.8].)

[12] Односторонняя функция — функция, обратное значение которой сложно вычислить (например, показательная функция в кольце вычетов или конечном поле). — *Прим. перев.*

**8.14.** Обратите внимание, что если $c = a^n + b^n$, то $x = ac$, $y = bc$, $z = c$ является решением уравнения $x^n + y^n = z^{n+1}$. Покажите, что справедливо более общее утверждение: если НОД$(pq, r) = 1$, то уравнение Ферма—Каталана $x^p + y^q = z^r$ имеет бесконечно много положительных решений. Почему это не опровергает гипотезу Ферма—Каталана? Покажите, что положительных решений не существует при НОД$(p, q, r) \geqslant 3$. Что можно сказать о случаях НОД$(p, q, r) = 1$ и 2? (Ответ на последний вопрос авторам не известен.)

**8.15.** Оформите более или менее правдоподобное эвристическое рассуждение в пользу гипотезы Ферма—Каталана. Например, приведем рассуждение для случая, когда $p, q, r$ все не меньше 4. Пусть $S$ есть множество четвертых и более высоких степеней натуральных чисел. Если на то нет очевидных причин, таких, как в упр. 8.14, ничто особенно не заставляет сумму двух элементов из $S$ быть равной третьему элементу из $S$. Рассмотрим выражение $a+b-c$, где $a \in S \cap [t/2, t]$, $b \in S \cap [1, t]$, $c \in S \cap [1, 2t]$, и НОД$(a, b) = 1$. Число вида $a+b-c$ лежит в интервале $(-2t, 2t)$, и вероятность того, что оно равно 0, по-видимому, должна быть порядка $1/t$. Следовательно, ожидаемое число решений уравнения $a+b = c$ для таких $a, b, c$ должно не превосходить $S(t)^2 S(2t)/t$, где $S(t)$ есть число элементов множества $S \cap [1, t]$. Но $S(t) = O(t^{1/4})$, так что наше ожидаемое число решений есть $O(t^{-1/4})$. А теперь пусть $t$ пробегает степени числа 2. Тогда ожидаемое общее число решений составит $O(1)$.

**8.16.** По аналогии с упр. 8.15 оформите более или менее правдоподобное эвристическое рассуждение в пользу ABC-гипотезы.

**8.17.** Покажите, что ABC-гипотеза неверна в случае $\varepsilon = 0$. Более конкретно, покажите, что существует бесконечно много троек $a, b, c$ натуральных чисел, таких что $a + b = c$ и $c = o(\gamma(abc))$. (Как и прежде, через $\gamma(n)$ обозначен наибольший свободный от квадратов делитель $n$.)

**8.18.** [Тайдеман] Покажите, что из ABC-гипотезы следует гипотеза Ферма—Каталана.

**8.19.** [Сильверман] Покажите, что из ABC-гипотезы следует, что существует бесконечно много простых чисел $p$, которые не являются числами Вифериха.

**8.20.** Пусть $q_1 < q_2 < \ldots$ — последовательность неединичных степеней натуральных чисел, т. е. $q_1 = 1$, $q_2 = 4$, $q_3 = 8$, $q_4 = 9$, и т. д. Достоверно не известно, стремится ли длина промежутка $q_{n+1} - q_n$ к бесконечности с ростом $n$, однако покажите, что это следует из ABC-гипотезы. А именно, с помощью ABC-гипотезы докажите, что для любого $\varepsilon > 0$ справедливо неравенство $q_{n+1} - q_n > n^{1/12-\varepsilon}$, начиная с некоторого достаточно большого значения $n$.

**8.21.** Покажите, что существует многочлен двух переменных с целыми коэффициентами, множество значений которого на натуральных аргументах совпадает с множеством составных натуральных чисел. Затем, с помощью теоремы Лагранжа о том, что всякое натуральное число представимо в виде суммы 4 квадратов (см. упр. 9.41), постройте многочлен 8 переменных с целыми коэффициентами, значения которого на множестве целочисленных аргументов составляют множество составных натуральных чисел.

**8.22.** Пусть целое число $n$ из предложения 8.5.1 имеет различные простые делители $p_1, \ldots, p_k$, где $2^{s_i} \| p_i - 1$ и $s_1 \leqslant \ldots \leqslant s_k$. Покажите, что в этом случае соответствующая вероятность составляет

$$1 - 2^{-(s_1+\ldots+s_k)} \left(1 + \frac{2^{s_1 k} - 1}{2^k - 1}\right),$$

что не меньше, чем $1 - 2^{1-k}$. (Сравните с упр. 3.15.)

**8.23.** Завершите проверку корректности одного из шагов алгоритма факторизации Шора предлагаемым ниже образом. В соотношении (8.4) задана вероятность $P_{c,k}$ обнаружить QTM в смешанном состоянии $|\, c \,\rangle |\, x^k \,\rangle$. Приведите расчет, подтверждающий, что выражение для вероятности (при фиксированном $k$ и переменной $c$) демонстрирует «всплески» в точках, соответствующих решениям $d$ задачи нахождения диофантовых приближений

$$\left| \frac{c}{q} - \frac{d}{r} \right| \leqslant \frac{1}{2q}.$$

Покажите затем, как, если известно (измерено) $c$, можно найти подходящую дробь $d/r$ с минимальными числителем и знаменателем. Докажите, что если НОД$(d, r)$ равен 1, то такая процедура определяет в точности период $r$ для алгоритма Шора, а мы знаем, что вероятность взаимной простоты двух случайных чисел равна $6/\pi^2$.

Выполните численное моделирование (конечно, с помощью классической машины Тьюринга) спектрального поведения QTM в конце алгоритма 8.5.2 на следующем показательном примере. Пусть $n = 77$, так что на шаге [Инициализация] выполняется присвоение $q = 8192$. Выбирая теперь $x = 3$ (исходя из предшествующего опыта), на шаге [Определение периода...] получаем $r = 30$. Конечно, в действительности, QTM должна измерить величину периода физически и быстро! Далее, в продолжение моделирования работы QTM, примените (обычное) БПФ и изобразите графически с помощью формулы (8.4) функцию вероятности $P_{c,1}$ в зависимости от $c$. При определенных значениях $c$ должны наблюдаться ярко выраженные «всплески», например, при $c = 273$. Затем из соотношения

$$\left| \frac{273}{8192} - \frac{d}{r} \right| \leqslant \frac{1}{2q}$$

выводится результат $r = 30$ (в литературе описывается, как с помощью цепных дробей находить подходящие приближения $d/r$). Окончательно, вычислив НОД$(x^{r/2} - 1, n)$, найдите делитель числа $n$. Мы предлагаем проделать все эти шаги, чтобы читатель прочувствовал некоторые моменты, опущенные при изложении алгоритма 8.5.2. Кроме того, это служит подготовкой к более основательной эмуляции функционирования QTM (см. упр. 8.24). Важно заметить, что эмуляция на обычной машине Тьюринга факторизации даже такого маленького числа $n$ требует БПФ с использованием тысяч битов памяти, тогда как настоящая QTM обходится всего-то дюжиной кубитов или около того.

**8.24.** Было бы очень полезно построить детальную формализацию алгоритма 8.5.2, опираясь как на наше краткое изложение, так и на многочисленные

публикации, где некоторые детали освещаются более подробно (и, в том числе, на соображения из упр. 8.23).

Вторая задача, которая с педагогической точки зрения представляется очень важной, состоит в эмуляции QTM на базе стандартной программно-реализуемой машины Тьюринга, написанной на некотором стандартном языке. Естественно, это не позволит нам выполнять факторизацию за полиномиальное время, но лишь потому, что на машину Тьюринга возложена функция QTM (первая работает в экспоненциальное число раз медленнее). Для тренировки можно начать с численного примера из упр. 8.23. Заметим, что свобода выбора в отдельных аспектах реализации еще есть. Так, при скрупулезном подходе, можно очень точно смоделировать квантовую интерференцию. Иначе, при реализации алгебраических шагов алгоритма 8.5.2, можно использовать классическую арифметику и БПФ.

## 8.8. Проблемы для исследования

**8.25.** Докажите или опровергните утверждение физика Бродхерста:

$$P_{3139} = \frac{2^{903}5^{682}}{514269}\int_0^\infty dx\,\frac{x^{906}\sin(x\ln 2)}{\sinh(\frac{1}{2}\pi x)}\left(\frac{1}{\cosh(\frac{1}{5}\pi x)}+8\sinh^2(\tfrac{1}{5}\pi x\right)$$

есть не просто целое, но даже и простое число. Интеграл такого вида возникает при изучении кратных дзета-функций, которые используются в теоретической физике, в частности, в квантовой теории поля (здесь имеются в виду физические поля, а не алгебраические!).

**8.26.** Здесь мы устанавливаем связь между простыми числами и фракталами. Рассмотрим бесконечномерную матрицу Паскаля $P$, составленную из элементов

$$P_{i,j} = \binom{i+j}{i},$$

где $i$ и $j$ принимают значения 0, 1, 2, 3, ...; это есть просто классический треугольник биномиальных коэффициентов Паскаля с вершиной в левом верхнем углу матрицы $P$, а именно:

$$P = \begin{pmatrix} 1 & 1 & 1 & 1 & \dots \\ 1 & 2 & \mathbf{3} & 4 & \dots \\ 1 & \mathbf{3} & 6 & 10 & \dots \\ 1 & 4 & 10 & 20 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

Данная матрица $P$ обладает многими замечательными свойствами (см., например, в книге [Higham 1996, p. 520]), но в рамках настоящего упражнения нас интересует фрактальная структура этой матрицы с элементами, приведенными по модулю простого числа.

Введем обозначение $Q_n = P \bmod n$, где операция модуля применяется поэлементно. Представьте теперь геометрическую фигуру, которая получится,

если все ячейки матрицы $Q_n$ с нулевыми элементами раскрасить в черный цвет, а с ненулевыми — в белый. Теперь представьте, что берется целиком такое изображение бесконечномерной матрицы $Q_n$ и сжимается до размера некоторого ограниченного квадрата на плоскости. В результате должен получиться некоторый узор, напоминающий «снежинку», из множества черных пятен на белом фоне. А теперь докажите, что если выбран простой модуль $p$, и, соотвественно, матрица $Q_p$, то фрактальная размерность этой похожей на «снежинку» фигуры задается формулой

$$\delta = \frac{\ln\left(p^2\right)}{\ln(p(p+1)/2)}.$$

Формально, это так называемая «размерность бокса»: этот и другие типы размерностей определяются в работе [Crandall 1994b] (см. также ссылки в ней). (Указание: стандартный способ вычисления $\delta$ состоит в подсчете количества ненулевых элементов в левой верхней $(p^k \times p^k)$-подматрице матрицы $Q_p$, которое затем нужно соотнести с аналогичным показателем для подматрицы порядка $p^{2k}$.) Так, например, треугольник Паскаля по модулю 2 имеет размерность $\delta = (\ln 4)/(\ln 3)$, а по модулю 3 — $\delta = (\ln 9)/(\ln 6)$. В случае $p = 2$ получается знаменитая «салфетка Серпинского» — хорошо изученный в теории фракталов объект. Иногда говорят, что эта салфетка «больше, чем линия, но меньше, чем плоскость». Придайте этому неформальному утверждению ясный строгий смысл, используя величину размерности $\delta$.

С данным упражнением на дробные размерности связаны разнообразные задачи. Оказывается, например, что, какое бы ни было простое число $p$, в левой верхней $(p \times p)$-подматрице матрицы $Q_p$ всегда содержится треугольное число ненулевых элементов. (Треугольное число — это число вида $1 + 2 + \ldots + n = n(n+1)/2$.) Вопрос: для какого *составного* числа $n$ количество ненулевых элементов в левой верхней $(n \times n)$-подматрице есть треугольное число? А вот действительно сложный вопрос: какой будет фрактальная размерность, если мы раскрасим нашу фигуру оттенками серого? Т. е., если для вычисления $\delta$ будем использовать не бинарную (черно-белую) структуру «салфетки», а истинный вес каждого элемента матрицы $Q_p$, определяемый по остатку от деления на $p$ числом из интервала $[0, p-1]$.

**8.27.** В криптографии на эллиптических кривых (ECC) важно уметь построить эллиптическую кривую *простого* порядка. Подумайте, как можно модифицировать метод Шуфа (алгоритм 7.5.6), чтобы он служил для «отбора» кривых простого порядка. Другими словами, параметры $a, b$ кривой выбираются случайным, скажем, образом, и небольшие простые числа $L$ используются, чтобы «отвергнуть» рассматриваемую кандидатуру кривой как только $p + 1 - t$ оказывается составным. В предположении, что сложность алгоритма Шуфа $O\left(\ln^k p\right)$, оцените сложность построенной «просеивающей» схемы, имея в виду, что она предназначена для нахождения лишь *одной* эллиптической кривой простого порядка. К слову, выбор максимально допустимых степеней простых чисел $L = 2^a, 3^b$ и т. п. может быть не всегда эффективен (хоть, как

мы объясняли выше, он оправдан в алгоритме Шуфа) для данного «решета». Покажите, почему это так. Обратим внимание, что некоторые из вопросов по теории сложности, которые здесь поднимаются, также обозначены в упр. 7.29 и других упражнениях по данной тематике в той же главе.

Для того, кто справился с реализацией «решета Шуфа» поиска кривой простого порядка, предлагается следующий пример для тестирования программы:

$$p = 2^{113} - 133, \quad a = -3, \quad b = 10018.$$

В качестве модулей рассматриваются простые числа (мы здесь предлагаем набор значений $L$ из степеней простых чисел, хотя, как было сказано, эффективность такого подхода не гарантирована)

$$7, 11, 13, 17, 19, 23, 25, 27, 29, 31, 32, 37, 41, 43,$$

откуда для вычисления порядка кривой $\#E = p+1-t$ получаем набор значений $t \bmod L$ соответственно

$$2, 10, 3, 4, 6, 11, 14, 9, 26, 1, 1, 10, 8, 8,$$

и, в конечном итоге, имеем

$$\#E = 10384593717069655112027224311117371.$$

Заметим, что задача поиска кривых, у которых как порядок $p + 1 - t$, так и сопряженный порядок $p + 1 + t$ являются простыми, весьма нетривиальна, — сродни задаче нахождения простых близнецов против обычного поиска простых чисел. Задание для исследования: методами аналитической теории чисел доказать, что существует константа $c$ такая, что для большинства простых чисел $p$ найдется не менее $c\sqrt{p}/\ln^2 p$ целых чисел $t$ с условием $0 < t < 2\sqrt{p}$, и при этом оба числа $p + 1 \pm t$ простые.

**8.28.** Напишите программу, эффективно тестирующую генераторы случайных чисел. Основная идея проста: обрабатывается входной поток, например, целых чисел. Трудности возникают с ее реализацией. Существуют спектральные, нормальные, общие статистические, конфликтов и многие другие критерии. Программа должна ставить «баллы» генерируемым потокам и, тем самым, выявлять «хорошие» генераторы случайных чисел. Конечно, понятие о том, хороший генератор или плохой, часто зависит от контекста. Например, генератор случайных чисел, успешно применяемый в вычислительной физике для численного интегрирования, с точки зрения криптографии может быть совсем не успешным и т. д. В рамках данного исследования интересно было бы проверить распространенное мнение о том, что хаотические генераторы не удовлетворяют требованиям криптографической безопасности. В этой связи подумайте, можно ли при помощи оценки показателя Ляпунова или фрактальной размерности сгенерированной псевдослучайной последовательности как-то пополнить арсенал статистических тестов.

**8.29.** Изучите вопрос генерирования случайных чисел на основе эллиптических кривых. Возможные направления для исследований намечены в тексте после определения итерации (8.1). В том числе, предполагается возможным

распространить метод генерирования Гонга—Берсона—Стинсона (см. [Gong et al. 1999]) на кривые над полями нечетной характеристики.

**8.30.** Попробуйте построить генератор случайных чисел с периодом, еще большим, чем у генератора Марсальи в тексте. Например, в работе [Brent 1994] замечено, что для любого простого числа Мерсенна $M_q = 2^q - 1$, где $q \equiv \pm 1 \pmod 8$, по всей видимости, существует примитивный трехчлен[13] степени $M_q$, что позволяет построить генератор Фибоначчи с периодом, не меньшим, чем $M_q$. Известен пример $q = 132049$ с просто потрясающе большим периодом!

**8.31.** Несмотря на то, что определение 8.3.1 излишне формально, а теория дискрепансов $D_N, D_N^*$ очень трудоемкая и по сей день незавершенная, все же для дискрепансов получены некоторые любопытные оценки достаточно общего характера. К таким результатам относится теорема Левека о последовательностях $P = (x_0, x_1, \ldots)$ точек, где $x_j \in [0,1]$. В книге [Kuipers and Niederreiter 1974] доказывается изящная оценка

$$D_N \leqslant \left( \frac{6}{\pi^2} \sum_{h=1}^{\infty} \frac{1}{h^2} \left| \frac{1}{N} \sum_{n=0}^{N-1} e^{2\pi i h x_n} \right|^2 \right)^{1/3}$$

Примечательно, что данная оценка неулучшаема в том смысле, что на последовательности $P = (0, 0, \ldots, 0)$ достигается равенство. Задача состоит в том, чтобы найти полезные или интересные последовательности, для которых оценка Левека может быть вычислена. Например, как изменится формула Левека, если последовательность $P$ генерируется линейным конгруэнтным генератором (и все $x_n$ нормированы, скажем, путем деления на модуль генератора)? Замечательно, что таким образом теория сумм рядов Фурье приходит на помощь при изучении qMC-последовательностей.

**8.32.** Здесь рассматривается одна непростая и интересная открытая проблема из qMC-теории. В то время как для qMC-последовательностей с малым дискрепансом известна верхняя оценка

$$D_N^* = O\left( \frac{\ln^D N}{N} \right),$$

наилучшая известная *нижняя* оценка (см. [Veach 1997]) для больших размерностей $D$

$$D_N^* \geqslant C(D) \left( \frac{\ln^{D/2} N}{N} \right).$$

Проблема заключается в том, чтобы попытаться ликвидировать разрыв между порядками $\ln^{D/2}$ и $\ln^D$. Это важно, так как если размерность $D$ очень велика, лишний ln-множитель может фатально повлиять на оценку погрешности.

**8.33.** Постройте теорию для обоснования опытов, описанных в работе [Wolf 1997], из которых Вольф вывел свой степенной закон (8.6). (Заметим, *априори*

[13] Примитивный многочлен — многочлен с целыми взаимно простыми коэффициентами. — *Прим. перев.*

нет гарантии, что здесь поможет какая-то фундаментальная теория; этот закон, возможно, получится вывести, развивая некоторую сравнительно узкую область вычислительной математики!) Например, исследуйте асимптотическое поведение (с ростом $k$) следующего интеграла, который является непрерывным приближением рассматриваемого дискретного преобразования:

$$I(k) = \int_a^b \frac{e^{ikx}}{\ln(c+x)}\,dx,$$

где $a, b, c$ — это фиксированные вещественные константы. Можно ли таким образом обосновать экспериментальный закон $1/k^{1.64}$ (для $|I|^2$)?

**8.34.** Здесь мы расскажем о некоторых совсем недавно появившихся в литературе направлениях исследований, имеющих отношение к гипотезе Римана (RH). Содержание следующих пунктов в чем-то может пересекаться с материалом гл. 1, в которой указывалось на некоторые следствия из гипотезы Римана. Однако настоящая глава имеет выраженный прикладной характер, и нет ничего плохого в том, чтобы привести здесь независимое изложение.

Следующие утверждения равносильны гипотезе Римана. Рассматривайте их как направления для исследований, у которых сильнее выражен вычислительный аспект, однако многообещающим выглядит их теоретический потенциал:

(1) Очень старое условие Рисса (взято из книги [Titchmarsh 1986, Section 14.32]), эквивалентное гипотезе Римана, утверждает, что

$$\sum_{n=1}^{\infty} \frac{(-x)^n}{\zeta(2n)(n-1)!} = O\left(x^{1/4+\varepsilon}\right).$$

Заметьте, что в формуле используются только целочисленные аргументы дзета-функции. Один из вопросов состоит в следующем: можно ли получить хоть какое-нибудь численное выражение для этой суммы? Если такое выражение существует, тогда можно использовать так называемые «возвратные» методы вычисления значений дзета-функции. Эти методы позволяют найти большое множество значений дзета-функции, исходя из ее значений на членах арифметической прогрессии (см. [Borwein et al. 2000]).

(2) Недавно, в работе [Balazard et al. 1999] было показано, что

$$I = \int \frac{\ln|\zeta(s)|}{|s|^2}\,ds = 2\pi \sum_{\mathrm{Re}(\rho)>1/2} \ln\left|\frac{\rho}{1-\rho}\right|,$$

где интегрирование выполняется по критической прямой, а $\rho$ пробегает нетривиальные нули дзета-функции, однако, как и указано, лежащие *справа* от критической прямой. Таким образом, простое утверждение «$I = 0$» равносильно гипотезе Римана. Одна из задач состоит в том, чтобы изучить поведение функции $I(T)$, которая есть сужение интеграла $I$ на интервал $\mathrm{Im}(s) \in [-T, T]$, и попытаться получить напрашивающийся

результат $I(T) \to 0$, по возможности, оценив скорость сходимости. Другой вопрос носит как теоретический, так и вычислительный характер. Если найдется некоторый *исключительный*[14] нуль $\rho = \sigma + it$, где $\sigma > 1/2$ (со своими естественными симметриями), и интеграл может быть посчитан для некоторого $T$ с необходимой точностью, что можно тогда, во всяком случае, сказать о расположении этого нуля? Исключительный интерес представляет следующий вопрос. В случае если гипотеза Римана все же верна, какое положительное число $\alpha$ удовлетворяет

$$I(T) = O(T^{-\alpha}) \ ?$$

В работе [Borwein et al. 2000] было высказано предположение, что, по всей видимости, $\alpha = 2$.

(3) Некоторые из недавно установленных критериев справедливости гипотезы Римана формулируются в терминах стандартной функции

$$\xi(s) = \frac{1}{2} s(s-1)\pi^{-s/2}\Gamma(s/2)\zeta(s).$$

Заманчивый результат был получен в работе [Pustyl'nikov 1999]. Утверждается, что следующее условие, которое применяется к *одной* точке $s = 1/2$,

$$\frac{d^n \xi}{ds^n}\left(\frac{1}{2}\right) > 0$$

при всех $n = 2, 4, 6, \ldots$, эквивалентно гипотезе Римана. С вычислительной точки зрения, было бы интересно вычислить достаточно много таких производных. Одна лишь только отрицательная производная опровергнет гипотезу Римана. Другой критерий, равносильный гипотезе Римана, был предложен в работе [Lagarias 1999]:

$$\mathrm{Re}\left(\frac{\xi'(s)}{\xi(s)}\right) > 0$$

всегда, когда $\mathrm{Re}(s) > 1/2$. И опять, по меньшей мере, познавательный интерес представляет анализ с помощью графических или иных вычислительных средств. Далее, в работах [Li 1997], [Bombieri and Lagarias 1999] было сформулировано еще одно эквивалентное гипотезе Римана «положительное» условие

$$\lambda_n = \sum_{\rho}\left(1 - \left(1 - \frac{1}{\rho}\right)^n\right) > 0$$

для всех $n = 1, 2, 3, \ldots$. Константы $\lambda_n$ выражаются через производные $\ln \xi(s)$; в данном случае все они вычислены для аргумента $s = 1$. И вновь можно говорить о перспективности исследования всевозможных вычислительных аспектов.

Более подробно об этих, с множеством численных примеров, а также о многих других вариациях гипотезы Римана см. в работе [Borwein et al. 2000].

[14] Т. е. не лежащий на критической прямой. — *Прим. перев.*

**8.35.** На сегодняшний день не выяснено, какова граница поиска взаимно простых положительных решений уравнения Ферма—Каталана $x^p + y^q = z^r$ при $1/p + 1/q + 1/r \leqslant 1$. Эта граница поиска, разумеется, охватывает 10 известных решений, упомянутых в этой главе, но возможно, она не существенно больше. Расширьте область поиска решений так, чтобы наибольшая из степеней, а именно $z^r$, могла достигать $10^{25}$, а возможно и еще большей величины. Для ускорения такого вычисления не следует рассматривать те тройки $p, q, r$, для которых известно, что решений не существует. Например, если 2 и 3 принадлежат множеству $\{p, q, r\}$, то мы можем считать, что третий элемент этого множества не меньше 10. См. работы [Beukers 2004] и [Bruin 2003], в которых сообщается об известных на сегодняшний день тройках показателей, для которых производить поиск не обязательно. См. также работу [Bernstein 2004c], содержащую рациональный метод поиска решений в случае наиболее густых множеств.

**8.36.** Разработайте альтернативные методы факторизации и дискретного логарифмирования для квантовых машин Тьюринга (QTM). Мы приводим некоторые соображения (на которые не следует полагаться безусловно).

В методе Полларда—Штрассена из главы 5.5 используются быстрые детерминированные алгоритмы для разложения числа $N$ на множители за $O(N^{1/4})$ операций. Однако, как правило, для получения необходимой полиномиальной оценки применяются БПФ-подобные методы, и, часто, собственно БПФ. Дает ли это возможность, опираясь на присущий QTM глобальный параллелизм, усовершенствовать метод Полларда—Штрассена и получить эффективный детерминированный алгоритм?

Ранее уже предлагались аналогичные упражнения, ориентированные на расспараллеливание ро-метода Полларда, ECM-, QS- и NFS-методов факторизации. Можно взять за правило, что всякий раз как только возникает параллелизм, есть надежда получить эффективный QTM-алгоритм.

Что касается проблемы дискретного логарифмирования (DL), то $\rho$- и $\lambda$-методы допускают параллельные модификации. А DL-метод, предложенный в работе [Shor 1999], несомненно, очень близок к методам конфликтов, которые рассматривались выше. Однако, вероятно, можно построить алгоритм, более удобный для реализации. Так, не будет слишком смелым предположить, что в основу самого первого из алгоритмов дискретного логарифмирования/факторизации, разработанных для QTM, будет положен один из наименее популярных сегодня методов за его простоту. Заметим, что ро-методы практически не требуют никаких других операций, кроме модулярного возведения в квадрат и сложения. (Как и во многих алгоритмах факторизации, перспективных для QTM-реализации, едва ли не к классическим можно отнести операцию НОД.) Более того, в самой сути ро-методов лежит понятие периодичности, а, как мы убедились, квантовые машины (QTM) являются превосходными детекторами периодичности.

Глава 9

# БЫСТРЫЕ АЛГОРИТМЫ ДЛЯ РАБОТЫ С БОЛЬШИМИ ЧИСЛАМИ

В этой главе мы исследуем мир так называемых «быстрых» алгоритмов, которые допускают применение в вычислениях с простыми числами и при разложении на множители. В наше время просто необходимо уметь работать с числами большой разрядности, т. е. с числами, которые на большинстве существующих машин придется разбивать на части, затем к этим частям применять машинные операции и в конце концов заново собирать полученные результаты. Несмотря на то, что сложение и вычитание многоразрядных чисел довольно часто встречаются при изучении вопросов вычислительного характера, мы предполагаем, что концепции этих базовых операций ясны, и начнем с умножения, пожалуй, самой простой операции, классический алгоритм выполнения которой допускает существенные усовершенствования.

## 9.1. Обзор «школьных» методов

### 9.1.1. Умножение

Одной из самых общепринятых технических составляющих нашей культуры является классический, или же «школьный», метод умножения длинных чисел. Хотя позже мы займемся современными методами, обладающими значительной эффективностью, школьный метод остается действующим, особенно когда рассматриваемые числа не велики, и к тому же сам допускает улучшения в скорости. Согласно типичному описанию этого алгоритма, надо просто выписать два перемножаемых числа одно под другим и построить параллелограмм из произведений цифр одного числа на цифры другого. На самом деле параллелограмм будет ромбом, а для завершения умножения нужно будет сложить значения, расположенные по столбцам ромба, с учетом переносов. Если каждое из умножаемых чисел $x, y$ имеет по $D$ цифр в некоторой системе счисления с основанием $B$, то итоговое количество операций, требующихся для вычисления $xy$ равно $O(D^2)$, потому что как раз столько цифр имеется в этом ромбе. Под «операциями» здесь понимаются умножения и сложения чисел, меньших $B$. Такие «поцифровые» умножения мы будем называть «умножениями размера $B$».

Формальное описание школьного метода является простым, но очень содержательным, особенно с учетом последующих улучшений. Дадим два определения:

**Определение 9.1.1.** Представлением неотрицательного целого числа $x$ в системе счисления с основанием $B$ (или $B$-ичной записью числа $x$) называется кратчайшая последовательность целых чисел $(x_i)$, называемых цифрами записи, такая что каждая из этих цифр удовлетворяет неравенству $0 \leqslant x_i < B$, и выполнено равенство

$$x = \sum_{i=0}^{D-1} x_i B^i.$$

**Определение 9.1.2.** Сбалансированным представлением неотрицательного целого числа $x$ в системе счисления с основания $B$ называется кратчайшая последовательность целых чисел $(x_i)$, называемых цифрами этой записи, такая что каждая цифра удовлетворяет неравенству $-\lfloor B/2 \rfloor \leqslant x_i \leqslant \lfloor (B-1)/2 \rfloor$, и выполнено равенство

$$x = \sum_{i=0}^{D-1} x_i B^i.$$

Пусть мы хотим вычислить произведение $z = xy$ для неотрицательных чисел $x, y$. После рассмотрения ромба из школьного метода становится очевидным, что если нам заданы $B$-ичные записи чисел $x, y$, то нам придется суммировать значения по столбцам и находить величины

$$w_n = \sum_{i+j=n} x_i y_j, \tag{9.1}$$

где индексы $i, j$ пробегают все допустимые значения для цифр чисел $x, y$. Такая последовательность $(w_n)$ в общем случае не является представлением произведения $z$ по основанию $B$. Что нам на самом деле было нужно делать — конечно, выполнять сложения $w_n$ с учетом переносов. Операцию переносов проще всего понимать так, как ее объясняют в школе: сумма чисел в столбце $w_n$ влияет не только на последнюю цифру $z_n$, но сверх того и на цифры, стоящие на предыдущих позициях. Так, например, если $w_0$ равно $B + 5$, то последняя цифра произведения $z_0$ будет 5, а 1 должна быть добавлена к разряду $z_1$; это означает, что произошел перенос.

Понятие переноса, разумеется, тривиально, однако мы его специально сформулировали, поскольку рассмотрения как раз такого рода играют важную роль в новейших улучшениях этого базового метода умножения. В реальных применениях рассмотрение переносов может быть более утонченным, а для программистов и более сложным, чем все прочие шаги алгоритма.

## 9.1.2. Возведение в квадрат

С точки зрения вычислительных применений интересной оказывается связь между умножением и возведением в квадрат. Мы ожидаем, что операция возведения в квадрат $xx$ будет содержать больше излишних действий, чем умножение двух произвольных чисел $x, y$, поэтому возведение в квадрат должно быть проще умножения в общем случае. В самом деле, это подозрение оказывается правильным. Пусть запись числа $x$ в системе с основанием $B$ имеет $D$

цифр. Заметим, что равенство (9.1) можно переписать в случае возведения в квадрат в виде

$$w_n = \sum_{i=0}^{n} x_i x_{n-i}, \tag{9.2}$$

$n \in [0, D-1]$. Эта сумма имеет зеркальную симметрию, так что можно записать

$$w_n = 2 \sum_{i=0}^{\lfloor n/2 \rfloor} x_i x_{n-i} - \delta_n, \tag{9.3}$$

где $\delta_n$ равно 0 для нечетных $n$, и $x_{n/2}^2$ для четных. Ясно, что сумма $w_n$ каждого столбца требует примерно половину того количества умножений размера $B$, которое бы потребовались для умножения в общем случае. Разумеется, для того чтобы получить окончательные цифры произведения $z$, требуется применить операции переносов к суммам $w_n$; тем не менее, практически во всех приложениях этот метод возведения в квадрат в самом деле оказывается почти в два раза быстрее, чем умножение чисел большой разрядности. В литературе имеются очень хорошие описания этого метода и алгоритмов, связанных с ним. Смотрите, например, [Menezes et al. 1997].

Имеется один элегантный, хотя и простой, довод, показывающий, что умножение в общем случае не может иметь сложность, более чем в два раза превосходящую сложность возведения в квадрат. Рассмотрим равенство

$$4xy = (x+y)^2 - (x-y)^2, \tag{9.4}$$

показывающее, что умножение может быть сведено к двум возведениям в квадрат и делению на четыре, причем последнее деление можно считать не имеющим сложности (потому что его можно представить как сдвиг вправо на два бита). Это наблюдение не просто теоретическое, в некоторых практических случаях это правило вполне можно использовать (см. упражнение 9.6).

### 9.1.3. Операции div и mod

Операции div и mod присутствуют всюду в программах, связанных с простыми числами и факторизацией. Они также часто оказываются связаны с операцией умножения, и эта взаимосвязь используется в некоторых алгоритмах, приведенных ниже. Довольно частой оказывается ситуация, когда вычислительные мощности тратятся на такие операции, как вычисление $xy \pmod p$ для простых чисел $p$, или, в случае факторизации, на вычисление $xy \pmod N$, где $N$ — число, которое требуется разложить на множители.

Основным наблюдением является то, что операция mod деления с остатком тесно связана с операцией div. Как и ранее, мы будем обозначать через $x \bmod N$ операцию, дающую наименьший неотрицательный остаток от деления $x \pmod N$, а результатом операции div будет наибольшее целое число, не превосходящее $x/N$, которое также обозначается $\lfloor x/N \rfloor$. В некоторых языках программирования эти операции записываются как «$x\%N$» и «$x$ div $N$» соответственно, в других языках целочисленное деление «$x/N$» означает именно

div, в третьих div записывается как «Floor$[x/N]$», и т. д. Для целых $x$ и натуральных $N$ верно основное соотношение, которое в наших текущих обозначениях имеет вид:

$$x \bmod N = x - N\lfloor x/N \rfloor. \tag{9.5}$$

Заметим, что это соотношение равносильно представлению деления с остатком $x = qN + r$, где $q, r$ являются результатами операций div и mod соответственно. Таким образом, операция div порождает mod, и мы можем перейти к описанию алгоритма для div.

Аналогией к «школьному» методу умножения оказывается школьный метод деления длинных чисел. Крайне полезным оказывается рассмотреть даже этот простой метод, особенно с точки зрения возможных усовершенствований. В обычном методе деления длинных чисел в системе с основанием $B$ делитель $N$ сначала выравнивается влево относительно делимого $x$. Другими словами, находится такая степень основания $B^b$, что $m = B^b N \leqslant x < B^{b+1}N$. Затем находится $\lfloor x/m \rfloor$, причем результат деления обязательно будет лежать в отрезке $[1, B-1]$. Результат, конечно, станет старшей цифрой результата операции div. После этого $x$ заменяется на $x - m\lfloor x/m \rfloor$, $m$ делится на $B$ (то есть сдвигается вниз на один разряд), и вся операция повторяется. Это объяснение тотчас же подсказывает, что для определенных оснований $B$ все довольно просто. В самом деле, если выбрать двоичное представление ($B = 2$), то алгоритм вычисления div можно преобразовать так, чтобы он не содержал вообще ни одного умножения. Такой метод может представлять интерес для практических применений, особенно на машинах, имеющих сложение, вычитание, битовые сдвиги (сдвиг влево означает умножение на 2, сдвиг вправо означает деление на 2), и почти никаких других операций. Явно этот алгоритм выглядит так:

**Алгоритм 9.1.3** (классическое двоичное деление). Пусть заданы положительные целые числа $x \geqslant N$. Алгоритм выполняет операцию div, вычисляющую $\lfloor x/N \rfloor$, и возвращает результат. (Как при этом дополнительно вернуть значение $x \bmod N$, смотрите в упражнении 9.7.)

1. [Начальная установка]
   Найти то единственное целое число $b$, для которого $2^b N \leqslant x < 2^{b+1}N$;
   // Это может быть сделано последовательными сдвигами влево двоичного представления числа $N$, или, что лучше, сравнением длин двоичных записей чисел $x, N$ и, при необходимости, добавлением дополнительного сдвига
   $m = 2^b N$; $c = 0$;
2. [Перебрать все $b$ битов]
   for($0 \leqslant j \leqslant b$) {
   $c = 2c$;
   $a = x - m$;
   if($a \geqslant 0$) {
   $c = c + 1$;
   $x = a$;
   }

```
    m = m/2;
}
return c;
```

Похожий двоичный подход можно использовать, чтобы выполнить операцию умножения по модулю, т. е. $(xy) \bmod N$. Мы использовали результат [Arazi 1994]:

**Алгоритм 9.1.4** (двоичный алгоритм умножения и взятия остатка). **Пусть заданы положительные целые числа $x, y$, такие что $0 \leqslant x, y < N$. Этот алгоритм возвращает результат составной операции $(xy) \bmod N$. Мы предполагаем, что задано двоичное представление числа $x$ согласно определению 9.1.1, так что биты числа $x$ имеют вид $(x_0, \ldots, x_{D-1})$, и $x_{D-1} > 0$ — самый старший бит.**

1. [Начальная установка]
   $s = 0$;
2. [Перебрать все $D$ битов]
   for($D - 1 \geqslant j \geqslant 0$) {
   $s = 2s$;
   if($s \geqslant N$) $s = s - N$;
   if($x_j == 1$) $s = s + y$;
   if($s \geqslant N$) $s = s - N$;
   }
   return $s$;

Хотя алгоритмы двоичного деления и умножения по модулю достаточно наглядны, они обладают одним практическим недостатком: они не используют преимущества многобитовой арифметики, доступной на любом достаточно мощном компьютере. Хотелось бы выполнять операции умножения больших групп битов с помощью машинных регистров вместо того, чтобы работать с отдельными битами.

По этой причине обычно используются основания системы счисления $B$ большие, чем 2. Многие современные реализации операции div используют «Алгоритм D», см. [Knuth 1981, p. 257], представляющий собой выверенную версию алгоритма классического деления длинных чисел. Это хороший пример алгоритма, куда более сложного с точки зрения псевдокода, чем наш алгоритм двоичного деления (9.1.3), однако добивающегося существенной оптимизации в реальных условиях.

## 9.2. Усовершенствования в модулярной арифметике

Все классические алгоритмы div и mod, обсуждавшиеся в разд. 9.1.3, включают в себя операцию деления того или иного рода. Для двоичного случая деление было тривиальным; а именно, если $0 \leqslant a < 2b$, то число $\lfloor a/b \rfloor$ — либо 0, либо 1. В случае алгоритма D Кнута для оснований систем счисления,

больших 2, требуется уметь оценивать результаты деления малых чисел. Однако сейчас существуют алгоритмы, не требующие ни одного деления. Такие методы обладают двумя значительными преимуществами. Во-первых, можно писать полноценные вычислительные программы, не использующие довольно сложного метода деления длинных чисел; во-вторых, для оптимизации всех компьютерных вычислений достаточно сосредоточить усилия лишь на одной составляющей, а именно, на умножении.

### 9.2.1. Метод Монтгомери

Исследование Монтгомери [Montgomery 1985] оказалось очень важным для области приложений, особенно для ситуаций, где требовалось выполнить возведение в степень $(x^y) \bmod N$ по модулю $N$ с теоретически оптимальной скоростью (и, как мы увидим далее, где аргументы $x, y$ не колоссально велики). Отметим, во-первых, что «наивное» умножение по модулю требует одного умножения и одного деления (без учета вычитаний), и потому суть метода Монтгомери — как и всех остальных методов этого раздела — уменьшить, или, если повезет, устранить все сложности этапа деления.

Метод Монтгомери, являющийся обобщением метода Гензеля для вычисления обратных величин к 2-адическим числам, основывается на следующей теореме, которая ведет к оптимальному способу вычисления величины $(xR^{-1}) \bmod N$ для некоторого удобно выбранного $R$:

**Теорема 9.2.1** (Монтгомери). *Пусть $N, R$ — взаимно простые положительные целые числа, а $N' = (-N^{-1}) \bmod R$. Тогда для любого целого $x$ число*

$$y = x + N((xN') \bmod R)$$

*делится на $R$, причем*

$$y/R \equiv xR^{-1} \pmod{N}. \tag{9.6}$$

*Более того, если $0 \leqslant x < RN$, то разность $y/R - ((xR^{-1}) \bmod N)$ равна либо 0, либо $N$.*

Как мы увидим далее, теорема 9.2.1 будет наиболее полезной в случаях, когда требуется произвести много умножений по модулю $N$, как, например, в случае алгоритма возведения в степень, где вычисление вспомогательного числа $N'$ выполняется один раз на всю процедуру. В случаях, когда $N$ нечетно и $R$ является степенью 2, что довольно часто встречается на практике, операция «mod $R$» становится тривиальной, так же как и деление $y$ на $R$. Помимо этого, имеется другой способ вычисления $N'$, а именно, с использованием метода Ньютона. По этому поводу см. упражнение 9.12. В случае $N$ — нечетного и $R$, являющегося степенью 2, может оказаться полезным перевести базовую операцию Монтгомери на язык побитовых операций. Пусть $R = 2^s$, «&» обозначает операцию побитового «и», а через «$>> c$» обозначим операцию правого сдвига на $c$ битов. Тогда левая часть равенства (9.6) может быть записана в виде

$$y/R = (x + N * ((x * N')\&(R-1))) \ >> \ s, \tag{9.7}$$

где оба необходимых умножения отмечены явно.

Таким образом, для $0 \leqslant x < RN$ у нас есть способ вычислить $(xR^{-1}) \bmod N$ с небольшим числом умножений (а именно, с двумя). Конечно же, это не в точности результат применения операции $x \bmod N$, однако метод Монтгомери позволяет производить возведение в степень $(x^y) \bmod N$. Это связано с тем, что умножение системы вычетов $\{x : 0 \leqslant x < N\}$ на $R$ или на $R^{-1}$ дает снова полную систему вычетов по модулю $N$. Таким образом, возведение в степень можно проводить в другой системе вычетов, с одной начальной операцией умножения по модулю и последующими вызовами метода умножения Монтгомери, дающими результаты по модулю $N$. Для того чтобы сформулировать эти мысли строго, мы даем следующие определения:

**Определение 9.2.2.** Для чисел $R, N, x$, таких что НОД$(R, N) = 1$ и $0 \leqslant x < N$, назовем $(R, N)$-остатком числа $x$ величину $\overline{x} = (xR) \bmod N$.

**Определение 9.2.3.** Произведением Монтгомери двух целых чисел $a, b$ называется число $M(a, b) = (abR^{-1}) \bmod N$.

Тогда все необходимые факты можно собрать в следующей теореме:

**Теорема 9.2.4** (правила Монтгомери). *Пусть числа $R, N$ взаимно просты, и $0 \leqslant a, b < N$. Тогда $a \bmod N = M(\overline{a}, 1)$ и $M(\overline{a}, \overline{b}) = \overline{ab}$.*

Эта теорема приводит нас к методу Монтгомери возведения в степень. Например, непосредственным следствием этой теоремы является равенство

$$M(M(M(\overline{x}, \overline{x}), \overline{x}), 1) = x^3 \bmod N. \tag{9.8}$$

Для того чтобы сделать явной суть общего метода Монтгомери возведения в степень, мы далее приводим все существенные алгоритмы.

**Алгоритм 9.2.5** (произведение Монтгомери). Пусть заданы целые числа $0 \leqslant c, d < N$, где $N$ нечетно, а $R = 2^s > N$. Этот алгоритм возвращает $M(c, d)$.

1. [Функция $M$ Монтгомери]
   $M(c, d)$ {
   $x = cd$;
   $z = y/R$; // В соответствии с теоремой 9.2.1.
2. [Поправить результат]
   if($z \geqslant N$) $z = z - N$;
   return $z$;
   }

Шаг [Поправить результат] этого алгоритма всегда работает правильно, так как по предположению выполнено $cd < RN$. Число $R$ выбрано равным степени двойки только для того, чтобы при вычислении $z = y/R$ можно было использовать методы быстрой арифметики.

**Алгоритм 9.2.6** (метод Монтгомери возведения в степень). Пусть заданы числа $0 \leqslant x < N$, $y > 0$, и $R$ выбрано так же, как для алгоритма 9.2.5. Этот алгоритм возвращает $x^y \bmod N$. Через $(y_0, \ldots, y_{D-1})$ мы обозначаем двоичные биты $y$.

1. [Начальная установка]
   $\overline{x} = (xR) \bmod N$;

```
        // С помощью какого-либо метода деления с остатком.
    p̄ = R mod N;
        // С помощью какого-либо метода деления с остатком.
2. [Схема возведения в степень]
    for(D − 1 ⩾ j ⩾ 0) {
        p̄ = M(p̄, p̄);                    // С помощью алгоритма 9.2.5.
        if(y_j == 1) p̄ = M(p̄, x̄);
    }                                    // Теперь p̄ равняется x̄^y.
3. [Окончательное получение степени]
    return M(p̄, 1);
```

Далее в этой главе мы расскажем больше об общих схемах возведения в степень. В нашем примере схема использована в первую очередь для того, чтобы показать, как можно получить выгоду от вызова функции $M()$.

Все улучшения скорости окончательной процедуры возведения в степень связаны с использованием функции $M()$, в частности, с вычислением $z = y/R$. Мы отметили, что для вычисления $z$ нужно два умножения, как, например, в уравнении (9.7). Оказывается, это еще не все. Сложность операции mod Монтгомери может быть уменьшена (асимптотически, при больших $N$) до сложности операции умножения двух чисел размера $N$. (Другими словами, составная операция $M(x * y)$ асимптотически требует двух умножений размера $N$, что можно понимать как утверждение о том, что операция «*» требует одного такого умножения.) Технические подробности такого улучшения довольно запутаны и включают различные проявления циклов внутреннего умножения функции $M()$ (см. [Koç et al. 1996], [Bosselaers et al. 1994]). Частично эти улучшения основаны на потраченной зря операции в уравнении (9.7): дело в том, что сдвиг вправо уничтожает часть битов, порожденных двумя умножениями. Мы снова встретимся с этим явлением, связанным с использованием сдвигов, в следующем разделе. В программных реализациях метода Монтгомери можно выбрать основание системы счисления равным машинному слову $B = 2^b$, так что допустимо использование удобного значения $R = B^k$, откуда следует, что значение $z$ из алгоритма 9.2.5 может быть получено повторением $k$ раз одних и тех же операций по модулю $B$ — что особенно удобно для вычислений на компьютере. Использование циклов, явно ориентированных на размер машинного слова, которые достигают оптимальной асимптотической сложности, хорошо изложено в работе [Menezes et al. 1997].

## 9.2.2. Методы Ньютона

В разделе 9.1 мы видели, что операцию div можно выполнить с помощью сложений, вычитаний и битовых сдвигов, хотя, как мы тоже видели, алгоритм может быть улучшен с помощью перехода от двоичного представления к произвольным системам счисления. Затем было показано, что подход Монтгомери дает асимптотически оптимальные методы возведения в степень при фиксированном модуле. Интересно, и в первый раз даже, наверное, неожиданно, что методы div и mod в общем случае могут быть выражены через одни умножения;

это означает, что избегаются даже «малые» операции деления, сопровождавшие улучшенный метод деления, равно как и специальные предварительные вычисления из метода Монтгомери.

Один из подходов к вычислению div и mod заключается в том, чтобы осознать, что классический метод Ньютона решения уравнений применим к задаче поиска обратной величины. Начнем с задачи нахождения обратной величины в случае вещественных чисел. Пусть требуется решить уравнение $f(x) = 0$. Необходимо взять начальное приближение $x$, обозначим его $x_0$, и далее повторять операцию

$$x_{n+1} = x_n - f(x_n)/f'(x_n), \tag{9.9}$$

при $n = 0, 1, 2\ldots$, тогда если начальное приближение $x_0$ является достаточно хорошим, то последовательность $(x_n)$ будет сходиться к требуемому решению. Чтобы найти обратную величину к вещественному числу $a > 0$, надо попытаться решить уравнение $1/x - a = 0$, так что соответствующий итерационный процесс будет иметь вид:

$$x_{n+1} = 2x_n - ax_n^2. \tag{9.10}$$

В предположении, что эта последовательность приближений сходится (см. упражнение 9.13), мы видим, что вещественное число $1/a$ можно вычислить с любой заданной точностью с помощью одних умножений. Для того чтобы вычислить результат деления $b/a$, необходимо умножить $b$ на обратную к $a$ величину $1/a$; таким образом, деление вещественных чисел сводится к одним умножениям.

Но можно ли применить метод Ньютона к проблеме целочисленного деления? Оказывается, что можно, при условии, что мы внимательно изучим определение обобщенной обратной величины, необходимой для целочисленного деления. Введем функцию $B(N)$, определенную для неотрицательных целых чисел $N$ и равную числу битов в двоичном представлении числа $N$; за исключением случая $N = 0$, где положим по определению $B(0) = 0$. Таким образом, $B(1) = 1, B(2) = B(3) = 2$, и так далее. Определим обобщенную обратную величину. Вместо обратной величины $1/a$ для вещественного числа $a$ мы теперь будем рассматривать обобщенную обратную величину целого числа $N$, равную целой части результата деления некоторой большой степени двойки на $N$.

**Определение 9.2.7.** Обобщенной обратной величиной $R(N)$ для положительного целого $N$ называется величина $\lfloor 4^{B(N-1)}/N \rfloor$.

Показатель степени в определении выбран так, чтобы окончательный алгоритм деления работал правильно. Далее мы даем алгоритм для быстрого вычисления $R(N)$, основанный исключительно на умножениях, сложениях и вычитаниях:

**Алгоритм 9.2.8** (вычисление обобщенной обратной величины). **По заданному целому положительному числу $N$ этот алгоритм находит и возвращает обобщенную обратную величину $R(N)$.**

1. [Начальная установка]
   $b = B(N-1)$; $r = 2^b$; $s = r$;

2. [Итерация дискретного метода Ньютона]
   $r = 2r - \lfloor N\lfloor r^2/2^b\rfloor/2^b\rfloor$;
   if($r \leqslant s$) goto [Поправить результат];
   $s = r$;
   goto [Итерация дискретного метода Ньютона];
3. [Поправить результат]
   $y = 4^b - Nr$;
   while($y < 0$) {
   $r = r - 1$;
   $y = y + N$;
   }
   return $r$;

Обратите внимание, что алгоритм 9.2.8 включает в себя возможный этап «починки» окончательного значения возврата в виде цикла «while($y < 0$)». Это важнейший момент для того, чтобы сделать алгоритм работающим правильно, как это будет видно из доказательства следующей теоремы:

**Теорема 9.2.9** (итерационный процесс для поиска обобщенной обратной величины). *Алгоритм* 9.2.8 *поиска обобщенной обратной величины работает правильно; другими словами, возвращаемое им значение равно* $R(N)$.

Доказательство. Изначально

$$2^{b-1} < N \leqslant 2^b.$$

Положим $c = 4^b/N$ (где деление обычное), так что $R(N) = \lfloor c\rfloor$. Пусть

$$f(r) = 2r - \left\lfloor \frac{N}{2^b} \left\lfloor \frac{r^2}{2^b} \right\rfloor \right\rfloor,$$

и $g(r) = 2r - Nr^2/4^b = 2r - r^2/c$. Поскольку снятие целых частей в определении $f(r)$ дает нам функцию $g(r)$, и так как $N/2^b \leqslant 1$, мы имеем

$$g(r) \leqslant f(r) < g(r) + 2$$

для любого $r$.

Поскольку $g(r) = c - (c-r)^2/c$, имеем

$$c - (c-r)^2/c \leqslant f(r) < c - (c-r)^2/c + 2.$$

Мы заключаем, что $f(r) < c + 2$ для всех $r$. Далее, если $r < c$, то

$$f(r) \geqslant g(r) = 2r - r^2/c > r.$$

Таким образом, производимая алгоритмом итерационная последовательность $2^b, f(2^b), f(f(2^b)), \ldots$ строго возрастает до тех пор, пока не будет достигнуто такое значение $s$, что $c \leqslant s < c + 2$. Число $r$, приходящее на шаг [Поправить результат], равно $f(s)$. Если $c \geqslant 4$, мы также имеем $c \leqslant r < c + 2$. Но $c \geqslant 4$ всегда, за исключением тех случаев, когда $N = 1$ или 2. В последних случаях, а на самом деле каждый раз, когда $N$ является степенью 2, алгоритм завершается немедленно со значением $r = N$. Таким образом, алгоритм всегда завершает свою работу с числом $\lfloor c\rfloor$, что и утверждалось. □

Отметим, что число шагов итерационного процесса Ньютона в алгоритме 9.2.8 составляет $O(\ln(b+1)) = O(\ln\ln(N+2))$. Помимо этого, число шагов в цикле while шага [Поправить результат] не превосходит 2.

Вооруженные итерационным алгоритмом для поиска обобщенной обратной величины, мы можем перейти к операции mod, которая сама требует только умножений, сложений, вычитаний и битовых сдвигов.

**Алгоритм 9.2.10** (операция mod без деления). Этот алгоритм находит и возвращает $x \bmod N$ и $\lfloor x/N \rfloor$ для любого заданного неотрицательного целого числа $x$. Единственная величина, которую требуется вычислить заранее — обобщенная обратная величина $R = R(N)$. Ее можно найти с помощью алгоритма 9.2.8.

1. [Начальная установка]
   $s = 2(B(R) - 1)$;
   $div = 0$;
2. [Понижающий цикл]
   $d = \lfloor xR/2^s \rfloor$;
   $x = x - Nd$;
   if($x \geqslant N$) {
   $x = x - N$;
   $d = d + 1$;
   }
   $div = div + d$;
   if($x < N$) return $(x, div)$; // $x$ — результат mod, $div$ — результат div.
   goto [Понижающий цикл];

Этот алгоритм, по существу, является методом Барретта (см. [Barrett 1987]), хотя обычно его формулируют для часто встречающихся значений $x$, а именно, когда $0 \leqslant x < N^2$. Но мы сняли это ограничение с помощью рекурсивного использования основной формулы

$$x \bmod N \sim x - N\lfloor xR/2^s \rfloor, \tag{9.11}$$

где под «$\sim$» понимается, что при правильном выборе величины $s$ погрешность в этом равенстве является небольшим числом, кратным $N$. Имеется много способов улучшения алгоритма 9.2.10, так как мы выбрали особое число битов $s$, на которое будет выполняться сдвиг вправо. Имеются и другие интересные способы выбора $s$; в самом деле, было замечено (см. [Bosselaers et al. 1994]), что имеются определенные преимущества от «разделения» правых сдвигов таким образом:

$$x \bmod N \sim x - N\lfloor R\lfloor x/2^{b-1} \rfloor /2^{b+1} \rfloor, \tag{9.12}$$

где $b = B(R) - 1$. В частности, такое разделение может сделать выполняемые умножения в чем-то проще. В самом деле, видно, что

$$\left\lfloor R\left\lfloor x/2^{b-1} \right\rfloor /2^{b+1} \right\rfloor = \lfloor x/N \rfloor - j \tag{9.13}$$

для $j = 0, 1$ и 2. Таким образом, использование левой части равенства в качестве величины $d$ из алгоритма 9.2.10 требует не более двух проходов цикла while. Также имеется несомненная выгода во времени, так как длина $x$ может

быть равной примерно двум $b$, а длина $R$ — примерно $b$. Следовательно, умножение $xR$ в алгоритме 9.2.10 имеет размер примерно $2b \times b$ битов, в то время как умножение в формуле (9.12) имеет размер около $b \times b$ битов. Поскольку некоторому числу битов произведения $xR$ суждено быть сдвинутыми вправо и потерянными (сдвиг полностью стирает значение младших битов), можно попробовать вмешаться в цикл обычного школьного метода умножения, урезая ромб попарных произведений и оставляя меньшую табличку значений. С помощью подобной процедуры можно показать, что для $0 \leqslant x < N^2$ операция взятия модуля $x \bmod N$ имеет асимптотически ту же сложность, что и умножение размера $N$. С другой стороны, можно показать, что совместная операция $(xy) \bmod N$, где $0 \leqslant x, y < N$, имеет такую же сложность, что и два умножения размера $N$.

Были проведены исследования для классического метода деления длинных чисел (алгоритм D [Knuth 1981]), методов Монтгомери и Барретта [Bosselaers et al. 1994], [Montgomery 1985], [Arazi 1994], [Koç et al. 1996]. Похоже, что новым алгоритмам вычисления div и mod нет конца; к примеру, имеется метод оценки знака [Koç and Hung 1997], подходящий для криптографических операций (таких как возведение в степень) в случае, когда операнды велики. В то время как метод Монтгомери и (правильно достроенный) метод Барретта имеют асимптотически одну и ту же сложность, разные реализации этих методов обнаруживают разные области аргументов, в которых тот или иной метод оказывается лучше. Для криптографических приложений метод Монтгомери иногда оказывается слегка лучше метода Барретта. Одной из причин этого является то, что для метода Монтгомери достижение асимптотически наилучшей сложности дается проще, чем для метода Барретта. Улучшение последнего требует вмешательства в структуру цикла. Тем не менее, случаются и исключения. Так, например, [De Win et al. 1998] переняли метод Барретта для своих целей вероятно потому, что его легко реализовать и потому, что он почти равен по мощности методу Монтгомери. Также имеет значение, что нахождение обратных величин, необходимых в методе Монтгомери, может оказаться проблематичным в случае больших операндов. В случае, когда требуется найти *только результат операции mod* (а не результат долгого возведения в степень), метод Монтгомери противопоказан. Похоже, что очень хорошим выбором на случай обычной арифметики больших чисел является сочетание наших алгоритмов 9.2.8 и 9.2.10. Для факторизации, например, обычно требуется так много раз выполнить операцию $(xy) \bmod N$ для одного и того же $N$, что однократное вычисление обобщенной обратной величины $R(N)$ — все, что требуется, чтобы подготовиться к выполнению операций mod, не требующих деления — может оказаться очень полезным.

Мы вкратце упомянем некоторые новые идеи для алгоритмов для div и mod. Одна идея принадлежит Вольтману, который нашел способ улучшения алгоритма 9.2.10 Барретта для деления в сложном случае, когда $x$ во много раз больше, чем довольно маленькое число $N$. Одно из его улучшений заключается в том, чтобы менять точность вычисления в ходе алгоритма в таких случаях. Другое новое исследование дает интересный рекурсивный метод деления, схо-

жий с методом Карацубы, описанный в [Burnikel and Ziegler 1998]. Этот метод обладает интересным свойством, что сложности вычисления только div или только mod не совпадают.

Методы Ньютона можно применять не только для решения задачи деления. Важным примером может служить вычисление $\lfloor\sqrt{N}\rfloor$. В вещественном случае для поиска $\sqrt{a}$ можно применить итерационный алгоритм Ньютона, имеющий вид:

$$x_{n+1} = \frac{x_n}{2} + \frac{a}{2x_n}. \tag{9.14}$$

На основе него строится алгоритм для вычисления целочисленного квадратного корня:

**Алгоритм 9.2.11** (целая часть квадратного корня). Для положительного целого $N$ этот алгоритм находит и возвращает $\lfloor\sqrt{N}\rfloor$.

1. [Начальная установка]
   $x = 2^{\lceil B(N)/2\rceil}$;
2. [Шаг итерации]
   $y = \lfloor(x + \lfloor N/x\rfloor)/2\rfloor$;
   if($y \geqslant x$) return $x$;
   $x = y$;
   goto [Шаг итерации];

Можно использовать алгоритм 9.2.11 для проверки того, является ли положительное целое число $N$ полным квадратом или нет. После того как найдено число $x = \lfloor\sqrt{N}\rfloor$, остается сделать один шаг, чтобы узнать, выполнено ли $x^2 = N$. Это равенство выполнено тогда и только тогда, когда $N$ является полным квадратом. Конечно, есть и другие способы быстро узнать, является ли $N$ полным квадратом; например, проверить значения $(\frac{N}{p})$ для разных малых простых чисел $p$, или посмотреть на остаток от деления $N$ на 8.

Можно возразить, что алгоритм 9.2.11 требует для завершения $O(\ln\ln N)$ итераций. Имеется много интересных вопросов, связанных со сложностью этого и других методов Ньютона. Например, часто оказывается выгодно менять точность вычислений по мере того, как приближение становится ближе к искомому значению, или модифицировать саму процедуру цикла (см. упражнения 9.14 и 4.11).

### 9.2.3. Модули особого вида

Можно достигнуть существенной производительности в вычислении операции mod в случае, если модуль $N$ имеет особый вид. Метод Барретта из предыдущего раздела является быстрым именно потому, что использует арифметику вычислений по модулю $2^q$. В этом разделе мы покажем, что если модуль $N$ близок к степени двойки, то можно использовать двоичную природу современных компьютеров и проводить вычисления крайне эффективно. А именно, числа вида

$$N = 2^q + c,$$

где $|c|$ в каком-то смысле «мало» (и может быть отрицательным), допускают эффективные операции по модулю $N$. Эти улучшения особенно важны в теории простых чисел Мерсенна вида $p = 2^q - 1$ и чисел Ферма, имеющих вид $F_n = 2^{2^n} + 1$; хотя описанные методы применимы в равной степени ко всем модулям вида $2^q \pm 1$, для любого $q$. Это означает, что тот факт, является ли модуль $N$ простым и обладает ли прочими дополнительными свойствами, не влияет на алгоритм вычисления mod этого раздела. Главный результат следующий:

**Теорема 9.2.12** (вычисления по модулям особого вида). *Для целого числа $c$, положительного целого $q$, целого $x$ и $N = 2^q + c$ имеет место сравнение по модулю*

$$x \equiv (x \bmod 2^q) - c\lfloor x/2^q \rfloor \pmod N. \tag{9.15}$$

*Более того, в случае Мерсенна (когда $c = -1$), умножение на $2^k$ по модулю $N$ равносильно левому циклическому сдвигу на $k$ битов (если $k < 0$, то сдвиг делается в обратную сторону). В случае Ферма, когда $c = +1$, умножение на $2^k$ при положительном $k$ равносильно $(-1)^{\lfloor k/q \rfloor}$-кратному левому циклическому сдвигу на $k$ битов, за исключением того, что перешедшие биты должны быть инвертированы и поправлены с учетом переносов.*

Начнем с обсуждения последних утверждений теоремы, так как их проще всего анализировать. Поскольку

$$2^k = 2^{k \bmod q} 2^{q\lfloor k/q \rfloor},$$

а также $2^q \equiv -c \pmod N$, утверждения достаточно разобрать для случая $k \in [1, q-1]$ и таких же отрицательных $k$. В качестве примера возьмем $N = 2^{17} - 1 = 131071 = 1111111111111111_2$, $x = 8977 = 10001100010001_2$, и рассмотрим произведение $2^5 x \pmod N$. Это должен быть левый циклический сдвиг $x$ на 5 битов, или $110001000100010_2 = 25122$, что является правильным результатом. Кстати, этот способ умножения на степени 2 важен для теоретико-числовых преобразований и прочих алгоритмов. В частности, вычисление дискретного преобразования Фурье в кольце $\mathbf{Z}_n$, где $n = 2^m + 1$, можно проводить с помощью битовых сдвигов вместо умножений, в случае если первообразный корень является степенью двойки.

Первый результат теоремы 9.2.12 позволяет вычислять $x \bmod N$ исключительно быстро при условии, что $c$ мало. Давайте сначала приведем пример вычисления остатка от деления числа $x = 13000$ на простое число Мерсенна $N = 2^7 - 1 = 127$. Его необходимо преобразовать в двоичный вид: $13000 = 11001011001000_2$, и затем в соответствии с теоремой просто разбивать $x$ на две части всякий раз, когда оно превосходит $N$ (все числа здесь записаны с учетом модуля $N$):

$$\begin{aligned} x &\equiv 11001011001000 \bmod 10000000 + \lfloor 11001011001000/10000000 \rfloor \\ &\equiv 1001000 + 1100101 \equiv 10101101 \equiv 101101 + 1 \equiv 101110. \end{aligned}$$

Поскольку результат $101110_2 = 46 < N$, мы получаем, что искомое значение $13000 \bmod 127 = 46$. Процедура, таким образом, особенно проста в случае простых чисел Мерсенна, $N = 2^q - 1$. А именно, достаточно взять «верхние» биты

$x$ (то есть биты, начиная с $2^q$ и выше, включительно), и сложить их с «нижними» битами (то есть битами числа $x$, меньшими $q$). В общем случае процедура выполняется следующим образом, где мы для удобства использовали операции битового «и» «&», правого сдвига «>>» и левого сдвига «<<»:

**Алгоритм 9.2.13** (быстрое взятие остатка для модулей особого вида). **Пусть модуль $N$ имеет вид $N = 2^q + c$, где $B(|c|) < q$. Этот алгоритм возвращает $x \bmod N$ для целых $x > 0$. Метод обычно более эффективен при малых $|c|$.**

1. **[Произвести понижение]**
   **while**$(B(x) > q)$ {
   $y = x >> q$; // **Правый сдвиг выполняет вычисление** $\lfloor x/2^q \rfloor$.
   $x = x - (y << q)$; // **Либо** $x = x\&(2^q - 1)$, **либо** $x = x \bmod 2^q$.
   $x = x - cy$;
   }
   **if**$(x == 0)$ **return** $x$;
2. **[Поправить результат]**
   $s = \operatorname{sgn}(x)$; // **sgn равен** $-1, 0, 1$, **если** $x <, =, > 0$.
   $x = |x|$;
   **if**$(x \geqslant N)$ $x = x - N$;
   **if**$(s < 0)$ $x = N - x$;
   **return** $x$;

Несложно показать, что этот алгоритм завершает свою работу и выдает в качестве результата $x \bmod N$.

Поскольку метод не требует ничего, кроме «маленьких» умножений (на $c$), он может применяться очень широко. Недавнее обнаружение новых простых чисел Мерсенна использовало в ходе обширной проверки чисел с помощью теста Люка—Лемера этот метод взятия остатка. Имеется даже запатентованный метод шифрования, основанный на эллиптических кривых над полями $\mathbf{F}_{p^k}$, где $p = 2^q + c$, и, если требуется повышенная эффективность, $p \equiv -1 \pmod 4$ (например, $p$ может быть любым простым Мерсенна, или простым числом вида $2^q + 7$, и т.д.), где эллиптические операции выполняются на основе почти бесплатных операций mod (см. [Crandall 1994a]). Такие поля называются полями оптимального расширения, и дальнейшие улучшения возможно получить с помощью хитроумного выбора показателя $k$ и неприводимого многочлена для вычислений в поле $\mathbf{F}_{p^k}$. Также о таких эллиптических кривых известно, что порядок кривой можно быстрее установить с помощью быстрой операции mod. Еще одним приложением рассматриваемого понижающего алгоритма является факторизация чисел Ферма. Этот метод был использован недавно для обнаружения новых множителей $F_n$ для $n = 13, 15, 16, 18$ [Brent et al. 2000]. Для таких больших чисел Ферма время работы программы столь велико, что любые усовершенствования алгоритмов, неважно, операции mod или других операций, всегда приветствуются. В последнее время была установлена природа даже еще больших чисел $F_n$, несмотря на то, что тест простоты Пепена включает в себя огромное число операций $\pmod{F_n}$. Доказательство того, что $F_{22}$, $F_{24}$ являются составными, использовало деление с остатком по модулям особого вида,

описанное в этом разделе [Crandall et al. 1995], [Crandall et al. 1999], вместе с методом быстрого умножения, изложенном далее в этой главе.

Интересно, что методы вычислений по модулям особого вида можно обобщить еще дальше. Рассмотрим числа Прота:

$$N = k \cdot 2^q + c.$$

Сейчас мы дадим алгоритм быстрого взятия остатка из [Gallot 1999]. Этот метод подходит для случаев, когда параметры $k$ и $c$ являются параметрами низкой точности (т.е. умещаются в одно машинное слово):

**Алгоритм 9.2.14** (быстрая операция mod для модулей Прота). Пусть модуль $N$ имеет вид $N = k \cdot 2^q + c$, где длина двоичной записи $B(|c|) < q$ (при этом $c$ не запрещается быть отрицательным или нулем). Этот алгоритм возвращает значение $x \bmod N$ для $0 < x < N^2$. Метод обычно более эффективен при малых $k$ и $|c|$.

1. [Полезная функция увеличения сдвига $n$]
   $n(y)$ {
   return $Ny$; // Вычислять быстро по правилу: $Ny = ((ky) << q) + cy$.
   }
2. [Найти приближение к частному]
   $y = \lfloor \frac{x>>q}{k} \rfloor$;
   $t = n(y)$;
   if($c < 0$) goto [Противоположный метод];
   while($t > x$) {
   $t = n(y)$;
   $y = y - 1$;
   }
   return $x - t$;
3. [Противоположный метод]
   while($t \leqslant x$) {
   $y = y + 1$;
   $t = n(y)$;
   }
   $y = y - 1$;
   $t = n(y)$;
   return $x - t$;

Редукции такого типа сейчас используются в программах, которые дали такой замечательный успех в обнаружении, например, новых множителей чисел Ферма и в доказательстве простоты чисел Прота.

## 9.3. Возведение в степень

Возведение в степень особенно важно в курсах, связанных с простыми числами и факторизацией, по той простой причине, что очень много имеющихся теорем включают в себя операцию $x^y$, или, что чаще, $x^y \pmod N$. Ниже мы

приводим различные алгоритмы, которые успешно используют структуру показателя $y$, и иногда — структуру основания $x$. Мы уже мельком видели в алгоритме 2.1.5 из раздела 2.1.2 немаловажный факт: хотя и верно, что выражение вроде $(x^y) \bmod N$ можно вычислить за $(y-1)$ последовательных умножений по модулю $N$, имеется куда лучший способ возведения в степень. А именно, использовать то, что сейчас является обычным вычислительным методом — схему возведения в степень (powering ladder), которую можно понимать как нерекурсивную (или «развернутую») реализацию эквивалентного рекурсивного алгоритма. Но она способна и на большее, благодаря таким методам, как предварительная обработка битов показателя, использование представлений показателя в других системах счисления, и тому подобное. Давайте для начала перечислим имеющиеся типы схем возведения в степень:

(1) Рекурсивные схемы (алгоритм 2.1.5).
(2) Двоичные схемы, выполняющие вычисления слева направо или справа налево.
(3) Просмотровые схемы, которые работают с битовыми последовательностями битов или представлениями в других системах счисления. Простой пример такого подхода дает по сути троичный алгоритм 7.2.7, точнее, его шаг [Перебрать все биты ...]. Этот метод, тем не менее, допускает улучшения [Müller 1997], [De Win et al. 1998], [Crandall 1999b].
(4) Схемы для фиксированного $x$, вычисляющие $x^y$ для различных $y$ и фиксированного $x$.
(5) Цепи сложения и схемы Люка, такие как в алгоритме 3.6.7. Ссылки по этому вопросу: [Montgomery 1992b], [Müller 1998].
(6) Современные методы, основанные на сжатии потоков битов показателя степени, как это делается в работе [Yacobi 1999]

Этот раздел начинается с простых двоичных схем. Оказывается, что даже для них имеются различные подходы; затем мы перейдем к просмотровым схемам, схемам с другими основаниями и схемам для фиксированного $x$.

## 9.3.1. Простые двоичные схемы

Сейчас мы приведем два типа двоичных схем возведения в степень. Первый из них, алгоритм типа «слева направо» (равносильный алгоритму 2.1.5), сравним по сложности (за исключением случаев, когда аргументы находятся в особом соотношении между собой) со вторым, работающим наоборот, справа налево.

**Алгоритм 9.3.1** (двоичная схема возведения в степень типа «слева направо»). Этот алгоритм вычисляет $x^y$. Мы предполагаем, что задано двоичное представление $(y_0, \ldots, y_{D-1})$ числа $y > 0$, где $y_{D-1} = 1$ — самый старший бит.

1. [Начальная установка]
   $z = x$;
2. [Перебрать все биты числа $y$, начиная со следующего за старшим]
   for$(D-2 \geqslant j \geqslant 0)$ {

$z = z^2$;
// В случае модулярной арифметики здесь добавляется $\bmod N$.
if($y_j == 1$) $z = zx$;
// В случае модулярной арифметики выполнить $\bmod N$.
}
return $z$;

Этот алгоритм вычисляет $x^y$ с помощью перебора битов показателя $y$. В самом деле, количество возведений в квадрат равно $(D-1)$, а количество операций $z = z * x$ на один меньше, чем число битов, равных 1, в показателе степени $y$. Обратите внимание, что операции в алгоритме оказались теми же самыми, что и в алгоритме 2.1.5. Правило для запоминания того, которая из двух схем («слева направо» или «справа налево») равносильна рекурсивной форме, заключается в том, чтобы заметить, что оба алгоритма 9.3.1 и 2.1.5 используют умножение только на неизменный множитель $x$.

Имеется и в некотором смысле дополняющий способ возведения в степень. Его суть хорошо передает пример:

$$x^{13} = x * (x^2)^2 * (x^4)^2,$$

где снова получается 2 умножения и 3 возведения в степень (потому что $x^4$ на самом деле является тем же самым, что и второй член $(x^2)^2$). Действительно, в этом примере мы видим двоичное разложение показателя степени. Общая формула имеет вид

$$x^y = x^{\sum y_j 2^j} = x^{y_0}(x^2)^{y_1}(x^4)^{y_2}\ldots, \tag{9.16}$$

где $y_j$ обозначают биты $y$. Соответствующий алгоритм — это схема, работающая справа налево, в которой ведется учет результатов последовательных возведений в квадрат числа $x$:

**Алгоритм 9.3.2** (двоичная схема возведения в степень типа «справа налево»). Этот алгоритм вычисляет $x^y$. Мы предполагаем, что задано двоичное представление $(y_0, \ldots, y_{D-1})$ числа $y > 0$, где $y_{D-1} = 1$ — самый старший бит.

1. [Начальная установка]
   $z = x$; $a = 1$;
2. [Перебрать все биты $y$, начиная с самого младшего]
   for($0 \leqslant j < D-1$) {
   if($y_j == 1$) $a = za$; // Для модулярной арифметики $\bmod N$.
   $z = z^2$; // Для модулярной арифметики добавляется $\bmod N$.
   }
   return $az$; // Для модулярной арифметики добавляется $\bmod N$.

Как видно, этот метод включает в себя также $(D-1)$ возведений в квадрат, и, за исключением того случая, когда выполняется тривиальное умножение $a = z * 1$, содержит такое же число умножений, как и предыдущий алгоритм.

Несмотря на то, что число операций совпадает, определенное превосходство имеет первый алгоритм, так как в операции $z = zx$ содержится неизменный

сомножитель $x$. Так, например, если $x = 2$ или какое-либо другое маленькое число, что может иметь место в случае теста простоты, где малое число возводится в очень большую степень по модулю N, то умножение будет быстрым. На самом деле для $x = 2$ мы можем заменить умножение операцией сложения $z = z + z$, совершенно избежав умножений в этом шаге алгоритма. Такое преимущество наиболее заметно, когда двоичное представление показателя степени $y$ содержит много единиц.

Эти наблюдения ведут, в свою очередь, к вопросу об асимптотической сложности схем возведения в степень. Это притягательная — и во многом еще неизученная — область исследований. К счастью, на большинство вопросов о двоичных схемах возведения в степень ответы уже имеются. Давайте введем несколько поясняющих обозначений: пусть $S$ означает сложность возведения в квадрат (в соответствующем нашей задаче кольце или поле), а $M$ — сложность умножения. Очевидно, что сложность $C$ любой из приведенных схем асимптотически эквивалентна

$$C \sim (\lg y)S + HM,$$

где $H$ обозначает число единиц в двоичном представлении показателя степени $y$. Поскольку мы ожидаем, что выбранный наугад показатель степени будет содержать около половины единиц, то сложность вычислений в среднем случае равна

$$C \sim (\lg y)S + (\tfrac{1}{2}\lg y)M.$$

Обратите внимание, что использование равенства (9.4) позволяет сделать $S \sim M/2$, уменьшая тем самым выражение для сложности схем в среднем случае до $C \sim (\lg y)M$. Оценка $S \sim M/2$ не является абсолютно истинной. Во-первых, такая оценка предполагает, что вычисления не содержат модулярных вычислений, т. е. что используются только обычные сложения и умножения. Даже вне области модулярной арифметики встречаются проблемы. Например, при использовании умножения с помощью быстрого преобразования Фурье (БПФ) (для очень больших сомножителей это будет объяснено далее в этой главе), соотношение $S/M$ будет ближе к $2/3$. При других практических реализациях (например, при модулярной арифметике или при школьном методе), отношение $S/M$ примерно равно 0.8, как сообщается в [Cohen et al. 1998]. Какая бы процедура ни использовалась, конечно, желательно, чтобы требовалось выполнить меньше арифметических операций. Как мы увидим в следующем разделе, можно достичь еще большего сокращения количества операций.

## 9.3.2. Улучшения схем возведения в степень

В курсах, связанных с факторизацией и криптографией, имеется правило хорошего тона, требующее, чтобы возведение в степень занимало большую часть времени. При факторизации так называемая «фаза один» многих методов не требует ничего, кроме возведения в степень (в случае метода факторизации с помощью эллиптических кривых (ECM) роль возведения в степень играет эллиптическое умножение). В криптографии процедура создания публичных

ключей по секретным ключам включает в себя возведение в степень, так же как это происходит в случае цифровых подписей и т.д. Поэтому необходимо оптимизировать схемы возведения в степень настолько, насколько это возможно, ибо эти операции занимают почти все вычислительное время в соответствующих методах.

Один из интересных методов улучшения схем возведения в степень иногда называют «просматриванием» (windowing). Представьте, что мы рассматриваем не двоичное представление числа, а представление по основанию 4, и посчитали заранее степени $x^2$, $x^3$. Тогда каждый раз, когда мы сталкиваемся с *двумя* следующими битами показателя $y$, мы можем умножить текущее значение в регистре на одно из чисел $1 = x^0, x^1, x^2, x^3$, а затем возвести его в квадрат два раза, чтобы сдвинуть текущий регистр на два бита влево. Рассмотрим для примера задачу вычисления $x^{79}$, причем известно, что $79 = 1001111_2 = 1033_4$. Если мы запишем показатель степени $y$ по основанию 4, то это возведение в степень можно выполнить с помощью равенства

$$x^{79} = \left(x^{4^2} x^3\right)^4 x^3,$$

сложность которого равна $6S + 2M$ (напоминаем, что $S, M$ обозначают сложности возведения в квадрат и умножения соответственно). С другой стороны, алгоритм 9.3.1 схемы «слева направо» работал таким образом:

$$x^{79} = \left(\left(\left(x^{2^3} x\right)^2 x\right)^2 x\right)^2 x,$$

с итоговой сложностью $6S + 4M$ — большей, чем у метода с основанием 4. Мы не учли время, требуемое для предварительного вычисления величин $x^2, x^3$, так что выгода не так уж и очевидна. Но выгода была бы заметной в большинстве случаев, если бы показатель степени был больше, как это бывает в криптографических приложениях.

Имеется еще множество подробностей, которые мы пока не рассмотрели, но перед тем, как их затронуть, мы хотим дать более-менее общий алгоритм, содержащий большинство используемых идей:

**Алгоритм 9.3.3** (просмотровая схема возведения в степень). Этот алгоритм вычисляет $x^y$. Мы предполагаем, что задано представление $(y_0, \ldots, y_{D-1})$ показателя степени $y$ в системе счисления с основанием $B = 2^b$ (в смысле определения 9.1.1), где $y > 0$, старшая цифра $y_{D-1} \neq 0$, а каждая из цифр удовлетворяет условию $0 \leqslant y_i < B$. Мы также предполагаем, что значения $\{x^d : 1 < d < B;\ d \text{ нечетно}\}$ были посчитаны заранее.

1. [Начальная установка]
   $z = 1$;
2. [Перебрать все цифры]
   for($D - 1 \geqslant i \geqslant 0$) {;
   Представить $y_i$ в виде $y_i = 2^c d$, где $d$ — нечетное либо 0;
   $z = z(x^d)^{2^c}$; // $x^d$ берется из таблицы.

```
    if(i > 0) z = z^(2^b);
}
return z;
```

Для того чтобы дать пример, почему необходимо вычислить заранее только нечетные степени $x$, рассмотрим случай, когда $y = 262 = 406_8$. Глядя на это восьмеричное представление, мы видим, что

$$x^{262} = \left(\left(x^4\right)^8\right)^8 x^6,$$

но если степень $x^3$ была подсчитана заранее, то ее можно подставить в эту формулу в нужном месте, и возведение тогда, согласно алгоритму 9.3.3, будет проводиться следующим образом:

$$x^{262} = \left(\left(\left(x^4\right)^8\right)^4 x^3\right)^2.$$

Таким образом, предварительное вычисление остается делать только для нечетных степеней $x$. Еще один пример этого преимущества дает случай, когда основание системы счисления равно 16, тогда все три 4-битовых последовательностей 1100, 0110 и 0011 в записи $y$ обрабатываются с помощью одной и той же подсчитанной заранее степени $x^3$ и правильного выбора порядка возведения в квадрат.

Теперь перейдем к подробностям. Оказывается, можно разрешить «просмотровому окну» — по существу, основанию системы счисления $B$ — изменяться по мере того как выполняется процесс. Мы можем во время обработки показателя $y$ заглядывать вперед, пытаясь найти последовательности особого вида, чтобы еще немного увеличить производительность. Один метод «скользящего окна» представлен в работе [Menezes et al. 1997]. Также оказывается возможным использование сбалансированного представления по основанию $B$ (см. определение 9.1.2) для улучшения результата. Если мы потребуем, чтобы цифры показателя $y$ лежали в пределах

$$-\lfloor B/2 \rfloor \leqslant y_i \leqslant \lfloor (B-1)/2 \rfloor,$$

и подсчитаем заранее нечетные степени $x^d$, где $d$ лежит в тех же границах, что и цифры сбалансированного представления, то возникнет значительное преимущество при вычислении, при условии, конечно, что отрицательные степени допустимы. Пусть в случае эллиптического умножения требуется найти результат «возведения в степень» $[k]P$, где $P$ — точка кривой, а $k$ — показатель степени. Следовательно, достаточно подсчитать заранее только кратные точки

$$\{[d]P \ : \ 1 < d < \lfloor B/2 \rfloor;\ d \text{ — нечетное}\},$$

так как вычисление обратной величины $[-d]P$ выполняется мгновенно в соответствии с правилами эллиптической арифметики. Таким образом, можно создать эффективные «просмотровые» схемы возведения в степень для эллиптического умножения. Дальнейшее обсуждение смотрите в упражнении 9.77.

Можно показать, что требуемое для алгоритма 9.3.3 с основанием $B = 2^b$ асимптотическое (при больших $y$) количество операций, без учета предварительных вычислений, составляет $Db \sim \lg y$ возведений в квадрат, т. е. на каждый бит $y$ тратится одно возведение в квадрат. Это, конечно, не дает никакого превосходства в количестве операций по сравнению с простыми двоичными схемами. Разница, однако, заключается в количестве умножений. В то время как у простых двоичных схем асимптотическое количество умножений равняется числу единиц в представлении $y$, новые схемы требуют только по одному умножению на каждые $b$ битов. На самом деле, в среднем требуется $1 - 2^{-b}$ умножений на $b$ битов, из-за того что при случайном выборе $y$ последовательность битов может оказаться равной нулю. Следовательно, асимптотическая сложность «просмотрового» метода в среднем равна

$$C \sim (\lg y)S + (1 - 2^{-b})\frac{\lg y}{b}M,$$

что при $b = 1$ совпадает с предыдущей оценкой $C \sim (\lg y)S + (\frac{1}{2}\lg y)M$ для двоичных схем. Заметим, тем не менее, что по мере того как $b$ увеличивается, количество умножений становится все незначительнее. Верно, что предварительные вычисления требуют значительной части времени, однако на практике выбор $b = 3$ или $b = 4$ в самом деле значительно понижает время возведения в степень.

В соответствии с предыдущим замечанием о предварительных вычислениях, интересное улучшение возникает в случае, если число $x$ должно быть использовано повторно. Такое случается, если мы хотим вычислять $x^y$ для большого количества разных значений $y$ при фиксированном $x$. Мы можем вычислить и хранить неизменные степени фиксированного $x$, и использовать их для создания превосходства.

**Алгоритм 9.3.4** (возведение в степень $x^y$ при фиксированном основании $x$). Этот алгоритм производит возведение в степень $x^y$. Мы предполагаем, что задано представление показателя степени $y$ в системе счисления с основанием $B$ (не обязательно $B = 2$), имеющее вид $(y_0, \ldots, y_{D-1})$, где самая старшая цифра $y_{D-1} > 0$. Мы также предполагаем, что $(B-1)(D-1)$ значений

$$\{x^{iB^j} \ : i \in [1, B-1];\ j \in [1, D-1]\}$$

были подсчитаны заранее.

1. [Начальная установка]
   $z = 1$;
2. [Перебрать все цифры]
   for$(0 \leqslant j < D)$ $z = zx^{y_j B^j}$;
   return $z$;

Этот алгоритм, очевидно, требует числа операций

$$C \sim DM \sim \frac{\lg y}{\lg B}M,$$

(без учета предварительных вычислений), так что тот факт, что $x$ не меняется, действительно способен привести к высокой эффективности метода — благодаря множителю $(\lg B)^{-1}$. Для прочих конкретных условий и требований имеются и дальнейшие усовершенствования метода, включающие использование таблиц меньших размеров (т. е. хранение и использование значений только вида $x^{B^j}$), ослабление граничных условий в циклах for() в зависимости от значений, принимаемых цифрами $y$ (в некоторых ситуациях встречаться может не всякая цифра по основанию $B$), и т. д. Заметим, что если мы храним уменьшенный набор степеней $x^{B^j}$, то шаг [Перебрать все цифры] будет содержать вложенные циклы for(). Также имеются алгоритмы, работающие для фиксированного $y$, использующие для этого так называемые цепи сложения, так что возможны некоторые улучшения для случая, когда показатель степени фиксирован. Случаи, когда одновременно фиксированы $x$ и $y$, находят применения в криптографии. Например, если публичные ключи создаются как одно и то же значение $x$, возводимое в различные секретные степени $y$, улучшения, связанные с фиксированностью $x$, могут быть существенными. Подобным образом, если публичный ключ $x = g^h$ необходимо часто возводить в степень ключа $y$, то для получения дополнительной эффективности можно использовать методы, работающие при фиксированном $y$.

## 9.4. Улучшения для НОД и поиска обратного элемента

В разд. 2.1.1 мы изучили классические алгоритмы для функции НОД и поиска обратного элемента. В этом разделе мы рассмотрим более современные методы, особенно методы, применимые в случаях, когда рассматриваемые числа велики, либо когда определенные операции (такие как сдвиги) оказываются более эффективными.

### 9.4.1. Двоичные алгоритмы для НОД

Имеется одно несомненное улучшение алгоритма Евклида, обнаруженное Лемером в 1930-х годах. Его метод использует тут факт, что не каждое скрытое деление, содержащееся в алгоритме Евклида, требует вычислений с длинными числами, и с точки зрения статистики, в алгоритме должно использоваться много операций деления с малыми числами. Мы не приводим здесь метод Лемера (подробнее о методе см. в [Knuth 1981]), но отметим, что Лемер показал, как можно улучшить старый алгоритм для таких задач, как факторизация.

В 1960-х годах Сильвером и Терцианом [Knuth 1981], и независимо от них в [Stein 1967] было показано, что НОД можно вычислять с помощью определенного двоичного подхода. И в самом деле, следующие соотношения подсказывают изящное решение:

**Теорема 9.4.1** (Сильвер, Терциан и Штейн). *Для целых чисел $x, y$ имеют место утверждения*:

*Если оба числа $x, y$ — четные, то* $\text{НОД}(x,y) = 2\text{НОД}(x/2, y/2)$.

*Если число $x$ — четное, а $y$ — нечетное, то* $\text{НОД}(x,y) = \text{НОД}(x/2, y)$.

*(Согласно алгоритму Евклида)* $\text{НОД}(x,y) = \text{НОД}(x-y, y)$.

*Если оба числа $u, v$ — нечетные, то $|u-v|$ — четное и строго меньше* $\max\{u,v\}$.

Эти утверждения подводят нас к следующему алгоритму:

**Алгоритм 9.4.2** (двоичный алгоритм вычисления НОД). Следующий алгоритм возвращает наибольший общий делитель двух целых положительных чисел $x, y$. Для произвольного положительного целого $m$ обозначим через $v_2(m)$ количество идущих подряд нулей на самых младших позициях в двоичном представлении числа $m$. Это означает, что $2^{v_2(m)} \| m$. (Обратите внимание, что $m/2^{v_2(m)}$ — наибольший нечетный делитель числа $m$, и может быть получен из $m$ с помощью сдвига вправо всех нулей, стоящих на младших позициях; также заметим, что для удобства изложения мы могли бы положить $v_2(0) = \infty$.)

1. [Определить степень двойки, входящую в НОД]
   $\beta = \min\{v_2(x), v_2(y)\}$; // $2^\beta \| \text{НОД}(x,y)$
   $x = x/2^{v_2(x)}$;
   $y = y/2^{v_2(y)}$;
2. [Двоичный алгоритм для НОД]
   while($x \neq y$) {
   $(x,y) = (\min\{x,y\}, |y-x|/2^{v_2(|y-x|)})$;
   }
   return $2^\beta x$;

На большинстве существующих машин этот двоичный алгоритм часто оказывается быстрее алгоритма Евклида. Как мы ранее упоминали, улучшения Лемера применимы также и к этому двоичному методу.

Однако имеются другие, более современные улучшения. Вообще, похоже, что в печати продолжают появляться улучшения методов вычисления НОД. Имеется «$k$-ичный» алгоритм, принадлежащий Сёренсону, в котором выполняются понижения чисел по модулю $k > 2$. Также имеется новое расширение метода Сёренсона, которое, как утверждается, на типичном компьютере, обладающем операцией умножения, будет работать более чем в 5 раз быстрее, чем только что рассмотренный двоичный алгоритм для НОД [Weber 1995]. Метод Вебера довольно сложный, включает в себя несколько особых функций для нестандартного понижения по модулю, тем не менее, метод следует принимать всерьез в любом проекте, где вычисление НОД оказывается узким местом.

Совсем недавно Вебер [Weber 2005] построил новый модулярный алгоритм вычисления НОД, который мог бы быть наилучшим выбором для определенных областей значений операндов.

Интересно, что существуют такие вариации метода Сёренсона, для которых сложность вычисления НОД составляет $O(n^2/\ln n)$, в противоположность сложности алгоритма Евклида $O(n^2)$ [Sorenson 1994]. Помимо этого, имеется расширенная форма метода Сёренсона для вычисления не только НОД, но также и обратных величин.

Можно ли расширить этот эффективный двоичный подход таким же образом, как и классический алгоритм Евклида? Оказывается, такой «расширенный» двоичный алгоритм для НОД, вычисляющий заодно обратные величины, существует. Кнут [Knuth 1981] приписывает этот метод М. Пенку:

**Алгоритм 9.4.3** (двоичный алгоритм для НОД, вычисляющий также обратные величины). Пусть даны положительные целые числа $x, y$. Этот алгоритм возвращает тройку целых чисел $(a, b, g)$, такую что $ax + by = g = \text{НОД}(x, y)$. Мы предполагаем, что заданы двоичные представления $x, y$, и используем показатель $\beta$ из алгоритма 9.4.2.

1. [Начальная установка]
   $x = x/2^\beta; y = y/2^\beta;$
   $(a, b, h) = (1, 0, x);$
   $(v_1, v_2, v_3) = (y, 1 - x, y);$
   if($x$ — четное) $(t_1, t_2, t_3) = (1, 0, x);$
   else {
   $(t_1, t_2, t_3) = (0, -1, -y);$
   goto [Проверка на четность];
   }
2. [Поделить $t_3$ пополам]
   if(оба числа $t_1, t_2$ четные) $(t_1, t_2, t_3) = (t_1, t_2, t_3)/2;$
   else $(t_1, t_2, t_3) = (t_1 + y, t_2 - x, t_3)/2;$
3. [Проверка на четность]
   if($t_3$ even) goto [Поделить $t_3$ пополам];
4. [Произвести понижение]
   if($t_3 > 0$) $(a, b, h) = (t_1, t_2, t_3);$
   else $(v_1, v_2, v_3) = (y - t_1, -x - t_2, -t_3);$
5. [Вычесть]
   $(t_1, t_2, t_3) = (a, b, h) - (v_1, v_2, v_3);$
   if($t_1 < 0$) $(t_1, t_2) = (t_1 + y, t_2 - x)$
   if($t_3 \neq 0$) goto [Поделить $t_3$ пополам];
   return $(a, b, 2^\beta h);$

Как и двоичный алгоритм для НОД, этот алгоритм оказывается наиболее эффективным в реализациях на настоящих машинах. Если известно что-либо о характере операндов (например, если известно, что $y$ — простое), этот и подобные ему алгоритмы могут быть улучшены (см. упражнения).

## 9.4.2. Особые алгоритмы обращения

В течение последних лет были обнаружены варианты алгоритма НОД, находящие также обратную величину, в некоторых случаях зависящие от характера операндов $x, y$. Одним из примеров является схема обращения [Thomas et al. 1986], находящая $x^{-1} \bmod p$ для простых $p$. На самом деле алгоритм работает для любого модуля (возвращая либо правильную обратную величину, либо 0, если ее не существует), но авторы метода сосредоточили свои усилия на

простых модулях $p$, для которых главную в алгоритма величину $\lfloor p/z \rfloor$ можно вычислить легко.

**Алгоритм 9.4.4** (обращение по модулю). **Для модуля $p$ (обязательно простого) и числа $x \not\equiv 0 \pmod{p}$ этот алгоритм возвращает $x^{-1} \bmod p$.**

1. [Начальная установка]
   $z = x \bmod p$;
   $a = 1$;
2. [Цикл]
   while($z \neq 1$) {
   $q = -\lfloor p/z \rfloor$;
   // Алгоритм работает лучше всего, если эта операция выполняется быстро
   $z = p + qz$;
   $a = (qa) \bmod p$;
   }
   return $a$; // $a = x^{-1} \bmod p$.

Удобно то, что этот алгоритм легко реализовать. Помимо этого, для значений простых чисел, лежащих в определенных пределах, утверждается, что этот алгоритм работает насколько-то быстрее, чем расширенный алгоритм 2.1.4. Кстати, авторы алгоритма дают также интересный метод быстрого вычисления $\lfloor p/z \rfloor$ в случае, когда $p = 2^q - 1$, т. е. $p$ является простым числом Мерсенна. Может быть интересным вопрос исследования этого алгоритма, когда $p$ — не обязательно простое число. В частности, будет ли он обычно срабатывать и находить обратное, когда $p$ не имеет малых простых делителей?

Впрочем, и другие методы обращения используют тот особый случай, когда $p$ является простым числом Мерсенна. Следующий алгоритм является интересной попыткой использовать особый вид модуля:

**Алгоритм 9.4.5** (обращение по модулю простого числа Мерсенна). **Для простого числа $p = 2^q - 1$ и числа $x \not\equiv 0 \pmod{p}$ этот алгоритм возвращает обратную величину $x^{-1} \bmod p$.**

1. [Начальная установка]
   $(a, b, y, z) = (1, 0, x, p)$;
2. [Понижение]
   Найти число $e$, такое что $2^e \| y$;
   $y = y/2^e$; // Избавиться от младших нулей.
   $a = (2^{q-e} a) \bmod p$;
   // Циклический сдвиг, в соответствии с теоремой 9.2.12.
   if($y == 1$) return $a$;
   $(a, b, y, z) = (a + b, a, y + z, y)$;
   goto [Понижение];

### 9.4.3. Рекурсивные алгоритмы для НОД в случае очень больших операндов

Оказывается, что битовая сложность $O(\ln^2 N)$ вычисления НОД двух чисел, не превосходящих $N$, может быть по-настоящему уменьшена с помощью методов рекурсивного понижения, как впервые было показано в [Knuth 1971]. Позже было установлено, что подобный рекурсивный подход может привести к оценке сложности

$$O(M(\ln N)\ln\ln N),$$

где $M(b)$ обозначает битовую сложность умножения двух $b$-битовых целых чисел. В соответствии с наилучшей известной оценкой для $M(b)$, описанной далее в этой главе, сложность рекурсивных алгоритмов для НОД составляет

$$O\left(\ln N(\ln\ln N)^2\ln\ln\ln N\right).$$

Исследования рекурсивного подхода продолжались в течение нескольких десятилетий; ссылки на работы включают [Schönhage 1971], [Aho et al. 1974, pp. 300–310], [Bürgisser et al. 1997, p. 98], [Cesari 1998], [Stehlé and Zimmermann 2004]. Сейчас мы увидим, что как и для других ранее рассматривавшихся алгоритмов — таких как китайская теорема об остатках с предварительной обработкой данных — рекурсивный подход для НОД не может на самом деле использовать для ускорения школьный метод умножения.

Мы представим в этом разделе два рекурсивных алгоритма для поиска НОД — исходный алгоритм 1970-х гг., который мы для удобства назовем алгоритмом НОД Кнута—Шёнхаге (или НОДКШ), и очень новый, чисто двоичный алгоритм Штеле—Циммермана, называемый НОДШЦ. Оказывается, что оба варианта имеют одинаковую асимптотическую сложность, но существенно отличаются в способе реализации.

На практике оказывается, что рекурсивные методы поиска НОД работают быстрее любых других альтернатив (таких как двоичный алгоритм для НОД, как с улучшением Лемера, так и без него) в случае, когда входные аргументы $x, y$ существенно велики, например, в области десятков тысяч битов (хотя граница превосходства сильно зависит от оборудования и различных условий, таких как выбор другого метода вычисления НОД для малых значений аргументов). В качестве примера приложения, вспомним, что для безинверсионного метода факторизации с помощью эллиптических кривых, алгоритм 7.4.4, требуется НОД. Если попытаться найти множитель числа Ферма $F_{24}$ (пока это никому не удалось), аргументы НОД будут иметь длину порядка 16 миллионов битов; это область, где рекурсивные методы поиска НОД с указанной выше сложностью совершенно превосходят по скорости все возможные альтернативы. Позже в этом разделе мы приведем конкретные оценки времени работы.

Основная идея метода НОДКШ заключается в том, что последовательности частных и остатков классического алгоритма НОД в корне отличаются в следующем смысле. Пусть числа $x, y$ не превосходят $N$. В соответствии с алгоритмом Евклида 2.1.2, обозначим через $(r_j, r_{j+1})$ для $j \geqslant 0$ пары чисел,

появляющиеся после $j$ проходов цикла. Последовательность остатков определяется как $(r_0 = x,\ r_1 = y,\ r_2, r_3, \ldots)$. Подобным образом имеется неявная последовательность частных $(q_1, q_2, \ldots)$, определяемая равенствами

$$r_j = q_{j+1} r_{j+1} + r_{j+2}.$$

При исполнении классический алгоритм применяет это равенство до тех пор, пока некоторое $r_k$ не станет равно 0, в случае чего предыдущий остаток $r_{k-1}$ является НОД. Что касается коренного отличия последовательностей $q$ и $r$: как было элегантно показано в [Cesari 1998], можно ожидать, что *полное* число битов в последовательности остатков составляет $O(\ln^2 N)$ и потому очевидно, что любой алгоритм для НОД, обращающийся ко всем остаткам $r_j$, имеет в лучшем случае квадратичную сложность. С другой стороны, последовательность частных $(q_1, \ldots, q_{k-1})$ обычно состоит из относительно небольших элементов. Рекурсивный подход основывается на том факте, что знание величин $q_j$ дает любой остаток $r_j$ за почти линейное время [Cesari 1998].

Попробуем рассмотреть пример последовательностей частных и остатков. (Мы выбрали входные параметры $(x, y)$ средней величины для иллюстрации рекурсивной идеи.) Положим

$$(r_0, r_1) = (x, y) = (31416, 27183),$$

откуда

$$\begin{aligned}
r_0 &= q_1 r_1 + r_2 = 1 \cdot r_1 + 4233,\\
r_1 &= q_2 r_2 + r_3 = 6 \cdot r_2 + 1785,\\
r_2 &= q_3 r_3 + r_4 = 2 \cdot r_3 + 663,\\
r_3 &= q_4 r_4 + r_5 = 2 \cdot r_4 + 459,\\
r_4 &= q_5 r_5 + r_6 = 1 \cdot r_5 + 204,\\
r_5 &= q_6 r_6 + r_7 = 2 \cdot r_6 + 51,\\
r_6 &= q_7 r_7 + r_8 = 4 \cdot r_7 + 0.
\end{aligned}$$

Очевидно, $\text{НОД}(x, y) = r_7 = 51$, однако заметим, что последовательность частных имеет вид $(1, 6, 2, 2, 1, 2, 4)$; на самом деле, это элементы простой цепной дроби рационального числа $x/y$. Такое явление типично: можно ожидать, что большинство элементов последовательности частных будут малы.

Чтобы формализовать процедуру, позволяющую получить последовательность остатков по известной последовательности частных, мы можем использовать векторно-матричное соотношение, верное для $i < j$:

$$\begin{pmatrix} r_j \\ r_{j+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -q_j \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -q_{i+1} \end{pmatrix} \begin{pmatrix} r_i \\ r_{i+1} \end{pmatrix}.$$

Идея заключается в том, чтобы использовать обычно малые значения $q$, чтобы вычислить такую матрицу $G$, чтобы вектор $G(x, y)^T$ совпадал со столбцом $(r_j, r_{j+1})^T$, причем *битовая длина $r_j$ примерно равна половине $x$.* Затем нужно повторно применять этот подход до тех пор, пока не станет возможно работать с операндами быстро с помощью классического алгоритма НОД. В

следующем алгоритме при вызове главной функции $rgcd()$ рано или поздно (в шаге [Уменьшить аргументы]) произойдет вызов процедуры $hgcd()$, изменяющей матрицу $G$ так, чтобы результирующее произведение $G(u,v)^T$ являлось столбцом со значительно меньшими компонентами. В нашем простом примере, если мы остановимся посередине пути, будет верно

$$G = \begin{pmatrix}0 & 1\\ 1 & -2\end{pmatrix}\begin{pmatrix}0 & 1\\ 1 & -2\end{pmatrix}\begin{pmatrix}0 & 1\\ 1 & -6\end{pmatrix}\begin{pmatrix}0 & 1\\ 1 & -1\end{pmatrix} = \begin{pmatrix}12 & -15\\ -32 & 37\end{pmatrix}$$

и

$$G\begin{pmatrix}r_0\\ r_1\end{pmatrix} = \begin{pmatrix}12 & -15\\ -32 & 37\end{pmatrix}\begin{pmatrix}31416\\ 27183\end{pmatrix} = \begin{pmatrix}663\\ 459\end{pmatrix} = \begin{pmatrix}r_4\\ r_5\end{pmatrix}.$$

Таким образом, мы существенно перемещаемся дальше по цепочке остатков с помощью лишь одного вызова процедуры $hgcd()$. В нашем конкретном примере, мы могли бы затем перейти к классическому алгоритму НОД с меньшими операндами $r_4$ и $r_5$. Для очень больших начальных операндов потребовалось бы несколько рекурсивных проходов, чтобы существенно переместиться по цепочке остатков, где битовая длина $r_j$ становится примерно в два раза меньше после каждого прохода.

Для следующей формулировки алгоритма мы извлекли реализацию из [Buhler 1991]. (Замечание: как и во многих современных программных продуктах, мы полагаем $\text{НОД}(0,0) = 0$ для удобства.)

**Алгоритм 9.4.6** (рекурсивное вычисление НОД). Пусть заданы неотрицательные целые числа $x, y$. Этот алгоритм возвращает $\text{НОД}(x,y)$. Функция верхнего уровня $rgcd()$ вызывает рекурсивную функцию $hgcd()$, которая в свою очередь вызывает функцию «малых» значений $shgcd()$, с классической функцией $cgcd()$ (такой как алгоритм Евклида или двоичный алгоритм), вызываемой в основании рекурсии. Имеется глобальная переменная — матрица $G$, остальные же переменные являются локальными (в обычном для рекурсивных функций смысле)

1 [Начальная установка]
$lim = 2^{256}$,
// Порог для $cgcd()$, настраивается для получения наибольшей эффективности
$prec = 32$,
// Граничная длина для $shgcd()$, настраивается для получения наибольшей эффективности

2 [Функция $shgcd$, вычисляющая НОД для маленьких аргументов, должна возвращать матрицу]
$shgcd(x,y)$ {
// Функция для короткого НОД, локальные переменные: $u, v, q, A$
$A = \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$,
$(u,v) = (x,y)$,
**while**$(v^2 > x)$ {
$q = \lfloor u/v \rfloor$,
$(u,v) = (v, u \bmod v)$,

$$A = \begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix} A,$$

}
return $A$,
}

3 [Рекурсивная процедура $hgcd$ должна изменять глобальную матрицу $G$]
$hgcd(b, x, y)$ { // Локальные переменные: $u, v, q, m, C$
if($y == 0$) return,
$u = \lfloor x/2^b \rfloor$,
$v = \lfloor y/2^b \rfloor$,
$m = B(u)$, // $B$, как обычно — функция битовой длины.
if($m < prec$) {
$G = shgcd(u, v)$,
return,
}
$m = \lfloor m/2 \rfloor$,
$hgcd(m, u, v)$; // Рекурсия
$(u, v)^T = G(u, v)^T$, // Умножение матрицы на столбец.
if($u < 0$) $(u, G_{11}, G_{12}) = (-u, -G_{11}, -G_{12})$,
if($v < 0$) $(v, G_{21}, G_{22}) = (-v, -G_{21}, -G_{22})$,
if($u < v$) $(u, v, G_{11}, G_{12}, G_{21}, G_{22}) = (v, u, G_{21}, G_{22}, G_{11}, G_{12})$,
if($v \neq 0$) {
$(u, v) = (v, u)$,
$q = \lfloor v/u \rfloor$,
$G = \begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix} G$, // Матричное умножение.
$v = v - qu$;
$m = \lfloor m/2 \rfloor$,
$C = G$,
$hgcd(m.\ u, v)$, // Рекурсия.
$G = GC$,
}
return,
}

4 [Определить функцию верхнего уровня $rgcd$]
$rgcd(x, y)$ { // Функция верхнего уровня, локальные переменные. $u, v$.
$(u, v) = (x, y)$,

5 [Уменьшить аргументы]
$(u, v) = (|u|, |v|)$, // Взять абсолютные значения чисел.
if($u < v$) $(u, v) = (v, u)$,
if($v < lim$) goto [Ветвление],
$G = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$,
$hgcd(0, u, v)$,
$(u, v)^T = G(u, v)^T$;

$(u, v) = (|u|, |v|)$;
**if**$(u < v)$ $(u, v) = (v, u)$;
**if**$(v < lim)$ **goto** [Ветвление];
$(u, v) = (v, u \bmod v)$;
**goto** [Уменьшить аргументы];

6. [Ветвление]
**return** $cgcd(u, v)$;
// Рекурсия выполнена, выбираем другой метод вычисления НОД.

}

Поясним, как работает этот алгоритм. Требуется выбрать пороговые значения $lim$ и $prec$, при этом наибольший общий делитель чисел $x, y$ вычисляется основной функцией $rgcd(x, y)$. Отметим, что Вольтману удалось реализовать алгоритм 9.4.6 очень экономным с точки зрения использования памяти способом, а именно, с помощью повторного использования определенных участков памяти и проведения тщательного учета ресурсов. Он в 2000 г. сообщил о возможности вычислять произвольный НОД с числом Ферма $F_{24}$, менее чем за час на современном компьютере, в то время как классический алгоритм НОД такого порядка потребовал бы нескольких *дней* машинного времени. Это был один из первых практических успехов рекурсивного подхода в свое время. Таким образом, хотя алгоритм и является сложным и запутанным, он несомненно имеет свои преимущества, особенно при поиске делителей очень больших чисел, например, таких больших, как некоторые интересные «чисто составные» числа, такие как число Ферма $F_{20}$ и выше.

Другой рекурсивный подход — НОДШЦ — является очень новым результатом. Это двоично-рекурсивный алгоритм для НОД, не требующий почти ничего, кроме двоичных сдвигов и умножений больших чисел. Это впечатляющее открытие имеет ту же теоретическую оценку сложности, что и алгоритм 9.4.6, однако [Zimmerman 2004] сообщает, что реализация алгоритма, приведенного далее, в пакете GNU MP арифметических операций высокой точности, находит НОД двух чисел длины $2^{24}$ битов в каждом примерно за 45 секунд на современном компьютере. Время работы алгоритма 9.4.6 в 2000 г. уменьшилось благодаря современному (2004) оборудованию до нескольких минут, так что НОДШЦ вполне эффективен. Не будем забывать, однако, что теоретическая оценка сложности, как было упомянуто в начале раздела, применима к обоим алгоритмам — использование простых, быстрых двоичных операций в новом алгоритме приводит к меньшей эффективной константе в большом $O$. (Имеется также замечание [Stehlé and Zimmermann 2004], что намного проще доказывать строгие оценки сложности для алгоритма НОДШЦ.)

Основная идея алгоритма НОДШЦ заключается в том, чтобы разложить рациональное число в цепную дробь, элементы которой берутся не из обычных положительных чисел, а из множества

$$(\pm 1/2, \pm 1/4, \pm 3/4, \pm 1/8, \pm 3/8, \pm 5/8, \pm 7/8, \pm 1/16, \pm 3/16, \ldots).$$

Типичное разложение дроби таким методом поясняется на примере рациональ-

ного числа $525/266$ (пример, опубликованный Бернштейном):

$$\begin{aligned} 525 &= (1/2)266 + 392; \\ 266 &= (-3/4)392 + 560; \\ 392 &= (-1/2)560 + 672; \\ 560 &= (-1/2)672 + 896; \\ 672 &= (3/4)896 + 0. \end{aligned}$$

Теперь видно, что НОД$(525, 266)$ является нечетной частью числа 896, а именно, 7. На каждом шаге мы выбираем такое дробное «частное», чтобы степень двойки в остатке увеличивалась. Таким образом, приведенный далее алгоритм полностью двоичный, и потому особенно подходит для машин с быстрыми двоичными операциями, такими как векторные сдвиги и т.п. Заметим, что процедура *divbin* в алгоритме 9.4.7 является просто одним шагом того типа, что использовался выше, и что он всегда применяется, когда первое число не делится на такую большую степень 2, на которую делится второе число.

Следуя Штеле—Циммерману, мы используем модулярную редукцию со знаком $x \bmod m$, определяемую как единственный остаток $x$ по модулю $m$, лежащий на отрезке $[-\lfloor m/2 \rfloor + 1, \lfloor m/2 \rfloor]$. Функция $v_2$, возвращающая число завершающих нулевых битов, вычисляется по алгоритму 9.4.2. Как и в предыдущих алгоритмах, через $B(n)$ обозначаем число битов в двоичном представлении неотрицательного числа $n$.

**Алгоритм 9.4.7** (двоично-рекурсивный алгоритм НОД Штеле—Циммермана). Этот алгоритм по заданным неотрицательным $x, y$ вычисляет НОД$(x, y)$. Функция верхнего уровня $SZgcd()$ вызывает рекурсивную полудвоичную функцию $hbingcd()$, использующую классический двоичный алгоритм НОД, когда операнды будут существенно понижены.

1. [Начальная установка]
   $thresh = 10000$; // Настраиваемый порог перехода к двоичному НОД.
2. [Определение функции верхнего уровня, возвращающей НОД]
   $SZgcd(x, y)$ { // Переменные $u, v, k, q, r, G$ — локальные.
   $(u, v) = (x, y)$;
   if$(v_2(v) < v_2(u))$ $(u, v) = (v, u)$;
   if$(v_2(v) == v_2(u))$ $(u, v) = (u, u + v)$;
   if$(v == 0)$ return $u$;
   $k = v_2(u)$;
   $(u, v) = (u/2^k, v/2^k)$;
3. [Понижение]
   if($(B(u)$ или $B(v) < thresh)$ return $2^k$НОД$(u, v)$; // Алгоритм 9.4.2.
   $G = hbingcd(\lfloor B(u)/2 \rfloor, u, v)$; // $G$ — матрица 2x2.
   $(u, v)^T = G(u, v)^T$; // Умножение матрицы на столбец.
   if$(v == 0)$ return $2^{k - v_2(u)}u$;
   $(q, r) = divbin(u, v)$;
   $(u, v) = (v/2^{v_2(v)}, r/2^{v_2(v)})$;

```
    if(v == 0) return 2^k u;
    goto [Понижение];
  }
```

4. [Полудвоичная функция деления]

```
  divbin(x, y) {      // Возвращается столбец высоты 2. q, r — локальные.
    r = x;
    q = 0;
    while(v_2(r) ⩽ v_2(y)) {
      q = q − 2^{v_2(r)−v_2(x)};
      r = r − 2^{v_2(r)−v_2(y)} y;
    }
    q = q cmod 2^{v_2(y)−v_2(x)+1};
    r = x + qy/2^{v_2(y)−v_2(x)};
    return (q, r);
  }
```

5. [Полудвоичная функция НОД (рекурсивная)]

```
  hbingcd(k, x, y) {                          // Возвращает матрицу;
                          // G, u, v, k_1, k_2, k_3, q, r — локальные.
    G = ( 1  0 )
        ( 0  1 );
    if(v_2(y) > k) return G;
    k_1 = ⌊k/2⌋;
    k_3 = 2^{2k_1+1};
    u = x mod k_3;
    v = y mod k_3;
    G = hbingcd(k_1, u, v);                   // Рекурсивный вызов.
    (u, v)^T = G(x, y)^T;
    k_2 = k − v_2(v);
    if(k_2 < 0) return G;
    (q, r) = divbin(u, v);
    k_3 = 2^{v_2(v)−v_2(u)};
    G = ( 0    k_3 )
        ( k_3  q   ) G;
    k_3 = 2^{2k_2+1};
    u = (v2^{−v_2(v)}) mod k_3;
    v = (r2^{−v_2(v)}) mod k_3;
    G = hbingcd(k_2, u, v)G;
    return G;
  }
```

В последней части упражнения 9.20 содержится некая скрытая подсказка по поводу того, какие операции алгоритма 9.4.7 важны для хорошей скорости выполнения.

В дополнение отметим, что для того, чтобы получить такую удивительно низкую сложность обоих рекурсивных алгоритмов для НОД, обязательно тре-

буется иметь эффективный алгоритм умножения больших чисел. Независимо от того, используется ли умножение в произведении матриц или где-либо еще, обязательно применение пороговых методов. Это значит, что для малых операндов используется школьный метод умножения, для больших можно применять метод Карацубы или Тоома—Кука, но для очень больших значений операндов нужно использовать оптимальные методы, основанные на быстром преобразовании Фурье. Другими словами, при реализации требуется серьезно подходить к сложности умножения $M(N)$, появлявшейся в формуле сложности в начале этого раздела. Эти различные методы быстрого умножения обсуждаются далее в этой главе (разделы 9.5.1 и 9.5.2).

Логично задаться вопросом, существуют ли расширенные формы таких рекурсивных алгоритмов НОД, аналогичных, например, алгоритмам 2.1.4 и 9.4.3, позволяющих асимптотически быстро вычислять обращение по модулю. Ответ на этот вопрос положительный, как рассказывается в [Stehlé and Zimmermann 2004] и [Cesari 1998].

## 9.5. Умножение больших чисел

Для случая, когда числа содержат сотни тысяч (или даже миллионы) десятичных знаков, используются особые способы умножения. На практике оказывается, что классический «школьный» метод умножения просто не применим к числам определенного размера. Это, конечно, связано с тем, что битовая сложность «школьного» метода умножения двух целых чисел, не превосходящих $N$, составляет $O\left(\ln^2 N\right)$. Оказывается, что с помощью современных преобразований сигналов и вычисления сверток сложность можно понизить до величины

$$O(\ln N(\ln\ln N)(\ln\ln\ln N)),$$

что мы обсудим более подробно далее в этом разделе.

Методы арифметики больших чисел пересматривались много раз, особенно в последнее время. Как и в случае с литературой, посвященной быстрому преобразованию Фурье (здесь и далее — БПФ), похоже, что не имеет конца поток публикаций новых подходов, новых оптимизаций и новых приложений вычислительной теории чисел. Достаточно ясно, что в этом разделе мы представим, скорее, обзор, чем энциклопедический учет всех методов этой богатой и необычной области. Интересный с теоретической точки зрения обзор методов умножения содержится в работе [Bernstein 1997], а современные исследования обсуждаются (и на них приводятся ссылки) в работе [Crandall 1994b, 1996a].

### 9.5.1. Методы Карацубы и Тоома—Кука

Как было показано Карацубой, для того, чтобы ускорить умножение больших чисел, к их частям можно применять классические методы умножения. В его рекурсивном алгоритме предполагается, что числа представляются в разделенной форме

$$x = x_0 + x_1 W,$$

где $x_0, x_1 \in [0, W-1]$. Это равносильно представлению по основанию системы счисления $W$, где $W$ берется примерно равным корню из $x$. Число $x$, таким образом, не превосходит $W^2$. Соотношение Карацубы для двух целых чисел $x, y$ такого размера имеет вид:

$$xy = \frac{t+u}{2} - v + \frac{t-u}{2}W + vW^2, \tag{9.17}$$

где

$$t = (x_0 + x_1)(y_0 + y_1),$$
$$u = (x_0 - x_1)(y_0 - y_1),$$
$$v = x_1 y_1,$$

и мы получаем число $xy$, являвшееся изначально умножением размера $W^2$, по цене трех умножений размера $W$ (и какого-то количества переносов, обеспечивающего, что результат имеет вид представления числа по основанию $W$). В этом и заключается основное превосходство, поскольку умножение с помощью школьного метода было бы в четыре, а не в три раза дороже умножения размера $W$. Можно показать, что если попробовать применить соотношение Карацубы к самим величинам $t, u, v$, и таким же образом действовать дальше, то асимптотическая сложность умножения чисел размера $N$ составит

$$O\left((\ln N)^{\ln 3/\ln 2}\right)$$

битовых операций, что является теоретическим улучшением по сравнению со школьным методом. Мы говорим «теоретическое», поскольку у реализаций метода будут возникать проблемы накладных расходов, и то время, которое требуется для организации хранения переменных в памяти, перегруппировки промежуточных результатов и т. д., может помешать методу Карацубы стать жизнеспособной альтернативой. Тем не менее, часто оказывается, что метод Карацубы в самом деле обгоняет по скорости школьный метод при определенном размере чисел; вид этой области зависит от машины и реализации метода.

Похожий метод, а именно, метод Тоома—Кука, достигает теоретически минимальной границы в $O\left(\ln^{1+\epsilon} N\right)$ битовых операций для мультипликативной части умножения размера $N$ — т. е. без учета всех сложений, входящих в метод. Тем не менее, имеются причины, почему этот метод не является последним словом в искусстве умножения больших чисел. Во-первых, для больших $N$ число сложений становится значимым. Во-вторых, эта оценка сложности предполагает, что умножение на константы (такие как $1/2$, что является, по сути, двоичным сдвигом) является несложной операцией. Для небольших констант это действительно так, но коэффициенты Тоома—Кука сильно растут по мере того как увеличивается $N$. Тем не менее, метод представляет теоретический интерес и имеет свои практические приложения, такие как быстрое умножение на машинах, обычное умножение которых особенно медлительно по сравнению со сложением. Метод Тоома—Кука использует идею, что если заданы два многочлена

$$x(t) = x_0 + x_1 t + \ldots + x_{D-1} t^{D-1}, \tag{9.18}$$

$$y(t) = y_0 + y_1 t + \quad + y_{D-1} t^{D-1}, \tag{9 19}$$

то их произведение $z(t) = x(t)y(t)$ полностью определяется своими значениями в $2D-1$ точках $t$, например, последовательностью значений $(z(\jmath))$, $\jmath \in [1-D, D-1]$

**Алгоритм 9.5.1** (символическое умножение Тоома—Кука). Пусть задано $D$ Этот алгоритм порождает (символическую) схему Тоома—Кука умножения двух целых чисел, имеющих $D$ цифр в системе счисления с основанием $B$

1 [Начальная установка]
Создать два символических многочлена $x(t), y(t)$ степени $(D-1)$, как в уравнениях (9 18) и (9 19),

2 [Вычисление]
Подсчитать символически $z(\jmath) = x(\jmath)y(\jmath)$ для каждого $\jmath \in [1-D, D-1]$, так чтобы каждая величина $z(\jmath)$ была выражена через коэффициенты исходных многочленов $x$ и $y$,

3 [Восстановление]
Решить символически следующую систему из $(2D-1)$ линейных уравнений относительно переменных $z_\jmath$
$z(t) = \sum_{k=0}^{2D-2} z_k t^k$, $t \in [1-D, D-1]$,

4 [Выписать схему]
Выписать список из $(2D-1)$ соотношений, каждое из которых выражает $z_\jmath$ через исходные коэффициенты $x, y$,

Результатом работы этого алгоритма будет набор формул, выражающих коэффициенты произведения многочленов $z(t) = x(t)y(t)$ через коэффициенты исходных многочленов. Но это в точности то, что понимается под целочисленным умножением, если каждый из многочленов соответствует представлению числа, имеющему $D$ цифр в системе счисления с основанием $B$

Для того чтобы подчеркнуть идею метода Тоома—Кука, отметим, что все умножения в алгоритме 9 5.1 происходят в шаге [Вычисление]. Сейчас мы дадим особый метод умножения, который требует пять таких умножений. Мы использовали предыдущий, символический алгоритм, чтобы найти соотношения следующего алгоритма

**Алгоритм 9.5.2** (явное умножение Тоома—Кука для $D = 3$). Пусть заданы целые числа $x, y$ в системе счисления с основанием $B$ в виде

$$x = x_0 + x_1 B + x_2 B^2,$$
$$y = y_0 + y_1 B + y_2 B^2$$

Этот алгоритм возвращает цифры $B$-ичного представления результата умножения $z = xy$, используя теоретически минимально возможное количество $2D-1 = 5$ умножений для вычисления ациклической свертки последовательностей длины 3

1 [Начальная установка]
$r_0 = x_0 - 2x_1 + 4x_2$,
$r_1 = x_0 - x_1 + x_2$,

$r_2 = x_0$;
$r_3 = x_0 + x_1 + x_2$;
$r_4 = x_0 + 2x_1 + 4x_2$;
$s_0 = y_0 - 2y_1 + 4y_2$;
$s_1 = y_0 - y_1 + y_2$;
$s_2 = y_0$;
$s_3 = y_0 + y_1 + y_2$;
$s_4 = y_0 + 2y_1 + 4y_2$;

2. [Умножения Тоома—Кука]
for($0 \leqslant j < 5$) $t_j = r_j s_j$;

3. [Восстановление]
$z_0 = t_2$;
$z_1 = t_0/12 - 2t_1/3 + 2t_3/3 - t_4/12$;
$z_2 = -t_0/24 + 2t_1/3 - 5t_2/4 + 2t_3/3 - t_4/24$;
$z_3 = -t_0/12 + t_1/6 - t_3/6 + t_4/12$;
$z_4 = t_0/24 - t_1/6 + t_2/4 - t_3/6 + t_4/24$;

4. [Выполнение переносов]
carry = 0;
for($0 \leqslant n < 5$) {
$v = z_n +$ carry;
$z_n = v \bmod B$;
$carry = \lfloor v/B \rfloor$;
}
return $(z_0, z_1, z_2, z_3, z_4, carry)$;

Теперь по сравнению с методом Карацубы, в котором умножение размера $B^2$ сводится к трем умножениям размера $B$, что можно назвать выгодой в $4/3$ раза, алгоритм 9.5.2 выполняет умножение размера $B^3$ с помощью пяти умножений размера $B$, т. е. с выгодой в $9/5$ раз. Когда любой из этих алгоритмов используется рекурсивно (например, если шаг [Умножения Тоома—Кука] выполняется с помощью рекурсивного вызова того же самого или другого метода Тоома—Кука), сложность умножения двух чисел размера $N$ снижается до

$$O\left((\ln N)^{\ln(2D-1)/\ln D}\right)$$

умножений малых чисел (т. е. чисел размера, не зависящего от $N$), что при достаточно высокой степени $d = D-1$ метода Тоома—Кука может быть сделано ниже любой заранее заданной оценки сложности вида $O\left(\ln^{1+\epsilon} N\right)$ умножений малых чисел. Однако необходимо подчеркнуть, что эта оценка сложности не учитывает количества сложений, а также умножений на постоянные величины (см. упражнения 9.37 и 9.78, а также раздел 9.5.8).

Метод Тоома—Кука можно понимать как схему для вычисления ациклической свертки, которую мы рассмотрим далее в этой главе вместе с другими типами сверток. Более подробно о методах Карацубы и Тоома—Кука можно прочитать в книгах [Knuth 1981], [Crandall 1996a], [Bernstein 1997].

### 9.5.2. Алгоритмы вычисления преобразования Фурье

После того как мы обсудили методы умножения, имеющие такую низкую сложность, как $O\left(\ln^{1+\epsilon} N\right)$ умножений малых чисел фиксированного размера (но, возможно, большое количество сложений), мы рассмотрим класс методов умножения, имеющих малое количество операций всех типов. Эти методы основаны на понятии дискретного преобразования Фурье (ДПФ), которое мы сейчас рассмотрим достаточно подробно, чтобы объяснить последующие алгоритмы умножения.

Сейчас мы можем понимать под «сигналом» просто последовательность элементов, это нам нужно, чтобы объяснить связь между теорией преобразований Фурье и теорией обработки сигналов. В течение всей оставшейся главы сигналы могут быть последовательностями коэффициентов многочленов или последовательностями в общем случае, и будут обозначаться $x = (x_n)$, $n \in [0, D-1]$, для некоторой «длины сигнала» $D$.

Первая существенная концепция, которая нам понадобится, заключается в том, что умножение является в определенном смысле сверткой. Мы сделаем эту связь более явной позже, отметив сейчас лишь то, что ДПФ является естественным преобразованием, которое используется для решения проблем вычисления сверток, поскольку ДПФ имеет уникальное свойство преобразовывать свертки в более дешевое покомпонентное произведение.

**Определение 9.5.3** (дискретное преобразование Фурье (ДПФ)). Пусть $x$ — сигнал длины $D$, состоящий из элементов некоторого кольца, где существует $D^{-1}$, и пусть $g$ является примитивным корнем степени $D$ в этом кольце; это означает, что $g^k = 1$ тогда и только тогда, когда $k \equiv 0 \pmod{D}$. Тогда дискретным преобразованием Фурье сигнала $x$ является сигнал $X = \text{ДПФ}(x)$, элементы которого имеют вид

$$X_k = \sum_{j=0}^{D-1} x_j g^{-jk}, \tag{9.20}$$

и обратное преобразование Фурье $\text{ДПФ}^{-1}(X) = x$ задается равенствами

$$x_j = \frac{1}{D} \sum_{k=0}^{D-1} X_k g^{jk}. \tag{9.21}$$

Утверждение о том, что обратное преобразование Фурье $\text{ДПФ}^{-1}$ является корректно определенным и обратным преобразованием к преобразованию Фурье, оставлено в качестве упражнения. Имеется несколько важных видов ДПФ:

Комплексное ДПФ: $x, X \in \mathbf{C}^D$, $g$ — примитивный корень степени $D$ из 1, такой как $e^{2\pi\imath/D}$;

ДПФ в конечном поле: $x, X \in \mathbf{F}_{p^k}^D$, $g$ — примитивный корень степени $D$ из 1 в том же поле;

ДПФ в целочисленном кольце: $x, X \in \mathbf{Z}_N^D$, $g$ — примитивный корень степени $D$ из 1 в этом кольце, и существуют $D^{-1}$, $g^{-1}$.

Следует обратить внимание на то, что перечисленные примеры общеизвестны, но возможны также и другие. В качестве еще одного примера можно определить ДПФ над квадратичными полями (см. упражнение 9.50).

В комплексном случае все практические реализации используют операции с плавающей точкой для работы с комплексными числами (хотя в случаях, когда сигнал имеет только вещественную компоненту, как мы увидим, возможна существенная экономия). В случае конечного поля вся арифметика проводится по модулю $p$. В третьем случае преобразование ДПФ в целочисленном кольце иногда применяют одновременно к $N = 2^n - 1$ и $N' = 2^n + 1$, в случае чего можно выполнить присваивания $g = 2$ и $D = n$, $D' = 2n$, если $n$ взаимно просто с каждым из чисел $N, N'$.

Следует сказать, что существует целое семейство альтернативных преобразований, многие из которых основываются на базисных функциях, отличных от сложных функций возведения в степень привычного метода ДПФ. Часто такие альтернативные подходы допускают применение быстрых алгоритмов, либо предполагают использование вещественных сигналов и т.п. Хотя такие преобразования лежат вне области рассмотрения этой книги, мы отметим, что часть из них также подходит для вычисления свертки, так что мы назовем некоторые: преобразование Уолша—Адамара, не требующее умножений, а только сложений; дискретное преобразование косинусов (DCT), которое является аналогом ДПФ для случая вещественного сигнала; всевозможные волновые преобразования, которые иногда допускают очень быстрые реализации ($O(N)$ вместо $O(N \ln N)$); вещественнозначное быстрое преобразование Фурье, которое использует функции sin или cos и суммирует только вещественные числа; преобразование Хартли для вещественных сигналов и т. д. Многие из этих методов обсуждаются в работе [Crandall 1994b, 1996a].

Для того чтобы устранить недоразумения, мы сейчас сделаем явным почти тривиальное различие между дискретным преобразованием Фурье (ДПФ) и знаменитым быстрым преобразованием Фурье (БПФ). БПФ — это алгоритм, принадлежащий классу методов типа «разделяй и властвуй», который вычисляет ДПФ из определения 9.5.3. БПФ будет часто появляться в наших алгоритмах в виде $X = \text{БПФ}(x)$, где понимается, что вычисляется тем самым ДПФ. Подобным образом, операция $\text{БПФ}^{-1}(x)$ выполняет обратное дискретное преобразование Фурье. Мы делаем различие явным, поскольку «БПФ» является в некотором смысле неправильным названием: ДПФ — это определенная сумма, т. е. алгебраическая величина, но БПФ — это алгоритм. Имеется поясняющая аналогия с этим различием: в этой книге классы эквивалентности $x \pmod N$ являются теоретическими сущностями, в то время как операцию взятия остатка от деления $x$ на $p$ мы договорились писать немного по-другому, в виде $x \bmod p$. Таким же образом, внутри алгоритмов выражение $X = \text{БПФ}(x)$ обозначает, что мы применяем алгоритм БПФ к сигналу $x$; возвращает этот алгоритм, несомненно, значение $\text{ДПФ}(x)$. (Еще одна причина, по которой мы

подчеркиваем это почти тривиальное различие, заключается в том, что некоторые студенты считают БПФ в каком-то смысле «приближением» к ДПФ, в то время как на самом деле БПФ в некоторых случаях является более точным, чем буквальное сложение в соответствии с суммой ДПФ, потому что погрешность округления становится меньше, в основном из-за меньшего числа операций в БПФ).

Авторство основной идеи алгоритма БПФ может быть прослежено вплоть до некоторых рассмотрений Гаусса, тем не менее некоторые авторы приписывают рождение современной теории равенству Даниэльсона—Ланцоша, которое применимо, когда длина сигнала $D$ четная:

$$\text{ДПФ}(x) = \sum_{j=0}^{D-1} x_j g^{-jk} = \sum_{j=0}^{D/2-1} x_{2j} \left(g^2\right)^{-jk} + g^{-k} \sum_{j=0}^{D/2-1} x_{2j+1} \left(g^2\right)^{-jk}. \quad (9.22)$$

И в самом деле, красивое равенство: сумма ДПФ для сигнала длины $D$ разделяется на две суммы, каждая длины $D/2$. Таким образом, равенство Даниэльсона—Ланцоша порождает рекурсивный метод вычисления преобразования Фурье. Обратите внимание на так называемые «поворачивающие» сомножители $g^{-k}$, которые естественно вписываются в следующую рекурсивную форму алгоритма БПФ. В этом и последующем текстах алгоритмов мы обозначаем через $len(x)$ длину сигнала $x$. Помимо этого, когда мы выполняем конкатенации вида $(a_j)_{j \in J}$, мы имеем в виду, что результат будет конкатенацией, выполненной в естественном порядке, слева направо. Подобным образом, через $U \cup V$ мы обозначаем сигнал, имеющий элементы сигнала $V$ присоединенными справа к элементам сигнала $U$.

**Алгоритм 9.5.4** (БПФ, рекурсивная форма). Пусть задан сигнал $x$ длины $D = 2^d$, для которого существует ДПФ согласно определению 9.5.3. Этот алгоритм вычисляет это преобразование с помощью одного вызова БПФ$(x)$. Мы используем функцию длины сигнала $len()$, и внутри рекурсии корень из единицы $g$ должен иметь порядок, равный текущей длине сигнала.

1. [Рекурсивная функция БПФ]
   БПФ$(x)$ {
   $n = len(x)$;
   if$(n == 1)$ return $x$;
   $m = n/2$;
   $X = (x_{2j})_{j=0}^{m-1}$; // Четная часть $x$.
   $Y = (x_{2j+1})_{j=0}^{m-1}$; // Нечетная часть $x$.
   $X =$ БПФ$(X)$;
   $Y =$ БПФ$(Y)$; // Два рекурсивных вызова половинной длины.
   $U = (X_{k \bmod m})_{k=0}^{n-1}$;
   $V = (g^{-k} Y_{k \bmod m})_{k=0}^{n-1}$; // Использовать корень $g$ порядка $n$.
   return $U + V$; // Использование равенства (9.22).
   }

Небольшой подсчет показывает, что количество операций в рассматривае-

мом кольце составляет

$$O(D \ln D),$$

и эта оценка верна как для умножений, так и для сложений/вычитаний. Сложность $D \ln D$ типична для алгоритмов типа «разделяй и властвуй», другими примерами которых являются некоторые алгоритмы быстрой сортировки. Рекурсивная форма алгоритма удобна тем, что прямо указывает последовательность действий; она имеет свою область применения, но подавляющее большинство реализаций БПФ используют хитрую структуру для цикла, впервые полученную в [Cooley and Tukey 1965]. Алгоритм Кули—Тьюки основывается на том, что если к элементам исходного сигнала $x$ длины $D = 2^d$ применить определенную перестановку битов, то БПФ можно вычислить с помощью простых вложенных циклов. Перестановка заключается в обратной двоичной переиндексации, что означает, что $x_j$ заменяется на $x_k$, где $k$ является перевернутой двоичной записью числа $j$. Например, если длина сигнала $D = 2^5$, то новый элемент $x_5$ после перестановки будет равен старому элементу $x_{20}$, потому что обратной записью к $5 = 00101_2$ является $10100_2 = 20$. Обратите внимание, что эту перестановку индексов можно было бы в принципе провести исключительно с помощью манипуляций с индексами, порождающими новый переиндексированный массив. Однако часто более эффективным оказывается выполнить перестановку прямо на месте, используя определенную последовательность двухэлементных перестановок. Именно последний метод используется в последующем алгоритме.

Самое важное замечание заключается в том, что метод Кули—Тьюки в самом деле позволяет выполнить БПФ на месте, т. е. заменить исходный сигнал $x$, элемент за элементом, на значения ДПФ. Это необычайно эффективный с точки зрения экономии памяти метод, так что он заслуживает внимания, особенно учитывая огромную популярность схемы Кули—Тьюки и подобных ей методов. После того, как перестановка битов выполнена правильно, сам алгоритм приводит к правильно расположенному результату БПФ. Исторически фраза «прореживание по времени» приписывается методу Кули—Тьюки и означает, что как и в разделяющем равенстве Даниэльсона—Ланцоша (9.22), мы разделяем область времени — индексы исходного сигнала. Метод БПФ Джентльмена—Сейнда подпадает под категорию «прореживание по частоте», где похожие операции выполняются над индексами $k$ результата преобразования $X_k$.

**Алгоритм 9.5.5** (БПФ, сохраняет результат на том же месте и в том же порядке, использует битовую перестановку). Пусть задан сигнал $x$ длины $D = 2^d$. Приведенные функции вычисляют БПФ с помощью вложенных циклов. Представлено два существенных алгоритма БПФ: в форме алгоритма с «прореживанием по времени» (Кули—Тьюки) и с «прореживанием по частоте» (Джентльмена—Сейнда). Отметим, что эти алгоритмы могут применяться не только численно, но и символически, и для теоретико-числовых преобразований, если правильно определены корень из единицы и операции в рассматриваемом кольце.

1. [БПФ Кули—Тьюки, с прореживанием по времени]

```
БПФ(x) {
    Перестановка(x);
    n = len(x);
    for(m = 1; m < n; m = 2m) {          // m пробегает степени двойки.
        for(0 ⩽ j < m) {
            a = g^{-jn/(2m)};
            for(i = j; i < n; i = i + 2m)
                (x_i, x_{i+m}) = (x_i + a x_{i+m}, x_i - a x_{i+m});
        }
    }
    return x;
}
```

2. [БПФ Джентльмена—Сейнда, с прореживанием по частоте]

```
БПФ(x) {
    n = len(x);
    for(m = n/2; m ⩾ 1; m = m/2) {
        // m пробегает степени двойки сверху вниз.
        for(0 ⩽ j < m) {
            a = g^{-jn/(2m)};
            for(i = j; i < n; i = i + 2m)
                (x_i, x_{i+m}) = (x_i + x_{i+m}, a(x_i - x_{i+m}));
        }
    }
    Перестановка(x);
    return x;
}
```

3. [Описание процедуры перестановки]

```
Перестановка(x) {                    // Обратная двоичная переиндексация.
    n = len(x);
    j = 0;
    for(0 ⩽ i < n - 1) {
        if(i < j) (x_i, x_j) = (x_j, x_i);        // Поменять элементы местами.
        k = ⌊n/2⌋;
        while(k ⩽ j) {
            j = j - k;
            k = ⌊k/2⌋;
        }
        j = j + k;
    }
return;
}
```

Следует отметить, что когда свертка вычисляется так, как мы это будем делать далее, операции перестановки битов не нужны; для этого необходимо выполнять БПФ в определенном порядке. Порядок должен быть таким: сна-

чала выполняется алгоритм Джентльмена—Сейнда (без последнего шага перестановки битов), затем метод Кули—Тьюки (без первого шага перестановки битов). Это работает, конечно же, потому, что операция перестановки битов имеет порядок два.

К счастью, для случаев, когда операция перестановки не желательна, или когда важен последовательный доступ к памяти (как, например, на векторных компьютерах), имеется метод Стокхема для БПФ, который избегает перестановки битов, а также самый внутренний цикл которого по сути перебирает ячейки памяти последовательно. В расплату за это приходится иметь еще одну копию входного массива. Типичные реализации метода Стокхема элегантны [Van Loan 1992], однако имеется одна реализация, оказавшаяся довольно полезной на современных векторных компьютерах. Эта реализация называется «пинг-понговым» БПФ, потому что ему приходится постоянно ходить туда-сюда между исходными данными и отдельной копией. Следующий алгоритм основан на решении, предложенном [Papadopoulos 1999].

**Алгоритм 9.5.6** (БПФ, «пинг-понговый» вариант, сохраняет порядок, не требует перестановки битов). Пусть задан сигнал $x$ длины $D = 2^d$. Алгоритм работает с исходным сигналом $x$ и дополнительной копией $y$. Под $X, Y$ мы понимаем указатели на составные сигналы $x$ и $y$ соответственно, работающие по обычным законам арифметики указателей; так, например, $X[0]$ — изначально это первый элемент составной структуры $x$, но если к $X$ добавить 4, то будет $X[0] = x_4$, и т.п. Если показатель степени $d$ четный, то по окончании работы алгоритма указатель $X$ содержит результат работы БПФ, в противном случае он содержится в $Y$.

1. [Начальная установка]
   $J = 1$;
   $X = x$; $Y = y$; // Установить указатели.
2. [Внешний цикл]
   for($d \geqslant i > 0$) {
      $m = 0$;
      while($m < D/2$) {
         $a = e^{-2\pi\imath m/D}$;
         for($J \geqslant j > 0$) {
            $Y[0] = X[0] + X[D/2]$;
            $Y[J] = a(X[0] - X[D/2])$;
            $X = X + 1$;
            $Y = Y + 1$;
         }
         $Y = Y + J$;
         $m = m + J$;
      }
      $J = 2 * J$;
      $X = X - D/2$; $Y = Y - D$;
      $(X, Y) = (Y, X)$; // Поменять указатели местами.
   }

3. [Проверка четности для пинг-понга]
   if($d$ четно) return (массив, начинающийся с $X$);
   return (массив, начинающийся с $Y$);

Полезной особенностью цикла в этом алгоритме является то, что переменная цикла $j$ изменяется последовательно (от $J$ и вниз), так что на векторной машине данные можно обрабатывать целыми группами, сначала взяв их, потом положив обратно, как массив данных.

Кстати, выполнение обратного БПФ оказывается чрезвычайно простым, если прямой метод вычисления БПФ уже реализован. Один подход заключается в том, чтобы просто посмотреть на определение 9.5.3 и обнаружить, что замена корня $g$ на $g^{-1}$ и нормализующее умножение в конце вычислений на $1/D$ приводят к обратному преобразованию Фурье. В случае комплексных чисел $g^{-1} = g^*$, и процедуру вычисления $\text{БПФ}^{-1}$ можно представить в виде последовательности операций:

$x = x^*$; // Взять сопряженный сигнал.
$X = \text{БПФ}(x)$; // Обычный алгоритм БПФ с обычным корнем $g$.
$X = X^*/D$; // Еще одно сопряжение и нормализация.

Несмотря на то, что методы Кули—Тьюки и Джентльмена—Сейнда чаще всего применяются к комплекснозначным сигналам, так что корень можно выбрать, например, равным $g = e^{2\pi i/D}$, их также можно использовать для вычисления теоретико-числовых преобразований, где все операции должны проводиться над конечным кольцом или полем. И в комплексном, и в конечном случае общим является то, что исходный сигнал имеет только вещественную компоненту. Такой сигнал мы называем «чисто вещественным». Для комплекснозначного сигнала $x \in C^D$ это будет так, если $x_j = a_j + 0i$ для каждого $j \in [0, D-1]$. Важно отметить, что аналогичный класс сигналов может встречаться в некоторых полях, например, в поле $\mathbf{F}_{p^2}$, где $p \equiv 3 \pmod 4$, поскольку в таком поле каждый элемент может быть представлен в виде $x_j = a_j + b_j i$, и мы можем назвать сигнал $x \in \mathbf{F}_{p^2}^D$ чисто вещественным, тогда и только тогда, когда все $b_j$ равны нулю. Смысл введения класса чисто вещественных сигналов тот, что обычно алгоритмы БПФ для класса чисто вещественных сигналов требуют примерно в два раза меньшей сложности, чем алгоритмы БПФ в общем случае. Это имеет смысл с точки зрения теории информации: и в самом деле, в вещественном сигнале содержится примерно в два раза меньше той информации, чем в комплексном. Обычный способ понизить сложность алгоритма БПФ для чисто вещественных сигналов заключается в том, чтобы заключить половину данных чисто вещественного сигнала в мнимую часть сигнала вдвое меньшего размера, т. е. сформировать комплекснозначный сигнал

$$y_j = x_j + ix_{j+D/2}$$

для $j \in [0, D/2 - 1]$. Обратите внимание, что сигнал $y$ имеет длину $D/2$. После этого надо выполнить полный алгоритм БПФ для комплексного сигнала $y$ вдвое меньшей длины, и использовать какую-либо формулу реконструкции, чтобы получить правильные значения ДПФ для исходного сигнала $x$. Пример такого подхода к чисто вещественным сигналам в случае теоретико-числовых преобразований применительно к вычислению циклической свертки содержится в алгоритме 9.5.22, где символический псевдокод с разделенным основанием приведен в [Crandall 1997b]. Обсуждение случая негациклической свертки смотрите в упражнении 9.51.

Кстати, имеются алгоритмы быстрого преобразования Фурье еще меньшей сложности, называемые БПФ с разделенным основанием, использующие равенство более сложное, чем формула Даниэльсона—Ланцоша. А также имеется алгоритм Сёренсона для БПФ с разделенным основанием в чисто вещественном случае, который вполне эффективен и широко применяется [Crandall 1994b]. Эта область переполнена особо оптимизированными методами БПФ, порой о структуре алгоритма БПФ писались целые трактаты; см., например, [Van Loan 1992].

Даже в конце XX в. каждый год продолжало появляться большое количество публикаций о новых оптимизациях БПФ. Поскольку нашей текущей темой является реализация БПФ для вычислений с числами большой разрядности, мы заканчиваем этот раздел еще одним алгоритмом: «параллельным», или «покусочным», методом БПФ, который вполне можно использовать как минимум в двух практических случаях. Первый случай, когда данных сигнала особенно много, алгоритм БПФ должен выполняться, укладываясь в ограниченный объем памяти. На практике это означает, что сигнал может физически оставаться в дисковой памяти и превосходить объем машинной памяти произвольного доступа. Идея заключается в том, чтобы «подкачивать» данные из большего объема памяти, обрабатывать их и сохранять результат прямо в предназначенное для него место. Поскольку вычисления происходят большей частью на не связанных между собой частях данных, алгоритм может также быть использован при параллельных расчетах, где каждый отдельный процессор должен обрабатывать соответствующую часть БПФ. Алгоритм изучали различные исследователи [Agarwal and Cooley 1986], [Swarztrauber 1987], [Ashworth and Lyne 1988], [Bailey 1990], особенно в отношении практических затрат памяти. Любопытно, что основные идеи, похоже, впервые появились в работе [Gentleman and Sande 1966]. Видимо, с учетом крайней популярности и бурного распространения исследований БПФ, следует простить исследователям то, что они не замечали этого первоисточника в течение двух десятилетий.

Алгоритм параллельного вычисления БПФ основывается на наблюдении, что вычисление ДПФ размера $D = WH$ может быть выполнено с помощью перебора строк и столбцов матрицы размера $H \times W$ (высота на ширину). Все остальное получается из следующего правила алгебраического понижения

ДПФ $X$ для сигнала $x$:

$$\begin{aligned} X &= \text{ДПФ}(x) = \left( \sum_{j=0}^{D-1} x_j g^{-jk} \right)_{k=0}^{D-1} \\ &= \left( \sum_{J=0}^{W-1} \sum_{M=0}^{H-1} x_{J+MW} g^{-(J+MW)(K+NH)} \right)_{K+NH=0}^{D-1} \\ &= \left( \sum_{J=0}^{W-1} \left( \sum_{M=0}^{H-1} x_{J+MW} g_H^{-MK} \right) g^{-JK} g_W^{-JN} \right)_{K+NH=0}^{D-1}, \end{aligned}$$

где $g, g_H, g_W$ — корни из единицы порядка $WH, H, W$ соответственно, а для индексов $(K+NH)$ верно $K \in [0, H-1]$, $N \in [0, W-1]$. В последней двойной сумме можно усмотреть БПФ строк и столбцов некоторой матрицы, как это видно из следующего алгоритма:

**Алгоритм 9.5.7** (параллельный «четырехшаговый» метод БПФ). Пусть $x$ — сигнал длины $D = WH$. Для эффективности алгоритма мы предполагаем, что сигнал $x$ записан в матрицу $T$ по колонкам, т. е. для $j \in [0, W-1]$ $j$-я колонка матрицы $T$ содержит (изначально последовательные) $H$ элементов $(x_{jH+M})_{M=0}^{H-1}$. Затем этот же алгоритм применяется к некоторым строкам некоторой матрицы; вектор значений $k$-й строки матрицы $U$ мы будем обозначать через $U^{(k)}$. Результирующее значение ДПФ окажется сохранено в таком же порядке по колонкам матрицы.

1. [Выполнить $H$ операций БПФ длины $W$, сохраняющих результат на месте и выполняемых каждая над своей строкой матрицы $T$]
   for$(0 \leqslant M < H)$ $T^{(M)} = \text{БПФ}\left(T^{(M)}\right)$;
2. [Транспонировать и модифицировать матрицу $T$]
   $(T_{JK}) = (T_{KJ} g^{-JK})$;
3. [Выполнить $W$ операций БПФ длины $H$, сохраняющих результат на месте и выполняемых каждая над своей строкой новой матрицы $T$]
   for$(0 \leqslant J < W)$ $T^{(J)} = \text{БПФ}\left(T^{(J)}\right)$;
4. [Вернуть ДПФ$(x)$ как элементы, записанные по столбцам матрицы]
   return $T$; // $T_{MJ}$ равно ДПФ$(x)_{JH+M}$.

Обратите внимание, что независимо от того, какая схема используется, можно использовать транспонирование (см. упражнение 9.53), чтобы преобразовать лексикографически упорядоченные входные данные $x$ в требуемый формат, когда данные записаны по колонкам, и таким же образом для восстановления порядка ДПФ в конце алгоритма. Другими словами, если предполагается, что входные данные хранятся в лексикографическом порядке, то можно добавить две дополнительные транспозиции, одну перед началом алгоритма и одну в конце, чтобы заставить алгоритм 9.5.7 выдавать стандартное БПФ размера $WH$. Здесь будет полезен небольшой пример, показывающий, как работает алгоритм. Алгоритм 9.5.7 для преобразования БПФ размера $N = 4 = 2 \cdot 2$

требует первообразный корень из единицы $g = e^{2\pi\imath/4} = i$ и начальные данные, расположенные по колонкам матрицы таким образом:

$$T = \begin{pmatrix} x_0 & x_2 \\ x_1 & x_3 \end{pmatrix}.$$

Первый шаг алгоритма заключается в том, чтобы выполнить $H = 2$ преобразований Фурье, каждое над своей строкой матрицы $T$ длины $W = 2$, чтобы получить результат

$$T = \begin{pmatrix} x_0 + x_2 & x_0 - x_2 \\ x_1 + x_3 & x_1 - x_3 \end{pmatrix}.$$

Затем мы транспонируем и модифицируем матрицу с помощью умножения на матрицу фазы

$$(g^{-JK}) = \begin{pmatrix} 1 & 1 \\ 1 & -i \end{pmatrix},$$

и получаем

$$T = \begin{pmatrix} x_0 + x_2 & x_1 + x_3 \\ x_0 - x_2 & -i(x_1 - x_3) \end{pmatrix}.$$

Тогда применение еще одного набора преобразований Фурье к строкам матрицы дает

$$T = \begin{pmatrix} X_0 & X_2 \\ X_1 & X_3 \end{pmatrix},$$

где $X_k = \sum_j x_j g^{-jk}$ совпадают со значением компонент ДПФ, и мы замечаем, что результат вычислений расположен в матрице по колонкам в требуемом порядке.

Если заинтересоваться, чем это преобразование отличается от *двумерного* преобразования Фурье, которое используется в обработке изображений, ответ будет простым: эта четырехшаговая (или шестишаговая, если учитывать предварительное и заключительное транспонирование, необходимое для расположения данных по строкам) процедура включает в себя внутреннее преобразование умножения на матрицу фазы в шаге [Транспонировать и модифицировать матрицу $T$]. Двумерное БПФ не содержит никакого внутреннего шага модификации матрицы; вместо этого просто выполняются БПФ для строк, а затем для столбцов.

Конечно, с учетом применения алгоритма 9.5.7 к самому себе, можно выбрать правило для повышения его эффективности: сигналы и результаты их преобразования должны всегда быть записаны по столбцам матрицы, а не по строкам. Более того, можно потребовать, чтобы сигналы длины $N = 2^n$ сохранялись в матрице размера $W = H = \sqrt{N} = 2^{n/2}$ при четном $n$, и $W = 2H = 2^{(n+1)/2}$ при нечетном $n$. Тогда матрица является квадратной или прямоугольной с соотношением сторон 1:2. Далее, для вычисления обратного БПФ, которое происходит в точности так же, как выше, но с рекурсивными вызовами $\text{БПФ}^{-1}$ и умножением на противоположную фазовую матрицу $g^{+JK}$, и с окончательным делением результата на $N$, можно предполагать, что $W' = H'$ или $H' = 2W'$, так что в случае решения таких задач, как вычисле-

ние сверток, результирующую матрицу прямого БПФ можно подать на вход алгоритму для нахождения обратного БПФ, даже если матрица не будет квадратной. На самом деле, для сверток самих по себе имеются другие интересные методы оптимизации, принадлежащие Пападопулосу, такие как использование методики прореживания по времени/по частоте и битовых перестановок степеней $g$, а также реализация Майера БПФ для очень больших значений $N$, не требующая транспонирования; вместо этого в ней используется шаг быстрого экономного постолбцового вычисления БПФ; см. [Crandall et al. 1999].

Интересным побочным результатом такого подхода является то, что для решения задачи оказывается нужным изучить проблему обычного транспонирования матриц. Работа [Bailey 1989] дает интересный пример алгоритма быстрого транспонирования сохраненной матрицы, описанного в [Fraser 1976], в то время как работа [Van Loan 1992, p. 138] показывает, насколько сейчас активными являются исследования быстрого транспонирования. Подобные алгоритмы имеют применение и в других областях арифметики больших чисел, пример см. в разделе 9.5.7.

Рассмотрим теперь исследование, которое оказалось необычайно значительным с самого момента его обнаружения первооткрывателями Даттом и Рохлиным. Основной результат в их плодотворной работе [Dutt and Rokhlin 1993] затрагивает некоторое *неоднородное* БПФ длины $D$

$$X_k = \sum_{j=0}^{D-1} x_j e^{-2\pi\imath k\omega_j/D}, \tag{9.23}$$

где все, что известно заранее о (возможно неоднородных) частотах $\omega_j$ — это то, что они лежат в интервале $[0, D)$. Это представление для $X_k$ сравнивается со стандартным ДПФ (9.20) для корня $g = e^{-2\pi\imath/D}$; в последнем случае мы получаем однородный вариант, $\omega_j = j$. Заметным результатом — ему уже десять лет, но, как мы говорим, выдающимся по значительности — является то, что такие неоднородные БПФ можно вычислять с абсолютной погрешностью $\varepsilon$ за

$$O\left(D \ln D + D \ln \frac{1}{\varepsilon}\right)$$

операций. Одним словом: этот метод неоднородных БПФ почти настолько же эффективен, что и метод стандартного, однородного БПФ. Этот эффективный алгоритм нашел приложения в таких различных областях, как задача гравитационной динамики $N$ тел (поскольку силы притяжения большого числа тел можно представить как разновидность свертки) и вычисление дзета-функции Римана (см., например, упражнение 1.62).

В следующем изложении алгоритма мы отклонились от литературы в нескольких местах. Во-первых, мы заставляем индексы в таких суммах, как (9.23), пробегать значения $j, k \in [0, D-1]$ для последовательности изложения; во многих книгах используются равносильные суммы, где $j$ или $k \in [-D/2, D/2-1]$. Во-вторых, мы выбрали алгоритм, являющийся быстрым и устойчивым (в смысле гарантированной точности даже при необычном поведении входного сигнала), но не обязательно минимальным по сложности.

Устойчивость, несомненно, важна для строгих подсчетов в вычислительной теории чисел. В-третьих, мы выбрали именно этот алгоритм, поскольку он не полагается на вычисление функций специального вида, таких как гауссовы или просмотровые функции. Штрафом за все эти упрощения является то, что нам потребуется выполнить определенное число стандартных преобразований Фурье длины $D$, число этих преобразований зависит только *логарифмически* от желаемой точности.

В дальнейшем «погрешность» $\varepsilon$ означает, что настоящее ДПФ (9.23) $X_k$ и вычисленное следующим алгоритмом $X'_k$ отличаются на величину

$$|X_k - X'_k| \leqslant \varepsilon \sum_{j=0}^{D-1} |x_j|.$$

Далее мы приводим алгоритм, вычисляющий $X_k$ (9.23) с погрешностью $\varepsilon < 2^{-b}$, т. е. с точностью $b$ битов, за

$$O\left(\frac{b}{\lg b} D \ln D\right) \tag{9.24}$$

операций. Алгоритм основан на замечании, что

$$\frac{2\pi^B}{8^B B!} < 2^{-b} \quad \text{для} \quad B = \left\lceil \frac{2b}{\lg b} \right\rceil. \tag{9.25}$$

Такое неравенство позволяет нам строго оценить погрешность $\varepsilon$ в этом алгоритме (см. упражнение 9.54).

**Алгоритм 9.5.8** (неоднородное БПФ). **Пусть задан $x$ — сигнал длины $D \equiv 0 \pmod 8$, и вещественные (не обязательно целые) частоты $(\omega_j \in [0, D) : j \in [0, D-1])$. По заданной битовой точности $b$ (так что относительная погрешность составляет $\varepsilon < 2^{-b}$) этот алгоритм возвращает приближение $X'_k$ точного значения ДПФ (9.23).**

1. [Начальная установка]
   $B = \left\lceil \frac{2b}{\lg b} \right\rceil$;
   for($j \in [0, D-1]$) {
   $\mu_j = \lfloor \omega_j + \frac{1}{2} \rfloor$;
   $\theta_j = \omega_j - \mu_j$; // Так что $|\theta_j| \leqslant 1/2$.
   }
2. [Выполнить $8B$ стандартных БПФ]
   for($K \in [0, 7]$) {
   for($\beta \in [0, B-1]$) {
   $s = (0)_0^{D-1}$; // Нулевой сигнал длины $D$.
   for($j \in [0, D-1]$) {
   $\mu = \mu_j \bmod D$;
   $s_\mu = s_\mu + x_j e^{-2\pi\imath K\omega_j/8} \theta_j^\beta$.
   }
   $F_{K,\beta} = FFT(s)$; // $(F_{K,\beta,m} : m \in [0, D-1])$

// является результатом.

}

}

3. [Создать приближение к результату]

$X' = \cup_{K=0}^{7} \left( \sum_{\beta=0}^{B-1} F_{K,\beta,m} (-2\pi i m/D)^{\beta} \frac{1}{\beta!} \right)_{m=0}^{D/8-1}$

return $X'$; // Приближение неоднородного ДПФ (9.23).

Алгоритм 9.5.8 записан в достаточно сжатой форме, однако типичная реализация на компьютере выглядит похоже на этот псевдокод. Заметим, что объединение сигналов в конце, являющееся конкатенацией слева направо сигналов длины $D/8$, можно вычислить в некоторых языках программирования настолько же просто, как записано у нас. Кстати, символическая запись скрывает причину, почему этот алгоритм работает правильно, однако несложно видеть, что разложение Тейлора для $e^{-2\pi\imath(\mu_j+\theta_j)k/D}$ по степеням $\theta_j$ и искусные манипуляции с индексами приводят к успеху. Любопытным и удачным оказывается факт, что разбиения длины сигнала на фиксированный множитель 8 достаточно для всевозможных входных сигналов $x$ алгоритма. Такова выгода неравенства (9.25). В итоге, число требуемых стандартных БПФ, необходимых для точности $b$ битов, составляет примерно $16b/\lg b$. Это подсчет числа БПФ для наихудшего случая, поскольку практические приложения часто получают точность, намного лучшую, чем $b$ битов, когда заданный параметр $b$ подается на вход алгоритма 9.5.8.

После оригинальной работы Датта—Рохлина появились различные работы, такие как [Ware 1998], [Nguyen and Liu 1999], дающие в чем-то лучшую точность, или несколько улучшенную скорость исполнения, или другие усовершенствования. Имелся даже подход, минимизирующий погрешность в наихудшем случае для входных сигналов единичной нормы [Fessler and Sutton 2003]. Однако все время с момента выхода оригинальной работы Датта–Рохлина до текущего момента, основная идея оставалась той же: свести вычисление сигнала $X'$ к вычислению множества *однородных и стандартных* БПФ лишь несколько большей излишней длины.

Закончим раздел, посвященный БПФ, упоминанием некоторых новых исследований в области БПФ с «гигантскими элементами», которые стало возможно проводить за разумное время. Так, [Crandall et al. 2004] обсуждают теорию и реализацию каждого из следующих случаев гигантских элементов:

- Одномерное БПФ длины $2^{30}$ (вычисляемое алгоритмом 9.5.7).
- Двумерное БПФ длин $2^{15} \times 2^{15}$.
- Трехмерное БПФ длин $2^{10} \times 2^{10} \times 2^{10}$.

С такими массивными размерами сигналов приходят сложные, но завораживающие проблемы быстрой транспозиции матриц с использованием кэша и векторизации вычислений с плавающей точкой. Результат, касающийся скорости выполнения, заключается в том, что одномерное преобразование длины

$2^{30}$ занимает меньше одной минуты на современном аппаратном блоке, если использовать числа с плавающей точкой двойной точности. (Двумерный и трехмерный случаи примерно так же быстры; на самом деле двумерный случай обычно самый быстрый по техническим причинам.)

Для вычислительной теории чисел эти новые результаты означают следующее: на аппаратном блоке, скажем, помещающемся в шкаф, можно перемножать числа длины миллиард десятичных знаков примерно за минуту. Такие результаты зависят от правильного решения следующей проблемы: погрешность в таких устрашающих преобразованиях Фурье может быть нетривиальной. Имеется так много складываемых и перемножаемых членов, что результат отклоняется от истинного, скажем, как разновидность случайного блуждания. Интересно, что БПФ длины $D$ может быть смоделировано процессом случайного блуждания в $D$ измерениях, имеющим $O(\ln D)$ шагов. Работа [Crandall et al. 2004], таким образом, сообщает о количественных оценках погрешности БПФ; впервые такие оценки исследовали Майер и Персиваль.

### 9.5.3. Теория сверток

Пусть $x$ обозначает сигнал $(x_0, x_1, \ldots, x_{D-1})$, и пусть элементы сигнала $x$ будут, например, цифрами из определений (9.1.1) или (9.1.2) (хотя мы не настаиваем на том, чтобы они были цифрами; изложенная теория является достаточно общей). Определим базовые операции сверток сигналов. В дальнейшем мы предполагаем, что сигналы $x, y$ имеют одну и ту же длину — $D$ элементов. Во всех суммах следующего определения индексы $i, j$ пробегают значения из множества $\{0, \ldots, D-1\}$:

**Определение 9.5.9.** Циклической сверткой сигналов $x, y$ длины $D$ называется сигнал, обозначаемый $z = x \times y$, имеющий $D$ элементов, определяемых равенством

$$z_n = \sum_{i+j \equiv n \pmod D} x_i y_j,$$

негациклической сверткой двух сигналов $x, y$ называется сигнал $v = x \times_- y$, имеющий $D$ элементов, определяемых равенством

$$v_n = \sum_{i+j=n} x_i y_j - \sum_{i+j=D+n} x_i y_j,$$

и ациклической сверткой сигналов $x, y$ является сигнал $u = x \times_A y$, имеющий $2D$ элементов, определяемых равенством

$$u_n = \sum_{i+j=n} x_i y_j,$$

где $n \in \{0, \ldots, 2D-2\}$, плюс еще одним равенством $u_{2D-1} = 0$. Наконец, полуциклической сверткой $x, y$ называется сигнал $x \times_H y$ длины $D$, состоящий из первых $D$ элементов ациклической свертки $u$.

Эти основные виды сверток тесно связаны, как будет видно из следующей теоремы. В утверждениях такого типа мы понимаем сумму сигналов $c = a + b$

как результат поэлементного сложения $a$ и $b$, т. е. $c_n = a_n + b_n$ для всех $n$. Подобным образом, произведение скаляра и сигнала $qa$, где $q$ — число, а $a$ — сигнал, определяется как сигнал $(qa_n)$. Нам потребуется обозначение для разделения сигналов четной длины на две половинки, и мы обозначим через $L(a)$ и $H(a)$ соответственно младшую и старшую часть сигнала $a$. Таким образом, если $c = a \cup b$ является обычной конкатенацией двух сигналов равной длины, то $L(c) = a$ и $H(c) = b$.

**Теорема 9.5.10.** *Пусть сигналы $x, y$ имеют одну и ту же длину $D$. Тогда разные типы сверток связаны следующим образом (предполагается, что в рассматриваемой области, которой принадлежат элементы сигналов, существует $2^{-1}$):*

$$x \times_H y = \frac{1}{2}((x \times y) + (x \times_- y)).$$

*Далее,*

$$x \times_A y = (x \times_H y) \cup \frac{1}{2}((x \times y) - (x \times_- y)).$$

*Наконец, если длина $D$ является четной, а $x_j, y_j = 0$ для $j \geqslant D/2$, то*

$$L(x) \times_A L(y) = x \times y = x \times_- y.$$

Эти соотношения позволяют использовать некоторые алгоритмы более универсально. Например, пары алгоритмов для циклической и негациклической сверток достаточно, чтобы получить полуциклическую и ациклическую свертки, и аналогично для остальных видов сверток. В последнем утверждении теоремы мы ввели понятие «заполнения нулями» $(x_j, y_j = 0)$, которое на практике означает присоединение $D$ нулей к сигналу длины $D$, так что для двух дополненных нулями последовательностей ациклическая свертка совпадает с циклической (или негациклической).

Связь между сверткой и дискретным преобразованием Фурье из предыдущего раздела явно указана в следующей знаменитой теореме, где мы воспользовались оператором поэлементного умножения сигналов $*$, для которого сигнал $z = x * y$ имеет элементы $z_n = x_n y_n$:

**Теорема 9.5.11** (теорема о свертке). *Пусть сигналы $x, y$ имеют одну и ту же длину $D$. Тогда циклическая свертка сигналов $x, y$ удовлетворяет равенству*

$$x \times y = ДПФ^{-1}(ДПФ(x) * ДПФ(y)),$$

*которое означает*

$$(x \times y)_n = \frac{1}{D} \sum_{k=0}^{D-1} X_k Y_k \, g^{kn}.$$

Используя эту знаменитую теорему, мы можем выполнять циклическую свертку двух сигналов с помощью трех преобразований Фурье (одно из которых будет обратным), или циклическую свертку сигнала самого с собой с помощью двух преобразований Фурье. Поскольку известно, что сложность ДПФ не превосходит $O(D \ln D)$ операций в рассматриваемом поле, сложность поэлементного умножения, равная $O(D)$, является асимптотически незначительной.

Прямым и элегантным применением БПФ к вычислениям с большими числами является использование ДПФ в соответствии с теоремой 9.5.11 для вычисления ациклической свертки и, тем самым, выполнения умножения. Основная идея, впервые предложенная Штрассеном в 1960-х гг., и впоследствии улучшенная [Schönhage and Strassen 1971] (см. раздел 9.5.6), заключается в следующем. Если целое число $x$ представляется как сигнал длины $D$, состоящий из цифр системы счисления с основанием $B$, и то же самое верно для числа $y$, то целочисленное произведение $xy$ является сверткой длины $2D$. Теорема 9.5.11 работает с циклической, а не ациклической сверткой, однако у нас есть теорема 9.5.10, позволяющая дополнить сигналы нулями и вычислять впоследствии циклическую свертку. Эта идея в случае комплекснозначного преобразования приводит к следующей схеме, которую обычно реализуют, используя арифметику чисел с плавающей точкой, и где ДПФ вычисляются с помощью быстрого преобразования Фурье:

**Алгоритм 9.5.12** (базовое умножение с помощью БПФ). Пусть заданы два неотрицательных целых числа $x, y$, имеющие не больше $D$ цифр в некоторой системе счисления с основанием $B$ (согласно определению 9.1.1). Этот алгоритм возвращает представление произведения $xy$ в системе счисления с основанием $B$. (Следует иметь в виду, что используемые в алгоритме преобразования Фурье обычно являются операциями над числами с плавающей точкой, так что при вычислениях могут возникнуть погрешности округления.)

1. [Начальная установка]

   Дополнить нулями $x, y$ до тех пор, пока каждое из них не станет числом размера $2D$, так что циклическая свертка новых последовательностей будет содержать ациклическую свертку исходных последовательностей;

2. [Преобразования Фурье]
   $X = \text{ДПФ}(x)$; // Использовать быстрый метод БПФ.
   $Y = \text{ДПФ}(y)$;
3. [Покомпонентные умножения]
   $Z = X * Y$;
4. [Обратное преобразование]
   $z = \text{ДПФ}^{-1}(Z)$;
5. [Округлить цифры результата]
   $z = round(z)$;
   // Покомпонентное округление к ближайшему целому числу.
6. [Выполнить переносы по основанию $B$]
   $carry = 0$;
   for($0 \leqslant n < 2D$) {
   $v = z_n + carry$;
   $z_n = v \bmod B$;
   $carry = \lfloor v/B \rfloor$;
   }

7. [Учет последней цифры]
Удалить начальные нули, возможно, $carry > 0$ станет старшей цифрой числа $z$;
return $z$;

Это описание алгоритма достаточно общее, оно передает только основные принципы умножения с помощью БПФ, заключающиеся в использовании преобразований Фурье, округлении и выполнении переносов. Имеется огромное число не упомянутых деталей, не говоря уже об огромном числе улучшений этого метода, некоторые из которых мы рассмотрим позже. Но за всеми этими мелочами скрывается одно очень важное предупреждение: точность вычислений с плавающей точкой всегда должна оставаться под вопросом. Ключевым шагом в общем алгоритме 9.5.12 является поэлементное округление сигнала $z$. Если ошибки округления в БПФ будут слишком большими, элемент свертки $z$ может оказаться округлен к неправильному значению.

Одно немедленное практическое улучшение алгоритма 9.5.12 заключается в том, чтобы использовать сбалансированное представление (см. определение 9.1.2) вместо обычного представления по основанию $B$. Обнаруживается, что в этом представлении ошибки вычислений с плавающей точкой существенно уменьшаются [Crandall and Fagin 1994], [Crandall et al. 1999]. Это явление уменьшения ошибок еще не полностью объяснено, однако, видимо, оно имеет отношение к меньшей величине цифр сомножителей, к тому же, вероятно, происходит некоторое сокращение компонент ДПФ, поскольку среднее арифметическое цифр в сигнале должно быть малым благодаря сбалансированности.

Перед тем как переходить к таким вопросам, как дальнейшие улучшения метода умножения, использующего БПФ, и проблеме чисто целочисленных сверток, мы должны упомянуть, что свертки могут появляться в теоретико-числовых работах даже вдалеке от проблем арифметики больших чисел. В заключение раздела мы приводим два примера таких вычислений; а именно, свертки применительно к проблеме просеивания и к проблеме регулярности простых чисел.

Рассмотрим следующую теорему, напоминающую о знаменитой проблеме Гольдбаха:

**Теорема 9.5.13.** *Пусть $N = 2 \cdot 3 \cdot 5 \ldots p_m$ — произведение первых $m$ простых чисел. Тогда каждое достаточно большое $n$ представляется в виде суммы $n = a + b$ чисел $a, b$, взаимно простых с $N$.*

Интересно, что эту теорему можно без особых проблем доказать с использованием теории сверток. (Следует упомянуть, что имеются также другие доказательства, использующие идеи китайской теоремы об остатках, так что мы используем эту теорему лишь для того, чтобы дать пример применения свертки в дискретном случае (см. упражнение 9.40).) Основная идея заключается в том, чтобы рассмотреть особый сигнал $y$, определяемый равенствами $y_j = 1$, если НОД$(j, N) = 1$, и $y_j = 0$ иначе, определенной длины $D$. Тогда ациклическая свертка $y \times_A y$ будет в точности сообщать нам, какие $n < D$ из теоремы

имеют представление в виде $a+b$, и более того, $n$-й элемент свертки в точности является количеством таких представлений для $n$.

В качестве краткого отступления отметим, что сама гипотеза Гольдбаха верна, если другой сигнал бесконечной длины, а именно,

$$G = (1,1,1,0,1,1,0,1,1,0,\ldots),$$

где единицы стоят на позициях $(p-3)/2$ для нечетных простых чисел $p = 3,5,7,11,13,\ldots$, имеет то свойство, что ациклическая свертка $G \times_A G$ не содержит ни одного нуля. В этом случае $n$-й элемент свертки в точности равен числу представлений Гольдбаха числа $2n+6$.

Снова о теореме 9.5.13: полезным является изучение дискретного преобразования Фурье $Y$ длины $N$ вышеупомянутого сигнала $y$. Это ДПФ оказывается известной суммой:

$$Y_k(N) = c_N(k) = \sum_{\text{НОД}(j,N)=1} e^{\pm 2\pi ijk/N}, \tag{9.26}$$

где $j$ пробегает те числа отрезка $[0, N-1]$, которые взаимно просты с $N$, так что выбор знака у экспоненты не важен, а $c_N(k)$ — стандартное обозначение для суммы Рамануджана, о которой известно, что она обладает интересными мультипликативными свойствами [Hardy and Wright 1979]. На самом деле появление суммы Рамануджана в разделе 1.4.4 подсказывает, что $c_N$ должно иметь некоторые приложения в теории сверток в дискретном случае. Мы оставляем доказательство теоремы 9.5.13 читателю (см. упражнение 9.40), но хотим указать несколько главных моментов. Во-первых, сумму в равенстве (9.26) можно представлять себе как результат «отсеивания» конечных сумм, соответствующих делителям $N$. Это приводит к интересной алгебре последовательностей. Во-вторых, стоит отметить, что *циклическая* свертка длины $N$ сигнала $y$ самого с собой может быть выражена явной формулой. Результат имеет вид:

$$(y \times y)_n = \varphi_2(N,n) = \prod_{p|N}(p - \theta(n,p)), \tag{9.27}$$

где $\theta(n,p)$ равна 1, если $p|n$, и 2 в противном случае. Таким образом, для $0 \leqslant n < N$, это произведение равно точному числу представлений чисел $n$ или $n+N$ в виде $a+b$, где оба числа $a, b$ взаимно просты с $N$. Как объясняется в упражнениях, для того чтобы завершить эту линию рассуждения, требуется привлечь идеи негациклической свертки (или какой-либо другой способ, такой как просеивание), чтобы показать, что представлений чисел $n+N$ имеется в определенной области значений $n$ меньше, чем самих $n$. Эти рассуждения вместе с несколькими окончательными аргументами доказывают теорему 9.5.13.

Теперь о другом использовании сверток. В 1847 г. Куммер обнаружил, что для регулярного простого числа $p > 2$ верна великая теорема Ферма, утверждающая, что уравнение

$$x^p + y^p = z^p$$

не имеет целочисленных решений, таких что все три числа $x, y, z$ не равны

нулю, верна. (Отметим, что великая теорема Ферма теперь является настоящей теоремой Уайлса, однако рассматриваемые здесь методы предшествовали той работе и до сих пор имеют приложения к таким открытым проблемам, как гипотеза Вандивера.) Далее, $p$ является регулярным, если оно не делит ни одного из числителей чисел Бернулли с четными индексами

$$B_2, B_4, \ldots, B_{p-3}.$$

Имеется элегантное соотношение, открытое Шокроллахи в 1994, см. [Buhler et al. 2000], дающее точное выражение для чисел Бернулли:

**Теорема 9.5.14.** *Пусть $g$ — примитивный корень из нечетного числа $p$, и пусть*

$$c_j = \left\lfloor \frac{(g^{-1} \bmod p)(g^j \bmod p)}{p} \right\rfloor g^{-j}$$

*для $j \in [0, p-2]$. Тогда для $k \in [1, (p-3)/2]$ имеем*

$$\sum_{j=0}^{p-2} c_j g^{2kj} \equiv \left(1 - g^{2k}\right) \frac{B_{2k}}{2kg} \pmod{p}. \tag{9.28}$$

Видно, что соотношение Шокроллахи включает в себя ДПФ длины $(p-1)$, с полем операндов $\mathbf{F}_p$. Можно было бы продолжать использовать алгоритм для БПФ, если бы такой подход не содержал две проблемы. Во-первых, лучшие длины для БПФ являются степенями двойки; во-вторых, нельзя использовать арифметику с плавающей точкой, особенно когда простое число $p$ велико, за исключением тех случаев, когда точность вычислений чрезвычайно высока (и при этом каким-то образом гарантирована). Но у нас есть возможность вычисления самого ДПФ с помощью свертки (см. алгоритм 9.6.6), то же верно и для метода Шокроллахи поиска регулярных простых чисел. Это в самом деле так, потому что вычисление точных индексов нерегулярности любого простого числа можно выполнить с помощью сверток длин степеней двойки. Как мы увидим далее, для этого можно применять методы, использующие «символическое преобразование Фурье», например, свертку Нюссбаумера, не содержащую вычислений с плавающей точкой и потому подходящую для вычисления чисто целочисленной свертки. Эти подходы — равенство Шокроллахи и свертка Нюссбаумера — были использованы вместе, чтобы найти все регулярные простые числа для $p < 12000000$ [Buhler et al. 2000].

### 9.5.4. Методы взвешенного дискретного преобразования (ВДП)

Одним из вариантов сверток, основанных на ДПФ, оказавшимся важным в современной теории простых чисел и вопросов факторизации (для случая, когда интересующие нас числа велики, например, лежат в области $2^{1000000}$ и выше), является дискретное взвешенное преобразование (ВДП). Это преобразование определяется следующим образом:

**Определение 9.5.15** (дискретное взвешенное преобразование (ВДП)). Пусть $x$ — сигнал длины $D$, и пусть $a$ — сигнал той же длины, причем каждый его элемент $a_j$ имеет в рассматриваемом кольце обратный элемент. Сигнал $a$ называется весовым сигналом. Дискретным взвешенным преобразованием называется сигнал $X = \text{ВДП}(x, a)$, имеющий элементы

$$X_k = \sum_{j=0}^{D-1} (a * x)_j g^{-jk}, \tag{9.29}$$

а обратным преобразованием является преобразование $\text{ВДП}^{-1}(X, a) = x$, заданное равенствами

$$x_j = \frac{1}{Da_j} \sum_{k=0}^{D-1} X_k g^{jk}. \tag{9.30}$$

Далее, взвешенной циклической сверткой двух сигналов называется сигнал $z = x \times_a y$, имеющий

$$z_n = \frac{1}{a_n} \sum_{j+k \equiv n \pmod{D}} (a * x)_j (a * y)_k. \tag{9.31}$$

Ясно, что ВДП является просто преобразованием Фурье покомпонентного произведения $a * x$ с элементами $a_j x_j$. Существенная выгода ВДП заключается в том, что использование определенных весовых сигналов приводит к разным полезным сверткам. В некоторых случаях использование ВДП устраняет необходимость дополнения нулями чисел в стандартном алгоритме 9.5.12 для БПФ. Далее следует важный результат:

**Теорема 9.5.16** (теорема о взвешенной свертке). *Пусть сигналы $x, y$ и весовой сигнал $a$ имеют одну и ту же длину $D$. Тогда взвешенная циклическая свертка сигналов $x, y$ удовлетворяет равенству*

$$x \times_a y = \text{ВДП}^{-1}(\text{ВДП}(x, a) * \text{ВДП}(y, a), a),$$

*или, что то же самое,*

$$(x \times_a y)_n = \frac{1}{Da_n} \sum_{k=0}^{D-1} (X * Y)_k g^{kn}.$$

Таким образом, алгоритмы БПФ могут применяться теперь и ко взвешенной свертке. В частности, можно таким же образом вычислять не только циклическую, но и негациклическую свертку, поскольку выбор особого весового сигнала

$$a = \left(A^j\right), \quad j \in [0, D-1],$$

дает, при условии, что $A$ является примитивным корнем степени $2D$ из единицы в этом поле, следующее равенство:

$$x \times_- y = x \times_a y, \tag{9.32}$$

означающее, что взвешенная циклическая свертка является в данном случае негациклической. Обратите внимание, что в случае, если у корня $g$ степени $D$

из единицы в этом поле имеется квадратный корень, как, например, в случае комплексных чисел, можно просто положить $A^2 = g$, чтобы вычислить негациклическую свертку. Другой интересный пример порождающего элемента $A$, а именно, когда $A$ является примитивным корнем степени $4D$ из единицы, приводит к так называемой прямоугольной свертке [Crandall 1996a].

Эти рассмотрения ведут, в свою очередь, к важному алгоритму, который был использован, чтобы улучшить современные методы факторизации. С помощью преобразования ВДП этот метод полностью избегает дополнения сигналов нулями. Рассмотрим проблему умножения двух чисел по модулю числа Ферма $F_n = 2^{2^n} + 1$. Эта операция, разумеется, может производиться огромное число раз в ходе попыток факторизовать число $F_n$. Имеется по меньшей мере три метода нахождения $(xy) \bmod F_n$ с помощью сверток сигналов длины $D$ и основания системы счисления $B$, выбранного так, чтобы было выполнено равенство $F_n = B^D + 1$:

(1) Дополнить нулями каждое из чисел $x, y$ до длины $2D$, посчитать циклическую свертку, выполнить переносы и привести результат по модулю $F_n$.
(2) Выполнить взвешенную свертку длины $D$, с порождающим вес элементом $A$, равным примитивному корню степени $2D$ из единицы, и выполнить все необходимые переносы.
(3) Создать комплексные сигналы длины $D/2$, равные $x' = L(x) + iH(x)$, поступить так же с $y'$, использовать взвешенную свертку с порождающим вес элементом, равным примитивному корню степени $4D$ из единицы, выполнить переносы.

В методе (1), разумеется, можно использовать алгоритм 9.5.12, видимо, с быстрым умножением по модулю чисел Ферма из раздела 9.2.3. Однако вместо этого можно использовать чисто целочисленное умножение, которое обсуждается далее. Метод (2) является подходом с использованием негациклической свертки, в котором применение взвешенной свертки может пониматься как умножение по модулю $(\bmod F)_n$; это означает, что операции умножения выполняются почти бесплатно (см. упражнения). Метод (3) является подходом с использованием прямоугольной свертки, который также делает операции умножения бесплатными (см. упражнения). Обратите внимание, что ни метод (2), ни метод (3) не требуют дополнения нулями, а метод (3) по сути сокращает длины сигналов вдвое (но зато требует вычислений с комплексными числами). Мы сосредоточим внимание на методе (3), чтобы сформулировать следующий алгоритм, который, как и алгоритм 9.5.12, часто реализуют в рамках парадигмы вычислений с плавающей точкой:

**Алгоритм 9.5.17** (умножение с помощью ВДП по модулю чисел Ферма). Пусть задано число Ферма $F_n = 2^{2^n} + 1$ и положительные числа $x, y \not\equiv -1 \pmod{F_n}$. Этот алгоритм возвращает $(xy) \bmod F_n$. Мы выбираем $B, D$ такими, что $F_n = B^D + 1$, а входные числа $x, y$ представляются в виде сигналов длины $D$,

состоящих из цифр системы счисления с основанием $B$. Мы предполагаем, что в этом поле существует корень $A$ степени $4D$ из единицы.

1. [Начальная установка]
   $E = D/2$; // Перейти к комплексным сигналам.
   $x = L(x) + iH(x)$; // Сигналы длины $E$.
   $y = L(y) + iH(y)$;
   $a = (1, A, A^2, \ldots, A^{E-1})$; // Весовой сигнал.
2. [Выполнить преобразования]
   $X = \text{ВДП}(x, a)$;
   $Y = \text{ВДП}(y, a)$;
   // С помощью эффективного алгоритма БПФ для длины $E$.
3. [Покомпонентное умножение]
   $Z = X * Y$;
4. [Обратное преобразование]
   $z = \text{ВДП}^{-1}(Z, a)$;
5. [Восстановить сигнал]
   $z = \mathrm{Re}(z) \cup \mathrm{Im}(z)$; // Теперь $z$ имеет длину $D$.
6. [Округлить цифры]
   $z = round(z)$;
   // Поэлементное округление к ближайшему целому числу.
7. [Выполнить переносы по основанию $B$]
   $carry = 0$;
   for($0 \leqslant n < D$) {
   $v = z_n + carry$;
   $z_n = v \bmod B$;
   $carry = \lfloor v/B \rfloor$;
   }
8. [Учет последней цифры]
   Сделать возможно оставшийся перенос $carry > 0$ старшей цифрой числа $z$;
   $z = z \bmod F_n$;
   // С помощью еще одного цикла переносов или с помощью вычислений по модулям особого вида.
   return $z$;

Отметим, что в шагах [Выполнить переносы по основанию $B$] и [Учет последней цифры] действия основываются на том, что цифры восстановленного числа $z$ являются положительными. Мы упоминаем это, потому что имеются эффективные методы, использующие сбалансированное представление, в случае чего требуется позаботиться о том, чтобы правильно обращаться с отрицательными цифрами и отрицательными переносами.

Этот алгоритм был использован для обнаружения новых делителей чисел $F_{13}, F_{15}, F_{16}$ и $F_{18}$ [Brent et al. 2000] (см. таблицу сомножителей чисел Ферма в разделе 1.3.2), а также для того, чтобы установить, что числа $F_{22}$, $F_{24}$ и многие сомножители других $F_n$ являются составными [Crandall et al. 1995], [Crandall

et al. 1999]. Позже Вольтман в своей работе [Woltman 2000] реализовал этот алгоритм, для того чтобы создать высокоэффективную программу для факторизации чисел Ферма (см. примечания, следующие за алгоритмом 7.4.4).

Другой вариант ВДП был использован для обнаружения восьми чисел Мерсенна $2^{1398269}-1, 2^{2976221}-1, 2^{3021377}-1, 2^{6972593}-1$ (см. таблицу 1.2), последнее из которых является наибольшим известным простым числом такого вида. Для того чтобы сделать эти открытия, компьютерная сеть добровольцев использовала всесторонние тесты Люка—Лемера, включавшие огромное количество возведений в квадрат по модулю $p = 2^q - 1$. Рассматриваемый вариант алгоритма был назван взвешенным дискретным преобразованием с иррациональным основанием (ВДПИО) [Crandall and Fagin 1994], [Crandall 1996a] из-за того, что использовалось особое представление цифр, напоминающее представление по иррациональному основанию; это представление является попыткой разложения по иррациональному основанию в дискретном случае. Пусть $p = 2^q - 1$. Заметим сначала, что целое число $x$ может быть представлено по основанию системы счисления $B = 2$ в виде

$$x = \sum_{j=0}^{q-1} x_j 2^j,$$

или, что то же самое, $x$ является сигналом $(x_j)$ длины $q$; и то же для $y$. Тогда циклическая свертка $x \times y$ содержит (с еще не выполненными переносами) цифры произведения $(xy) \bmod p$. Следовательно, в принципе, стандартное умножение можно выполнять с использованием БПФ по модулю простых чисел Мерсенна таким способом без дополнения сигналов нулями. Однако у этого подхода есть две проблемы. Во-первых, вычисления являются почти побитовыми, не использующими типичные для компьютеров операции с машинными словами. Во-вторых, требуется выполнить преобразование Фурье длины $q$, которая в общем случае не обязательно степень двойки. Это в принципе выполнимая операция (см. упражнения), однако вычисления с длиной, равной степени двойки, обычно более эффективны и определенно более распространены.

Оказывается, что оба препятствия на пути метода умножения по модулю простых чисел Мерсенна без дополнения нулями можно преодолеть, если только суметь представить целые числа $x$ по иррациональному основанию $2^{q/D}$, где $1 < D < q$ — некоторая степень двойки. Это объясняется тем, что представление

$$x = \sum_{j=0}^{D-1} x_j 2^{qj/D},$$

и такое же для $y$ (где цифры по основанию $B$ являются в общем случае тоже иррациональными), ведет к тождеству, без учета выполнения переносов, произведения $(xy) \bmod p$ и свертки $x \times y$. Однако теперь длины сигналов являются степенями двойки, а цифры, хотя и не являются целыми числами, имеют уже размер машинного слова. Оказывается, что можно смоделировать это представление по иррациональному основанию с помощью определенного

представления с переменным основанием в соответствии со следующей теоремой:

**Теорема 9.5.18** (Крэндалл). *Пусть $p = 2^q - 1$ (не обязательно простое), заданы целые числа $0 \leqslant x, y < p$, и выбрана длина сигнала $1 < D < q$. Число $x$ будем понимать как сигнал $(x_0, \ldots, x_{D-1})$ согласно следующему представлению с переменным основанием:*

$$x = \sum_{j=0}^{D-1} x_j 2^{\lceil qj/D \rceil} = \sum_{j=0}^{D-1} x_j 2^{\sum_{i=1}^{j} d_i},$$

*где*

$$d_i = \lceil qi/D \rceil - \lceil q(i-1)/D \rceil,$$

*и каждая цифра $x_j$ лежит в интервале $[0, 2^{d_j+1} - 1]$; точно так же поступаем с $y$. Определим весовой сигнал $a$ длины $D$ с помощью равенства*

$$a_j = 2^{\lceil qj/D \rceil - qj/D}.$$

*Тогда взвешенная циклическая свертка $x \times_a y$ является целочисленным сигналом, эквивалентным (без учета переносов) представлению произведения $(xy) \bmod p$ по переменному основанию.*

Эта теорема доказывается и обсуждается в [Crandall and Fagin 1994], [Crandall 1996a], единственной нетривиальной ее частью является доказательство того, что элементы взвешенной свертки $x \times_a y$ в самом деле являются целыми числами. Теорема немедленно дает следующий алгоритм:

**Алгоритм 9.5.19** (умножение по модулю чисел Мерсенна с использованием ВДПИО). Пусть задано число Мересна $p = 2^q - 1$ (не обязательно простое) и положительные целые числа $x, y$. Этот алгоритм вычисляет с помощью преобразования Фурье чисел с плавающей точкой представление произведения $(xy) \bmod p$ по переменному основанию. Мы используем обозначения из теоремы 9.5.18 и предполагаем, что длина сигнала $D = 2^k$ такова, что $\lfloor 2^{q/D} \rfloor$ помещается в машинное слово (т. е. достаточно мало, чтобы не вызвать ошибки арифметики целых чисел).

1. [Подготовить представления по переменному основанию]
   Определить сигнал $x$ как набор цифр $(x_j)$ по переменному основанию, как в теореме 9.5.18. Сделать то же самое для $y$;
   Создать весовой сигнал $a$, снова как в теореме 9.5.18;
2. [Выполнить преобразования]
   $X = \text{ВДП}(x, a)$; // С помощью БПФ длины $D$ с плавающей точкой.
   $Y = \text{ВДП}(y, a)$;
3. [Покомпонентное произведение]
   $Z = X * Y$;
4. [Обратное преобразование]
   $z = \text{ВДП}^{-1}(Z, a)$;
5. [Округлить цифры]
   $z = round(z)$; // Поэлементное округление.

6. [Выполнить переносы по переменному основанию]
   $carry = 0$;
   for($0 \leqslant n < len(z)$) {
      $B = 2^{d_{n+1}}$; // Размер цифры, стоящей на месте $n$.
      $v = z_n + carry$;
      $z_n = v \bmod B$;
      $carry = \lfloor v/B \rfloor$;
   }
7. [Окончательное приведение по модулю]
   Включить возможный перенос $carry > 0$ в старшую цифру $z$;
   $z = z \bmod p$;
      // С помощью цикла переносов или деления по модулю особого вида.
   return $z$;

Поскольку этот метод довольно запутан, здесь будет уместен пример. Рассмотрим умножение по модулю числа Мерсенна $p = 2^{521} - 1$. Берем число $q = 521$ и выбираем длину сигнала $D = 16$. Можно заметить, что тогда сигнал $d$ из теоремы 9.5.18 будет равен

$$d = (33, 33, 32, 33, 32, 33, 32, 33, 33, 32, 33, 32, 33, 32, 33, 32),$$

а весовой сигнал будет

$$a = \left(1, 2^{7/16}, 2^{7/8}, 2^{5/16}, 2^{3/4}, 2^{3/16}, 2^{5/8}, 2^{1/16}, 2^{1/2}, 2^{15/16}, 2^{3/8}, 2^{13/16}, 2^{1/4}, 2^{11/16}, 2^{1/8}, 2^{9/16}\right)$$

При типичной реализации БПФ для вычислений с плавающей точкой сигнал $a$, разумеется, будет представлен неточными значениями. Но по теореме 9.5.18 взвешенная свертка (вычисленная прямо перед шагом [Округлить цифры] в алгоритме 9.5.19 приближенно) должна состоять из целых чисел. Поэтому задача состоит в том, чтобы выбрать длину сигнала $D$ настолько маленькой, насколько это возможно (чем она будет меньше, тем быстрее будут выполняться БПФ, входящие в вычисление ВДП), не позволяя в то же время ошибкам округления привести к неправильным цифрам $z$. Строгие теоремы об ошибке округления найти сложно, хотя несколько подходов — часть из них строгие, часть не совсем — разбирается в работе [Crandall and Fagin 1994] и содержащихся в ней ссылках. Современные методы отражены в очень полезной книге [Higham 1996] и работе [Percival 2000] по поводу обобщенного ВДПИО; см. упражнение 9.48.

### 9.5.5. Теоретико-числовые преобразования

Дискретное преобразование Фурье (ДПФ) из определения 9.5.3 можно определять над кольцами и полями, отличными от привычного нам поля комплексных чисел. В этом разделе мы приведем несколько примеров преобразований над конечными кольцами и полями. Первым замечанием будет то, что над кольцом или полем определяющие соотношения (9.20) и (9.21) для ДПФ не требуют никаких изменений, поскольку мы понимаем, что все требуемые операции

в кольце или поле выполнимы. В частности, теоретико-числовое ДПФ длины $D$ позволяет найти циклическую свертку длины $D$ благодаря теореме 9.5.11, как только в кольце существуют обратный элемент $D^{-1}$ и $g$, примитивный корень порядка $D$ из единицы. Благодаря этим факторам, теоретико-числовые преобразования заняли достойное место среди быстрых алгоритмов в области обработки сигналов. В литературе можно найти не только их связь с обычными свертками, но также и другие интересные приложения. Типичным примером является использование теоретико-числовых преобразований для проведения классических алгебраических операций [Yagle 1995], и еще больше методов приложений собрано в [Madisetti and Williams 1997].

Нашим первым примером будет случай, когда рассматриваемым полем является $\mathbf{F}_p$. Для некоторого простого $p$, и $d \mid p-1$ выберем в качестве поля $\mathbf{F}_p$, и рассмотрим преобразование

$$X_k = \sum_{j=0}^{(p-1)/d-1} x_j h^{-jk} \bmod p, \tag{9.33}$$

где $h$ — элемент, имеющий мультипликативный порядок $(p-1)/d$ в поле $\mathbf{F}_p$. Обратите внимание, что операция mod может в принципе применяться к каждому отдельному слагаемому, или ко всей сумме, или к отдельным ее частям, так что мы для простоты пишем «mod$p$», чтобы обозначить, что результаты преобразования берутся из отрезка $[0, p-1]$. Обратным преобразованием является

$$x_j = -d \sum_{k=0}^{(p-1)/d-1} X_k h^{jk} \bmod p, \tag{9.34}$$

где множитель перед суммой равен $((p-1)/d)^{-1} \bmod p \equiv -d$. Эти преобразования могут быть использованы для увеличения точности при вычислении сверток. Идея заключается в том, чтобы взять каждый элемент свертки по модулю $p_r$ для какого-либо удобного набора простых чисел $\{p_r\}$, тогда точные значения свертки смогут быть восстановлены с помощью китайской теоремы об остатках.

**Алгоритм 9.5.20** (целочисленная свертка с помощью китайской теоремы об остатках для набора простых чисел). Пусть заданы два сигнала $x, y$ длины $N = 2^m$, имеющих целочисленные цифры, лежащие в границах $0 \leqslant x_j, y_j < M$. Этот алгоритм возвращает циклическую свертку $x \times y$, вычисленную с использованием китайской теоремы об остатках по различным простым модулям $p_1, p_2, \ldots, p_q$.

1. [Начальная установка]
   Найти набор простых чисел вида $p_r = a_r N + 1$ для $r = 1, \ldots, q$, таких что $\prod p_r > NM^2$;
   for($1 \leqslant r \leqslant q$) {
   Найти примитивный корень $g_r$ из 1 для модуля $p_r$;
   $h_r = g_r^{a_r} \bmod p_r$; // $h_r$ является корнем степени $N$ из 1.
   }
2. [Цикл по простым числам]

for($1 \leqslant r \leqslant q$) {
  $h = h_r$; $p = p_r$; $d = a_r$; // Подготовка к ДПФ.
  $X^{(r)} = \text{ДПФ}(x)$; // С помощью (9.33).
  $Y^{(r)} = \text{ДПФ}(y)$;

3. [Покомпонентное умножение]
  $Z^{(r)} = X^{(r)} * Y^{(r)}$;

4. [Обратное преобразование]
  $z^{(r)} = \text{ВДП}^{-1}(Z^{(r)})$; // С помощью (9.34).

}

5. [Восстановить элементы]
  Исходя из соотношений $z_j \equiv z_j^{(r)} \pmod{p_r}$, найти единственный элемент $z_j$, лежащий в интервале $[0, NM^2)$, с помощью китайской теоремы об остатках. Можно использовать такой алгоритм, как 2.1.7 или 9.5.26;
  return $z$;

Главное, что этот алгоритм позволяет делать — это использовать БПФ длины $2^m$ для вычисления ДПФ и обратного преобразования, за исключением того, что в обычных «бабочках» БПФ будет использоваться только целочисленная арифметика (и, конечно, все слагаемые в ходе вычисления должны постоянно приводиться по модулю $p_r$). Эта схема была успешно использована в [Montgomery 1992a] в различных приложениях для факторизации. Заметим, что если прямое ДПФ (9.33) выполняется с помощью алгоритма «прореживания по частоте», а обратное преобразование (9.34) с помощью «прореживания по времени», то в вызове функции перемешивания битов в алгоритме 9.5.5 ни для одного из преобразований нет необходимости.

Вторым примером полезного теоретико-числового преобразования является дискретное преобразование Галуа (ДПГ) [Crandall 1996a], где в качестве поля, в котором будут проводиться вычисления, выбрано $\mathbf{F}_{p^2}$, а $p = 2^q - 1$ — простое число Мерсенна. Приятным фактом о таких полях является то, что порядок мультипликативной группы равен

$$|\mathbf{F}_{p^2}^*| = p^2 - 1 = 2^{q+1}(2^{q-1} - 1),$$

так что можно находить примитивные корни из единицы порядка $N = 2^k$ все время, пока $k \leqslant q + 1$. Следовательно, для таких длин мы можем определить дискретное преобразование

$$X_k = \sum_{j=0}^{N-1} x_j h^{-jk} \bmod p, \tag{9.35}$$

где на этот раз предполагается, благодаря известной структуре полей $\mathbf{F}_{p^2}$ для простых чисел $p$ вида $p \equiv 3 \pmod 4$, что все вычисления используют комплексные целые числа (гауссовы числа) по модулю $p$, и

$$\begin{aligned} N &= 2^k, \\ x_j &= \operatorname{Re}(x_j) + i \operatorname{Im}(x_j), \\ h &= \operatorname{Re}(h) + i \operatorname{Im}(h), \end{aligned}$$

где последнее число является элементом с мультипликативным порядком $N$ в поле $\mathbf{F}_{p^2}$, а элементы преобразования $X_k$ являются гауссовыми целыми числами по модулю $p$. К счастью, существует способ, позволяющий быстро найти элемент подходящего порядка, благодаря следующему результату [Creutzburg and Tasche 1989]:

**Теорема 9.5.21** (Кройцбург и Таше). *Пусть $p = 2^q - 1$ — простое число Мерсенна, где $q$ — нечетное. Тогда элемент*

$$g = 2^{2^{q-2}} + i(-3)^{2^{q-2}}$$

*имеет порядок $2^{q+1}$ в группе $\mathbf{F}^*_{p^2}$.*

Эти наблюдения ведут к следующему алгоритму вычисления целочисленной свертки, в котором мы также упоминаем возможные улучшения, позволяющие уменьшить количество вычислений с комплексными числами. В частности, мы используем тот факт, что целочисленные сигналы вещественны, так что их мнимые части исчезают, и длина преобразования, таким образом, может быть уменьшена вдвое:

**Алгоритм 9.5.22** (свертка с помощью ДПГ (Крэндалл)). Пусть заданы два сигнала $x, y$ длины $N = 2^k \geqslant 2$, чьи элементы являются целыми числами из отрезка $[0, M]$. Этот алгоритм возвращает целочисленную свертку $x \times y$. В алгоритме используется метод вычисления свертки с помощью «дискретного преобразования Галуа» (ДПГ).

1. [Начальная установка]
   Выбрать простое число Мерсенна $p = 2^q - 1$, такое что $p > NM^2$ и $q > k$;
   С помощью теоремы 9.5.21 найти элемент $g$ порядка $2^{q+1}$;
   $h = g^{2^{q+2-k}}$; // $h$ имеет порядок $N/2$.
2. [Сложить половинки сигналов, чтобы сократить длину]
   $x = \big(x_{2j} + ix_{2j+1}\big),\ j = 0, \ldots, N/2 - 1$;
   $y = \big(y_{2j} + iy_{2j+1}\big),\ j = 0, \ldots, N/2 - 1$;
3. [Выполнить преобразования длины $N/2$]
   $X = \text{ДПФ}(x)$;
   // С помощью, например, БПФ с разделенным основанием $(\bmod p)$ с корнем $h$.
   $Y = \text{ДПФ}(y)$;
4. [Покомпонентное произведение]
   for$(0 \leqslant k < N/2)$ {
   $Z_k = (X_k + X^*_{-k})(Y_k + Y^*_{-k}) + 2(X_kY_k - X^*_{-k}Y^*_{-k}) - h^{-k}(X_k - X^*_{-k})(Y_k - Y^*_{-k})$;
   }
5. [Обратное преобразование длины $N/2$]
   $z = \frac{1}{4}\text{ДПФ}^{-1}(Z)$;
   // С помощью БПФ с разделенным основанием $(\bmod p)$ с корнем $h$.
6. [Восстановить сигнал]
   $z = \big((\mathrm{Re}(z_j), \mathrm{Im}(z_j))\big),\ j = 0, \ldots, N/2 - 1$;
   return $z$;

Для того чтобы реализовать этот алгоритм, необходимы только алгоритм комплексного целочисленного БПФ по модулю $p$, комплексное умножение по модулю $p$ и двоичная схема возведения в степень в этом поле. Хотя обычно упомянутый в алгоритме метод БПФ с разделенным основанием используется в связи с обычным БПФ с плавающей точкой, его, однако, можно использовать в данной ситуации, так как «i» определено [Crandall 1997b].

Имеется еще один важный аспект метода сверток, использующего ДПГ: все операции приведения по модулю выполняются относительно простых чисел Мерсенна, так что можно получить существенное улучшение скорости, с которым мы имели дело ранее для модулей особого вида.

## 9.5.6. Метод Шёнхаге

В оригинальных работах [Schönhage and Strassen 1971], [Schönhage 1982], основанных на идеях Штрассена об использовании БПФ для умножения больших чисел, внимание заостряется на том факте, что в кольце $\mathbf{Z}_{2^m+1}$ возможно определенное теоретико-числовое преобразование, использующее только целочисленные операции. Его иногда называют числовым преобразованием Ферма (ЧПФ) (см. упражнение 9.52), которое можно использовать в одном подходе для вычисления негациклической свертки следующим образом (мы благодарны Циммерману за предоставление явного вида этого метода, из которого мы вывели наше описание):

**Алгоритм 9.5.23** (быстрое умножение $\pmod{2^n+1}$ (Шёнхаге)). Пусть заданы два целых числа $0 \leqslant x, y < 2^n + 1$. Этот алгоритм возвращает их произведение $xy \bmod (2^n + 1)$.

1. [Начальная установка]
   Выбрать размер БПФ, равный $D = 2^k$, делящий $n$;
   Написав $n = DM$, установить длину рекурсии $n' \geqslant 2M + k$, такую что $D$ делит $n'$, т. е. $n' = DM'$;
2. [Декомпозиция]
   Разделить $x$ и $y$ на $D$ частей по $M$ битов каждая, и сохранить эти части, считающиеся за остатки по модулю $(2^{n'} + 1)$, в два массива $A_0, \ldots, A_{D-1}$ и $B_0, \ldots, B_{D-1}$, позаботившись о том, что в принципе в дальнейшем элементы массивов могут иметь $n' + 1$ битов;
3. [Подготовиться к ВДП, добавив вес к сигналам $A, B$]
   for$(0 \leqslant j < D)$ {
   $A_j = (2^{jM'} A_j) \bmod (2^{n'} + 1)$;
   $B_j = (2^{jM'} B_j) \bmod (2^{n'} + 1)$;
   }
4. [Выполнить символическое БПФ, сохраняющее результат на месте]
   $A = \text{ДПФ}(A)$;
   $B = \text{ДПФ}(B)$;

// Использовать в качестве корня степени $D$ по модулю $2^{n'}+1$ число $2^{2M'}$.

5. [Покомпонентное умножение]
   for($0 \leqslant j < D$) $A_j = A_j B_j \bmod (2^{n'}+1)$;
6. [Обратное преобразование]
   $A = \text{ДПФ}^{-1}(A)$; // Обратить с помощью перестановки индексов, следующий цикл.
7. [Нормализация]
   for($0 \leqslant j < D$) { // $A_D$ определено как $A_0$.
   $C_j = A_{D-j}/2^{k+jM'} \bmod (2^{n'}+1)$;
8. [Определить знаки]
   if($C_j > (j+1)2^{2M}$) $C_j = C_j - (2^{n'}+1)$;
   } // $C_j$ теперь может быть отрицательным.
9. [Композиция]
   Выполнить переносы, как в шаге [Выполнить переносы по основанию $B$] для $B = 2^M$ (исходное основание декомпозиции) и в шаге [Учет последней цифры] алгоритма 9.5.17, и вернуть требуемую сумму:
   $xy \bmod (2^n+1) = \sum_{j=0}^{D-1} C_j 2^{jM} \bmod (2^n+1)$;

Обратите внимание, что в шаге [Декомпозиция] число $A_{D-1}$ или $B_{D-1}$ может равняться $2^M$ и иметь $M+1$ битов, в случае если $x$ или $y$ равнялось $2^n$. В шаге [Подготовиться к ВДП ...] каждое умножение может быть сделано с помощью одних сдвигов и вычитаний, так как $2^{n'} \equiv -1 (\bmod 2^{n'}+1)$. В шаге [Покомпонентное умножение] можно использовать любой алгоритм умножения, например, школьный метод, метод Карацубы, или рекурсивно сам метод Шёнхаге. В шаге [Нормализация] деление на степень двойки снова может быть сделано одними сдвигами и вычитаниями. Поэтому собственно умножение содержатся лишь в шаге [Покомпонентное умножение], и именно из-за этого метод может достигать столь низкой сложности. Обратите внимание, что два преобразования БПФ, требующиеся для получения негациклической свертки $C$, можно выполнять в порядке: «прореживание по частоте», «прореживание по времени», что тем самым избавляет нас от необходимости вообще выполнять какие-либо операции по перемешиванию битов.

Без сомнения, алгоритм 9.5.23 способен умножить два числа по модулю любого числа Ферма, и такое применение важно, как объясняется в других разделах этой книги. Для умножения двух чисел $x, y$ в общем случае можно выполнить метод Шёнхаге с $n \geqslant \lceil \lg x \rceil + \lceil \lg y \rceil$ и соответствующим дополнением $x$ и $y$ нулями, тогда произведение $xy \bmod 2^n+1$ будет равно целочисленному произведению. (Короче говоря, негациклическая свертка правильно дополненных нулями последовательностей является ациклической сверткой — по сути, результатом умножения.) На практике Шёнхаге предлагает использование того, что он называет «подходящими числами», т. е. $n = \nu 2^k$, где $k-1 \leqslant \nu \leqslant 2k-1$. Например, $688128 = 21 \cdot 2^{15}$ является подходящим числом. Такие числа обладают тем свойством, что если $k = \lceil n/2 \rceil + 1$, то $n' = \lceil \frac{\nu+1}{2} \rceil 2^k$ тоже является подходящим числом; в нашем примере мы и вправду получаем $n' = 11 \cdot 2^8 = 2816$.

Конечно, что касается умножения, изначально теряется множитель 2, но в рекурсивных вызовах все вычисления выполняются по модулю некоторого $2^M+1$, так что асимптотическая сложность в точности та, что приводится в разд. 9.5.8.

## 9.5.7. Метод Нюссбаумера

Важным наблюдением является то, что циклическая свертка четной длины $D$ может быть выражена через пару сверток: циклическую и негациклическую, каждая длины $D$. Главным равенством является

$$2(x \times y) = [(u_+ \times v_+) + (u_- \times_- v_-)] \cup [(u_+ \times v_+) - (u_- \times_- v_-)], \qquad (9.36)$$

где сигналы $u, v$ являются производными от половинок исходных сигналов:

$$u_\pm = L(x) \pm H(x),$$
$$v_\pm = L(y) \pm H(y).$$

Эту рекурсивную формулу для вычисления циклической свертки можно доказать с помощью операций над многочленами (см. упражнение 9.42). Это рекурсивное соотношение вместе с одним хитроумным алгебраическим замечанием привела Нюссбаумера [Nussbaumer 1981] к эффективной схеме вычисления свертки, не содержащей преобразования Фурье с плавающей точкой. Этот алгоритм, таким образом, свободен от проблем с ошибкой округления, и потому его часто выбирают в качестве метода для строгих машинных доказательств, требующих вычислений с большими числами.

Глядя на рекурсивное равенство, мы понимаем, что если бы только у нас был быстрый алгоритм вычисления негациклической свертки, то циклическую свертку можно было бы найти напрямую, во многом так же, как это делает метод БПФ с помощью прореживания по длине сигналов. Для этой цели обозначим через $R$ кольцо, где 2 обратимо; т. е. всякий раз, когда $2x = 2y$, верно также и $x = y$. (Интересно, что это и есть все требования, необходимые для того, чтобы метод заработал.) Предполагается, что длина негациклической свертки равна $D = 2^k$, и что $D$ разлагается на множители $D = mr$, где $m|r$. После этого негациклическая свертка равносильна умножению многочленов по модулю $t^D + 1$ (см. упражнения), и может в определенном смысле быть разложена на операции, как указано далее:

**Теорема 9.5.24** (Нюссбаумер). *Пусть $D = 2^k = mr$, $m|r$. Тогда негациклическая свертка сигналов длины $D$, элементы которых принадлежат кольцу $R$, равносильна (в том смысле, что коэффициенты многочленов соответствуют элементам сигналов) умножению в кольце многочленов*

$$S = R[t]/\left(t^D + 1\right).$$

*Далее, это кольцо изоморфно*

$$T[t]/\left(z - t^m\right),$$

*где $T$ есть кольцо многочленов*

$$T = R[z]/(z^r + 1).$$

*Наконец, $z^{r/m}$ — корень степени $m$ из $-1$ в кольце $T$.*

Идея Нюссбаумера заключается в том, чтобы использовать корень из $-1$ методом, похожим на использование ВДП, чтобы вычислить негациклическую свертку.

Давайте рассмотрим явные преобразования многочленов, чтобы пояснить суть теоремы 9.5.24. Пусть

$$x(t) = x_0 + x_1 t + \ldots + x_{D-1} t^{D-1},$$

и задан аналогичный многочлен для сигнала $y$, где коэффициенты $x_j, y_j$ лежат в кольце $R$. Обратите внимание, что $x \times_- y$ равносильно умножению $x(t)y(t)$ в кольце $S$. Теперь разложим

$$x(t) = \sum_{j=0}^{m-1} X_j(t^m) t^j,$$

и аналогично для $y(t)$, и станем понимать каждый из многочленов $X_j, Y_j$ как элементы кольца $T$; тогда

$$X_j(z) = x_j + x_{j+m} z + \ldots + x_{j+m(r-1)} z^{r-1},$$

и так же для $Y_j$. Ясно, что все $2m$ многочленов $X, Y$ можно сохранить в двух массивах, которые являются перестановками массивов для $x, y$ по правилу транспонирования прямоугольника размера $r \times m$. Далее мы выполняем умножение $x(t)y(t)$ с помощью вычисления циклической свертки

$$Z = (X_0, X_1, \ldots, X_{m-1}, 0, \ldots, 0) \times (Y_0, Y_1, \ldots, Y_{m-1}, 0, \ldots, 0),$$

где каждый операнд был дополнен нулями до длины $2m$. Ключевым моментом здесь является то, что $Z$ может быть вычислено с помощью *символического* ДПФ с использованием числа, о котором мы знаем, что оно является примитивным корнем степени $2m$ из единицы, а именно, $z^{r/m}$. Это на самом деле означает, что обычные операции перестановки в методе БПФ теперь будут требовать всего лишь курсирования вокруг многочленов, поскольку умножение на степени примитивного корня просто сдвигает коэффициенты многочлена. Другими словами, вычисления с многочленами далее следуют в соответствии с теоремой 9.2.12, в которой умножение на степень рассматриваемого корня равносильно операции сдвига.

В главном этапе обычного метода для нахождения свертки с помощью ДПФ, а именно, в вычислении покомпонентного произведения, покомпонентные операции сами по себе оказываются, как можно видеть, негациклическими свертками длины $r$. Это явствует из того, что каждый многочлен $X_j, Y_j$ имеет степень $(r-1)$ относительно переменной $z = t^m$, так что $z^r = t^D = -1$. Для того чтобы завершить вычисление свертки $Z$, надо использовать окончательное обратное преобразование ДПФ с корнем $z^{-r/m}$. Как видно, результатом этой

дополненной нулями свертки будет произведение в кольце $S$:

$$x(t)y(t) = \sum_{j=0}^{2m-2} Z_j(t^m)t^j, \tag{9.37}$$

из которого мы получаем элементы негациклической свертки $x \times_- y$ в виде коэффициентов при степенях $t$.

**Алгоритм 9.5.25** (циклическая и негациклическая свертка, метод Нюссбаумера). Пусть заданы сигналы $x, y$ длины $(D = 2^k)$, чьи элементы лежат в кольце $R$, и в этом кольце возможно сокращение на два. Этот алгоритм возвращает циклическую $(x \times y)$ или негациклическую $(x \times_- y)$ свертку. Рекурсивная функция $neg$ вызывает сама себя, а после определенного порога — функцию для малых $D$, $smallneg$, которую можно вычислять с помощью школьного метода или метода Карацубы.

1. [Начальная установка]
   $r = 2^{\lceil k/2 \rceil}$;
   $m = D/r$; // $m$ является делителем $r$.
   $blen = 16$; // Пороговое значение длины для вызова $smallneg$.
2. [Рекурсивная функция циклической свертки $cyc$]
   $cyc(x, y)$ {
   С помощью вызовов функции для циклической и негациклической свертки половинной длины и равенства (9.36) вернуть требуемую циклическую свертку;
   }
3. [Рекурсивная функция негациклической свертки $neg$]
   $neg(x, y)$ {
   if$(len(x) \leqslant blen)$ return $smallneg(x, y)$;
4. [Шаг транспозиции]
   Создать $2m$ массивов $X_j, Y_j$, каждый длины $r$;
   Дополнить семейство многочленов $X, Y$ нулевыми многочленами, так чтобы каждое семейство содержало $2m$ многочленов;
   Используя корень $g = z^{r/m}$, выполнить (символически) два ДПФ, длины $2m$, дающие результаты $\hat{X}, \hat{Y}$;
5. [Рекурсивное покомпонентное умножение]
   for$(0 \leqslant h < 2m)$ $\hat{Z}_h = neg(\hat{X}_h, \hat{Y}_h)$;
6. [Обратное преобразование]
   Используя корень $g = z^{-r/m}$, выполнить (символически) обратное преобразование ВДП длины $2m$ к результату $\hat{Z}$, и получить $Z$;
7. [Обратная транспозиция и восстановление]
   Работая в кольце $S$ (т.е. понижая многочлены с помощью равенства $t^D = -1$), найти коэффициенты $z_n$ при степенях $t^n$, $n \in [0, D-1]$, из равенства (9.37);
   return $(z_n)$; // Вернуть негациклическую свертку $x, y$.
   }

Подробную реализацию этого выдающегося метода Нюссбаумера можно найти в [Crandall 1996a], где обсуждаются также его улучшения. Одно из таких улучшений позволяет обойтись без дополнения нулями семейств $X, Y$ (см. по этому поводу упражнение 9.66). Другое заключается в том, чтобы заметить, что структура $X_j, Y_j$ означает транспозицию двумерного массива, и возможно существенно снизить затраты памяти, если выполнять такие транспозиции на том же месте. В [Knuth 1981] описаны алгоритмы транспозиций, сохраняющих результат на месте. Также интересным является алгоритм Фрейзера [Fraser 1976], упомянутый в связи с алгоритмом 9.5.7.

### 9.5.8. Сложность алгоритмов умножения

Для того чтобы свести результаты для сложности упомянутых ранее быстрых алгоритмов умножения, надо обсудить обозначения. В основном мы говорим о перемножаемых операндах $x, y$, имеющих $n$ двоичных битов, или размер $N = 2^n$, или $D$ цифр; далее это означает одно и то же. Так, например, если цифры берутся для основания системы счисления $B = 2^b$, то

$$Db \approx n$$

обозначает, что $n$ битов операндов разделены на $D$ элементов сигналов. Это обозначение полезно, так как нам необходимо различать сложность битовую и сложность операций в кольце.

Напомним, что оценки сложности школьного метода умножения, метода Карацубы и метода Тоома—Кука все имеют вид $O(n^\alpha) = O(\ln^\alpha N)$ *битовых* операций для выполнения всех содержащихся в них умножений. (Мы формулируем этот результат так, потому что в случае Тоома—Кука следует также считать и большое количество битовых операций из-за сложения.) Так, например, для школьного метода $\alpha = 2$, метод Карацубы и Тоома—Кука несколько понижают $\alpha$, и т. д.

Теперь рассмотрим базовый метод умножения Шёнхаге—Штрассена с использованием БПФ и его алгоритм 9.5.12. Неожиданно оказывается, что его сложность имеет другую природу, поскольку мы знаем, что сложность должна быть

$$O(D \ln D)$$

*операций в кольце*, и, как мы заметили раньше, обычно это операции с плавающей точкой (причем оценка верна и для количества сложений, и для количества умножений). Тогда *битовая* сложность не равна $O((n/b)\ln(n/b))$ — мы не можем просто подставить $D = n/b$ в эту оценку сложности, из-за того что вычисления с плавающей точкой на большем количестве цифр требуют больше битовых операций. При правильном анализе метода можно получить оценку Штрассена вида

$$O(n(C \ln n)(C \ln \ln n)(C \ln \ln \ln n) \ldots)$$

побитовых операций для метода умножения с помощью БПФ, где $C$ — константа, а цепочка из членов вида $\ln \ln \ldots$ прекращается, когда очередной член ста-

нет меньше единицы. Перед тем как переходить к другим оценкам, мы должны подчеркнуть, что хотя эта битовая сложность не является асимптотически оптимальной, некоторые из лучших достижений в области операций с большими числами были достигнуты именно с помощью базового метода умножения Шёнхаге—Штрассена с использованием БПФ, причем с использованием чисел с плавающей точкой.

Алгоритм Шёнхаге 9.5.23 успешно избегает той проблемы, что для фиксированного числа цифр сигнала $D$ операции с цифрами (малые умножения) оказываются сложнее, чем для больших операндов. Анализ рекурсии внутри алгоритма начинается с замечания, что на верхнем уровне рекурсии имеется два преобразования ДПФ (однако очень простых — происходят только сдвиги и сложения) и покомпонентное умножение. Подробный анализ дает лучшую на текущий момент оценку сложности:

$$O(n(\ln n)(\ln\ln n))$$

битовых операций, хотя сложность метода Нюссбаумера, которую мы обсудим ниже, ей асимптотически эквивалентна.

Далее, можно заметить (как это видно из упражнения 9.67), что сложность вычисления свертки методом Нюссбаумера есть

$$O(D \ln D)$$

операций в кольце $R$. Это эквивалентно сложности быстрого преобразования Фурье с плавающей точкой, если предполагается, что операции в кольце равноценны операциям с плавающей точкой. Однако в случае метода Нюссбаумера имеется одно отличие: можно безнаказанно выбирать основание системы счисления $B$ произвольным образом. Рассмотрим основание системы счисления $B \sim n$, так что $b \sim \ln n$, в случае чего можно эффективно использовать $D = n/\ln n$ цифр. Оказывается, что тогда метод Нюссбаумера целочисленного умножения требует $O(n \ln\ln n)$ сложений и $O(n)$ умножений чисел, имеющих $O(\ln n)$ битов. Отсюда следует, что сложность метода Нюссбаумера асимптотически равна сложности метода Шёнхаге, т. е. составляет $O(n \ln n \ln\ln n)$ битовых операций. Вопросы сложности для методов Нюссбаумера и исходного метода Шёнхаге—Штрассена обсуждаются в [Bernstein 1997]. В табл. 9.1 приведены сложности алгоритмов быстрого умножения.

## 9.5.9. Приложение к китайской теореме об остатках

В разделе 2.1.3 мы упомянули китайскую теорему об остатках и дали алгоритм 2.1.7 для восстановления значений с помощью китайской теоремы об остатках, при условии что выполнены определенные предварительные вычисления. Сейчас мы опишем метод, который использует не только преимущества предварительно заданных условий, но также и алгоритмы быстрого умножения.

**Алгоритм 9.5.26** (быстрое восстановление по китайской теореме об остатках с предварительными вычислениями). Используя обозначения из теоремы 2.1.6,

| Алгоритм | лучшее $B$ | сложность |
|---|---|---|
| БПФ с фиксированным основанием | ... | $O_{\text{op}}(D \ln D)$ |
| БПФ с переменным основанием | $O(\ln n)$ | $O(n(C \ln n)(C \ln \ln n)\ldots)$ |
| Метод Шёнхаге | $O(n^{1/2})$ | $O(n \ln n \ln \ln n)$ |
| Метод Нюссбаумера | $O(n/\ln n)$ | $O(n \ln n \ln \ln n)$ |

**Таблица 9.1.** Сложности алгоритмов быстрого умножения. Операнды имеют $n$ битов, которые в ходе рекурсии разбиваются на $D = n/b$ цифр, каждая длиной $b$ битов, так что размер цифр (равный основанию системы счисления) равен $B = 2^b$. Все оценки даются для битовой сложности, за исключением оценки $O_{\text{op}}$, которая означает сложность в операциях в кольце.

мы предполагаем, что заданы фиксированные модули $m_0, \ldots, m_{r-1}$, произведение которых равно $M$, однако положим для удобства $r = 2^k$. Задача алгоритма заключается в том, чтобы восстановить число $n$ по его заданным остаткам $(n_i)$ по модулям $(m_i)$. В ходе алгоритма вычисляется таблица $(q_{ij})$ частичных произведений и таблица $(n_{ij})$ частичных остатков. Алгоритм допускает повторный вход с другим $n$, если $m_i$ остаются теми же.

1. [Предварительные вычисления]
   for($0 \leqslant i < r$) { // Посчитать $M_i$ и их обратные.
   $M_i = M/m_i$;
   $v_i = M_i^{-1} \bmod m_i$;
   }
   for($0 \leqslant j \leqslant k$) { // Посчитать частичные произведения.
   for($0 \leqslant i \leqslant r - 2^j$) $q_{ij} = \prod_{a=i}^{i+2^j-1} m_a$;
   }
2. [Точка повторного входа, если заданы остатки $(n_i)$]
   for($0 \leqslant i < r$) $n_{i0} = v_i n_i$;
3. [Цикл реконструкции]
   for($1 \leqslant j \leqslant k$) {
   for($i = 0$; $i < r$; $i = i + 2^j$) $n_{ij} = n_{i,j-1} q_{i+2^{j-1},j-1} + n_{i+2^{j-1},j-1} q_{i,j-1}$;
   }
4. [Вернуть единственное $n$ из отрезка $[0, M-1]$]
   return $n_{0k} \bmod q_{0k}$;

Обратите внимание, что первую фазу алгоритма, делающую предварительные вычисления, достаточно выполнить лишь один раз, когда определенный набор остатков $(n_i)$ используется в первый раз. Также отметьте, что предварительное вычисление $(q_{ij})$ само по себе может выполняться с помощью быстрого алгоритма типа «разделяй и властвуй», наподобие того, что обсуждается в этой главе (пример см. в упражнении 9.74). Как пример действия алгоритма 9.5.26, давайте возьмем $r = 8 = 2^3$ и восемь модулей $(m_1, \ldots, m_8) = (3, 5, 7, 11, 13, 17, 19, 23)$. Мы используем эти модули вместе с их произведением $M = \prod_{i=1}^{8} m_i = 111546435$ и получаем на шаге [Предварительные вычисления]

числа $M_1, \ldots, M_8$, равные

$$37182145, 22309287, 15935205, 10140585, 8580495, 6561555, 5870865, 4849845,$$

$$(v_1, \ldots, v_8) = (1, 3, 6, 3, 1, 11, 9, 17),$$

и таблицу

$$\begin{aligned}
(q_{00}, \ldots, q_{70}) &= (3, 5, 7, 11, 13, 17, 19, 23), \\
(q_{01}, \ldots, q_{61}) &= (15, 35, 77, 143, 221, 323, 437), \\
(q_{02}, \ldots, q_{42}) &= (1155, 5005, 17017, 46189, 96577), \\
q_{03} &= 111546435,
\end{aligned}$$

где, напоминаем, для фиксированного $j$ существуют $q_{ij}$ при $i \in [0, r - 2^j]$. Важно помнить, что все вычисления вплоть до вычисления таблицы $q$ достаточно проделать лишь один раз — на все время, пока модули китайской теоремы об остатках $m_i$ остаются неизменными. Далее, когда требуется обработать конкретные остатки $n_i$ некоторого неизвестного числа $n$, например,

$$(n_1, \ldots, n_8) = (1, 1, 1, 1, 3, 3, 3, 3),$$

то после шага [Цикл реконструкции] величина

$$n_{0k} = 878271241$$

после приведения по модулю $q_{03}$ дает правильный результат $n = 97446196$. В самом деле, простая проверка показывает, что

$$97446196 \bmod (3, 5, 7, 11, 13, 17, 19, 23) = (1, 1, 1, 1, 3, 3, 3, 3).$$

Для сложности алгоритма 9.5.26 известна оценка следующего вида [Aho et al. 1974, pp. 294–298], в предположении, что используется быстрое умножение. Если каждый из $r$ модулей $m_i$ имеет $b$ битов, то сложность оценивается как

$$O(br \ln r \ln(br) \ln \ln(br))$$

битовых операций, в предположении, что все предварительные вычисления для алгоритма уже сделаны.

## 9.6. Операции с многочленами

Важным наблюдением является то, что умножение и деление многочленов — не то же самое, что умножение и деление больших чисел. Тем не менее, можно несколько иначе использовать те же идеи, что были использованы в предыдущих разделах, в области одномерных многочленов.

### 9.6.1. Умножение многочленов

Мы видели, что умножение многочленов равносильно нахождению ациклической свертки. Следовательно, умножение двух многочленов можно провести с помощью циклической и негациклической сверток. Надо просто построить

соответствующие сигналы, содержащие коэффициенты многочленов, и применить теорему 9.5.10. Альтернативный подход — можно просто дополнить нулями сигналы до их удвоенной длины и выполнить одну циклическую (или одну негациклическую) свертку.

Однако имеется один интересный (и часто довольно эффективный) способ умножения многочленов, если задан общий метод умножения целых чисел. Этот метод заключается в том, чтобы определенным образом расположить коэффициенты многочленов в больших числах и выполнить все вычисления с помощью одного большого целочисленного умножения. Мы представляем этот алгоритм для случая, когда все коэффициенты многочленов неотрицательны, хотя это условие несущественно в случае умножения в кольцах многочленов по модулю $p$:

**Алгоритм 9.6.1** (быстрое умножение многочленов: двоичное разбиение). Пусть заданы два многочлена $x(t) = \sum_{j=0}^{D-1} x_j t^j$ и $y(t) = \sum_{k=0}^{E-1} y_k t^k$, где все коэффициенты — целые неотрицательные числа. Этот алгоритм возвращает произведение многочленов $z(t) = x(t)y(t)$ в виде сигнала, содержащего коэффициенты многочлена $z$.

1. [Начальная установка]
   Выбрать $b$ таким, чтобы $2^b > \max\{D, E\} \max\{x_j\} \max\{y_k\}$;
2. [Создать двоичные разбиения]
   $X = x(2^b)$;
   $Y = y(2^b)$;
   // Эти $X, Y$ можно построить, создав массивы из достаточного количества нулей и затем записав в них коэффициенты, начиная с правильных позиций.
3. [Умножить]
   $u = XY$; // Целочисленное умножение.
4. [Собрать сигнал из имеющихся коэффициентов]
   for($0 \leqslant l < D + E - 1$) {
   $z_l = \lfloor u/2^{bl} \rfloor \bmod 2^b$; // Добыть следующие $b$ битов.
   }
   return $z = \sum_{l=0}^{D-E-2} z_l t^l$;
   // Цифры числа $u$ по основанию $B$ — требуемые коэффициенты.

Этот подход хорош тем, что если метод для умножения больших целых чисел задан, то требуется не слишком много работы для построения метода умножения многочленов. Несложно показать, что битовая сложность умножения двух многочленов степени $D$ в кольце многочленов $\mathbf{Z}_m[X]$, т. е. когда все коэффициенты приводятся по модулю $m$, составляет

$$O\left(M\left(D \ln\left(Dm^2\right)\right)\right),$$

где $M(n)$, как всегда, обозначает битовую сложность умножения двух целых чисел длиной $n$ битов.

Кстати, если умножение многочленов в кольцах выполняется с помощью быстрых целочисленных сверток (напомним, что использования ациклической

свертки достаточно, дополненная нулями циклическая свертка тоже подойдет), то можно получить другое выражение для оценки сложности. Для алгоритма Нюссбаумера 9.5.25 требуется $O(M(p)D\ln D)$ битовых операций, где $M$ — обычная сложность перемножения целых чисел. Интересно сравнивать эти различные оценки для умножения многочленов (см. упражнение 9.70).

### 9.6.2. Быстрое обращение многочленов и деление с остатком

Пусть задан многочлен $x(t) = \sum_{j=0}^{D-1} x_j t^j$. Если $x_0 \neq 0$, то существует формальная обратная величина

$$1/x(t) = 1/x_0 - (x_1/x_0^2)t + (x_1^2/x_0^3 - x_2/x_0^2)t^2 + \ldots,$$

которая допускает быстрое вычисление с помощью схемы, которую мы уже использовали для поиска обратной величины — известный метод Ньютона. Мы опишем этот метод для случая, когда $x_0 = 1$, откуда легко получить обобщения на случай произвольного $x_0$. Далее мы используем обозначение

$$z(t) \bmod t^k$$

для деления многочленов с остатком (которое мы обсудим ниже), но в нашем случае деление является просто укорачиванием: результатом операции mod является многочлен, состоящий из членов многочлена $z(t)$ до порядка $t^{k-1}$ включительно. Определим укороченную обратную величину

$$R[x, N] = x(t)^{-1} \bmod t^{N+1}$$

как многочлен, получающийся из $1/x(t)$ отбрасыванием членов старше $t^N$.

**Алгоритм 9.6.2** (быстрое обращение многочлена). Пусть задан многочлен $x(t)$ с младшим коэффициентом $x_0 = 1$. Этот алгоритм возвращает обратную величину $R[x, N]$, укороченную до степени $N$.

1. [Начальная установка]
   $g(t) = 1$; // Многочлен степени 0.
   $n = 1$; // Рабочая степень многочлена.
2. [Цикл Ньютона]
   while($n < N + 1$) {
   $n = 2n$; // Удвоить рабочую степень многочлена.
   if($n > N + 1$) $n = N + 1$;
   $h(t) = x(t) \bmod t^n$; // Просто отбросить лишнее.
   $h(t) = h(t)g(t) \bmod t^n$;
   $g(t) = g(t)(2 - h(t)) \bmod t^n$; // Итерация Ньютона.
   }
   return $g(t)$;

Сразу следует указать, что в принципе операция $f(t)g(t) \bmod t^n$ является просто укорачиванием произведения многочленов (обычно сами операнды имеют степень примерно $n$). Это означает, что внутри циклов умножения не требуется заботиться о членах степени больше указанной. На языке теории сверток это означает, что мы выполняем «полуциклическую» свертку, так что если для вычисления используются методы с преобразованиями, также возникает экономия из-за отсекания старших степеней.

Для метода Ньютона характерно, что переменная степень многочлена $n$ по сути удваивается с каждым шагом цикла Ньютона. Давайте рассмотрим пример работы алгоритма. Возьмем многочлен

$$x(t) = 1 + t + t^2 + 4t^3$$

и вызовем алгоритм, чтобы получить результат $R[x, 8]$. Тогда величины $g(t)$ в конце каждого прохода цикла Ньютона окажутся такими:

$$\begin{aligned}
&1 - t,\\
&1 - t - 3t^3,\\
&1 - t - 3t^3 + 7t^4 - 4t^5 + 9t^6 - 33t^7,\\
&1 - t - 3t^3 + 7t^4 - 4t^5 + 9t^6 - 33t^7 + 40t^8,
\end{aligned}$$

и при этом в самом деле последнее значение $g(t)$, умноженное на исходный многочлен $x(t)$, дает $1 + 43t^9 - 92t^{10} + 160t^{11}$, что показывает, что последний результат для $g(t)$ точен вплоть до членов порядка $O(t^8)$.

Деление многочленов с остатком можно выполнять во многом так же, как и деление целых чисел с остатком, при помощи обратной величины. Однако не всегда возможно поделить один многочлен на другой и получить единственный правильный остаток: это может зависеть от кольца, откуда берутся коэффициенты многочленов. Тем не менее, если старший коэффициент многочлена-делителя обратим в этом кольце, то с делением и остатком не возникает проблем; см. обсуждение этого в разделе 2.2.1. Для простоты мы ограничимся случаем, когда старший коэффициент многочлена-делителя равен 1; после этого обобщения строятся мгновенно. Пусть $x(t), y(t)$ — заданные многочлены, и старший коэффициент $y(t)$ равен 1. Тогда существуют единственные многочлены $q(t), r(t)$, такие что

$$x(t) = q(t)y(t) + r(t),$$

и $r = 0$ либо $\deg(r) < \deg(x)$. Мы будем использовать обозначение

$$r(t) = x(t) \bmod y(t),$$

и считать $q(t)$ частным, а $r(t)$ — остатком от деления $x(t)$ на $y(t)$. Кстати, для некоторых операций с многочленами требуется, чтобы коэффициенты были элементами поля, как, например, в операции взятия НОД двух многочленов; однако многие операции с многочленами этого не требуют. Перед тем как продемонстрировать алгоритм быстрого деления многочленов с остатком, мы введем некоторые обозначения:

**Определение 9.6.3** (операции с многочленами). Пусть задан многочлен $x(t) = \sum_{j=0}^{D-1} x_j t^j$. Мы определяем обращение для многочлена $x$ степени $d$ как

$$rev(x, d) = \sum_{j=0}^{d} x_{d-j} t^j,$$

где считается, что $x_j = 0$ при всех $j > D - 1$. Мы также определяем операцию индекса многочлена как

$$ind(x, d) = \min\{j \ : \ j \geqslant d;\ x_j \neq 0\},$$

либо $ind(x, d) = 0$, если указанное множество пустое.

Например,

$$rev(1 + 3t^2 + 6t^3 + 9t^5 + t^6, 3) = 6 + 3t + t^3,$$
$$ind(1 + 3t^2 + 6t^3, 1) = 2.$$

Теперь можно привести алгоритм деления с остатком:

**Алгоритм 9.6.4** (быстрое деление многочленов с остатком). Пусть заданы многочлены $x(t), y(t)$, причем старший коэффициент $y(t)$ равен 1. Этот алгоритм возвращает остаток от деления $x(t) \bmod y(t)$.

1. [Начальная установка]
   if($\deg(y) == 0$) return 0;
   $d = \deg(x) - \deg(y)$;
   if($d < 0$) return $x$;
2. [Взять обращения]
   $X = rev(x, \deg(x))$;
   $Y = rev(y, \deg(y))$;
3. [Вычисление обратной величины]
   $q = R[Y, d]$; // С помощью алгоритма 9.6.2.
4. [Умножение и понижение]
   $q = (qX) \bmod t^{d+1}$; // Умножить и отсечь после степени $d$.
   $r = X - qY$;
   $i = ind(r, d + 1)$;
   $r = r/t^i$;
   return $rev(r, \deg(x) - i)$;

Доказательство того, что алгоритм работает правильно, довольно сложное, однако ясно, что здесь применяется основная идея Барретта для целочисленного деления; вычисление $r = X - qY$ подобно преобразованиям, которые делались с обобщенными обратными величинами в методе Барретта.

Что касается сложности алгоритма 9.6.4, отметим, что как и метод Барретта, вся процедура выполняется с помощью умножения многочленов. Таким образом, деление многочленов с остатком с помощью этого метода имеет ту же сложность, что и умножение многочленов.

Проблема быстрого вычисления НОД многочленов довольно интересна. Имеется прямой аналог алгоритма Евклида для поиска целочисленного НОД,

а именно, алгоритм 2.2.2. Далее, сложный рекурсивный алгоритм 9.4.6 неожиданно оказывается проще в применении к многочленам, чем к целым числам [Aho et al. 1974, pp. 300–310]. Мы хотели бы также указать, что некоторые авторы связывают рекурсивную идею, изначально задуманную для НОД многочленов, с работой [Moenck 1973]. Какой бы метод ни использовался для нахождения НОД многочленов, внутри него можно использовать алгоритм быстрого деления многочленов с остатком, приведенный в этом разделе.

## 9.6.3. Вычисление значений многочлена в наборе точек

Обсудим далее методы вычисления значений многочленов. Задача заключается в том, чтобы вычислить значения многочлена $x(t) = \sum_{j=0}^{D-1} x_j t^j$ в $n$ точках $t_0, \ldots, t_{n-1}$. Оказывается, что можно вычислить значения целой последовательности $(x(t_0), x(t_1), \ldots, x(t_{n-1}))$ за

$$O\left(n \ln^2 \min\{n, D\}\right)$$

операций над элементами поля. Разделим задачу на три основных случая:

(1) Значения $t_0, \ldots, t_{n-1}$ образуют арифметическую прогрессию.
(2) Значения $t_0, \ldots, t_{n-1}$ образуют геометрическую прогрессию.
(3) Значения $t_0, \ldots, t_{n-1}$ — произвольные.

Разумеется, случай (3) включает в себя два остальных, но в случаях (1) и (2) возможно будут применимы особые методы улучшения схемы.

**Алгоритм 9.6.5** (вычисление значений многочлена в точках арифметической прогрессии). Пусть задан многочлен $x(t) = \sum_{j=0}^{D-1} x_j t^j$. Этот алгоритм возвращает $n$ значений $x(a), x(a+d), x(a+2d), \ldots, x(a+(n-1)d)$. (Метод наиболее эффективен, когда $n$ много больше $D$.)

1. [Вычислить значение $x$ в первых $D$ точках]
   for($0 \leqslant j < D$) $e_j = x(a + jd)$;
2. [Построить таблицу разностей]
   for($1 \leqslant q < D$) {
   for($D - 1 \geqslant k \geqslant q$) $e_k = e_k - e_{k-1}$;
   }
3. [Преобразовать таблицу]
   $E_0 = e_0$;
   for($1 \leqslant q < n$) {
   $E_q = E_{q-1} + e_1$;
   for($1 \leqslant k < D - 1$) $e_k = e_k + e_{k+1}$;
   }
   return $(E_q)$, $q \in [0, n-1]$;

Вариант этого алгоритма был использован для поиска простых чисел Вильсона (см. упражнение 9.73, где также обсуждаются вопросы сложности алгоритма).

Далее, предположим, что значения $t_k$ образуют геометрическую прогрессию $t_k = T^k$, где $T$ — константа, так что нам требуется вычислить каждую из $D$ сумм $\sum x_j T^{kj}$, $k \in [0, D-1]$. Имеется так называемый прием Блюштейна, согласно которому такие суммы представляются в виде

$$\sum_j x_j T^{kj} = T^{-k^2/2} \sum_j \left(x_j T^{-j^2/2}\right) T^{(-k-j)^2/2},$$

и таким образом сумма в левой части выражается через свертку, в неявном виде записанную в правой части. Тем не менее, в некоторых условиях в чем-то удобнее оказывается избегать деления пополам квадратов в показателе, воспользовавшись вместо этого свойствами треугольных чисел $\Delta_n = n(n+1)/2$. Эти свойства записываются в виде равенств

$$\begin{aligned} \Delta_{\alpha+\beta} &= \Delta_\alpha + \Delta_\beta + \alpha\beta, \\ \Delta_\alpha &= \Delta_{-\alpha-1}. \end{aligned}$$

Соответственно, можно вывести вариант приема Блюштейна в виде

$$\sum_j x_j T^{kj} = T^{\Delta_{-k}} \sum_j \left(x_j T^{\Delta_j}\right) T^{-\Delta_{-(k-j)}}.$$

Теперь свертка может быть вычислена с использованием только целых степеней константы $T$. Более того, можно использовать эффективный метод для вычисления циклической свертки, вложив сигнал $x$ в более длинный, дополненный нулями сигнал, и выполнив переиндексацию, как это сделано в следующем алгоритме:

**Алгоритм 9.6.6** (вычисление значений многочлена в точках геометрической прогрессии). Пусть задан многочлен $x(t) = \sum_{j=0}^{D-1} x_j t^j$, а $T$ — константа, обратимая в рассматриваемом кольце. Этот алгоритм возвращает набор значений $\big(x(T^k)\big)$, $k \in [0, D-1]$.

1. [Начальная установка]
   Выбрать $N = 2^n$ таким, чтобы $N \geqslant 2D$;
   for($0 \leqslant j < D$) $x_j = x_j T^{\Delta_j}$; // Добавить вес к сигналу $x$.
   Дополнить $x = (x_j)$ нулями до длины $N$;
   $y = \big(T^{-\Delta_{N/2-j-1}}\big)$, $j \in [0, N-1]$; // Создать симметричный сигнал $y$.
2. [Циклическая свертка длины $N$]
   $z = x \times y$;
3. [Собрать результат вычислений]
   return $\big(x(T^k)\big) = \big(T^{\Delta_{k-1}} z_{N/2+k-1}\big)$, $k \in [0, D-1]$;

Как видно, требуется одна свертка для того, чтобы вычислить сразу все значения $x(T^k)$. Ясно, что сложность всего алгоритма составляет $O(D \ln D)$ операций в кольце. Важным наблюдением является то, что обыкновенное ДПФ является по сути вычислением многочлена на последовательности точек, образующих геометрическую прогрессию, а именно, результатом ДПФ сигнала $(x_j)$ является последовательность $\big(x(g^{-k})\big)$, где $g$ — соответствующий преобразованию корень из единицы. Таким образом, алгоритм 9.6.6 утверждает, что

вычисление значений многочлена на геометрической прогрессии, за исключением разве что небольшого штрафа за дополнение нулями, имеет ту же сложность, что и БПФ, при условии, что $g$ обратимо. Ясно, что любое БПФ можно вложить в свертку длины степени двойки, и потому для нашего алгоритма требуется не более трех БПФ новой, увеличенной длины. (Обратите внимание, что в некоторых случаях симметрия сигнала $y$ допускает дальнейшие улучшения.)

Третий, самый общий, случай вычисления значений многочленов, начинается с наблюдения, что можно использовать деление многочленов с остатком, для того чтобы провести понижающую процедуру. Пусть $x(t)$ имеет степень $D-1$, и этот многочлен требуется вычислить в точках $t_0, t_1, \ldots, t_{D-1}$. Допустим для простоты, что $d$ является степенью двойки. Если мы определим два многочлена, каждый степени в два раза меньшей, чем $x$, с помощью равенств

$$\begin{aligned} y_0(t) &= (t-t_0)(t-t_1)\ldots(t-t_{D/2-1}), \\ y_1(t) &= (t-t_{D/2})(t-t_{D/2+1})\ldots(t-t_{D-1}), \end{aligned}$$

то можно будет поделить на них с остатком:

$$x(t) = q_0(t)y_0(t) + r_0(t) = q_1(t)y_1(t) + r_1(t).$$

Однако это означает, что требуемое значение $x(t_j)$ равно либо $r_0(t_j)$ (если $j < D/2$) или $r_1(t_j)$ (если $j \geqslant D/2$). Таким образом, проблема вычисления значений многочлена $x$ степени $D-1$ сводится к двум более простым проблемам: вычислить многочлен степени примерно $D/2$ в примерно $D/2$ точках. Рекурсивный алгоритм работает следующим образом:

**Алгоритм 9.6.7** (вычисление значений многочлена на наборе произвольных точек). Пусть задан многочлен $x(t) = \sum_{j=0}^{D-1} x_j t^j$. Этот алгоритм с помощью рекурсивной функции *eval* возвращает все значения $x(t_j)$ для всех точек $t_0, \ldots, t_{D-1}$, на которые не накладывается дополнительных условий. Пусть $T$ обозначает набор $(t_0, \ldots, t_{D-1})$. Для удобства предполагаем, что $D = 2^k$, от этого условия, однако, можно отказаться (см. упражнение 9.76).

1. [Установить границу рекурсии]
   $\delta = 4$;
   // Выбрать границу, при которой обычный метод становится лучше.
2. [Рекурсивная функция *eval*]
   $eval(x, T)$ {
   $d = len(x)$;
3. [Проверить границу рекурсии]
   if($len(T) \leqslant \delta$) return $(x(t_0), x(t_1), \ldots, x(t_{d-1}))$;
   // В таком случае вернуть просто значения $x$ в этих точках.
4. [Разделить сигнал на две части]
   $u = L(T)$; // Младшая часть сигнала.
   $v = H(T)$; // Старшая часть сигнала.
5. [Собрать половинные многочлены]
   $w(t) = \prod_{m=0}^{d/2-1}(t-u_m)$;
   $z(t) = \prod_{m=0}^{d/2-1}(t-v_m)$;

6. [Поделить $x(t)$ с остатком]
$a(t) = x(t) \bmod w(t)$;
$b(t) = x(t) \bmod z(t)$;
**return** $eval(a, u) \cup eval(b, v)$;
}

Обратите внимание, что при вычислении $w(t), z(t)$ подразумевается, что скобки будут раскрыты, так что $w$, $z$ — сигналы, состоящие из коэффициентов многочленов. Операции по раскрыванию скобок следует учитывать при установлении сложности алгоритма (см. упражнение 9.75). В том же духе, заметим, что особенно эффективный способ реализовать алгоритм 9.6.7 заключается в том, чтобы заранее построить дерево полиномиальных остатков; т. е. использовать тот факт, что многочлены в шаге [Собрать половинные многочлены] были вычислены из своих половинок, и так далее.

Чтобы дать опору читателям, желающим опробовать этот общий алгоритм 9.6.7, приведем пример его работы. Рассмотрим задачу вычисления 64! не методом стандартного, последовательного умножения, а с помощью вычисления многочлена

$$\begin{aligned} x(t) &= t(1+t)(2+t)(3+t)(4+t)(5+t)(6+t)(7+t) \\ &= 5040t + 13068t^2 + 13132t^3 + 6769t^4 + 1960t^5 322t^6 + 28t^7 + t^8 \end{aligned}$$

в восьми точках

$$T = (1, 9, 17, 25, 33, 41, 49, 57)$$

и последующего умножения полученных результатов. Поскольку алгоритм полностью рекурсивный, проследовать за его работой не так уж и просто. Однако если положить $\delta = 2$ в шаге [Установить границу рекурсии] и распечатывать половинные многочлены $w$, $z$ и результаты деления с остатком $a, b$ сразу после того, как они установлены, то мы увидим следующее. На первом вызове функции *eval* мы получим

$$\begin{aligned} w(t) &= 3825 - 4628t + 854t^2 - 52t^3, \\ z(t) &= 3778929 - 350100t + 11990t^2 - 180t^3 + t^4, \\ a(t) &= x(t) \bmod w(t) \\ &= -14821569000 + 17447650500t - 2735641440t^2 + 109600260t^3, \\ b(t) &= x(t) \bmod z(t) \\ &= -791762564494440 + 63916714435140t - 1735304951520t^2 \\ &\qquad + 16010208900t^3, \end{aligned}$$

а для вычисления многочленов $a, b$ будут выполнены рекурсивные вызовы *eval*. Если мы продолжим следить за ходом выполнения алгоритма, то эти вызовы

покажут:

$$\begin{aligned} w(t) &= 9 - 10t + t^2, \\ z(t) &= 425 - 42t + t^2, \\ a(t) &= -64819440 + 64859760t, \\ b(t) &= -808538598000 + 49305458160t, \end{aligned}$$

и далее

$$\begin{aligned} w(t) &= 1353 - 74t + t^2, \\ z(t) &= 2793 - 106t + t^2, \\ a(t) &= -46869100573680 + 1514239317360t, \\ b(t) &= -685006261415280 + 15148583316720t. \end{aligned}$$

Больше уровней рекурсии не имеется (так как мы выбрали для нашего примера $\delta = 2$), поскольку функция *eval* достигнет конца рекурсии и выполнит вызов какого-либо классического метода, такого как схема Горнера, вычисляющего значения многочленов $a(t), b(t)$ в каждой из четырех точек $t_i$ напрямую. Окончательный результат этого вычисления — набор из восьми чисел

$$\begin{aligned} (x(t_0), \ldots, x(t_7)) = (&40320, 518918400, 29654190720, 424097856000, \\ &3100796899200, 15214711438080, 57274321104000, 178462987637760). \end{aligned}$$

В самом деле, произведение этих чисел равняется 64!, как и ожидалось. Следует отметить, что при такой операции умножения — когда требуется перемножить все числа вместе — на последнем шаге функции *eval* не требуется соединять половинки сигналов, а можно вместо этого вернуть произведение $eval(a, u) * eval(b, v)$. Если выбран будет такой подход, то на шаге [Проверить границу рекурсии] также должно возвращаться произведение указанных $x(t_i)$.

Кстати, коэффициенты многочленов не обязаны расти так же быстро, как показывает предыдущий пример. Во-первых, когда речь касается задач такого умножения большого количества чисел, обычно следует приводить все коэффициенты по модулю некоторого числа $N$ на каждом шаге алгоритма. Также имеется простой способ вычисления многочлена $x(t)$ степени $D$ в некотором меньшем числе точек $t_0, \ldots, t_{n-1}$ при $n < D$. Можно просто определить новый многочлен $s$ как остаток от деления

$$s(t) = x(t) \bmod \left( \prod_{j=0}^{n-1} (t - t_j) \right),$$

тогда достаточно будет вычисления значений многочлена $s$ (степени, примерно равной $n$) в $n$ заданных точках.

## 9.7. Упражнения

**9.1.** Показать, что представление числа в системе счисления с основанием $B$ и сбалансированное представление в системе счисления с основанием $B$ определяются однозначно. Это означает, что для любого неотрицательного целого

числа $x$ существует единственный набор цифр, соответствующий каждому из определний.

**9.2.** Хотя эта глава начинается сразу с умножения, стоит хотя бы раз взглянуть на простое сложение и вычитание, особенно для арифметики целых чисел со знаком.

(1) Пусть задано представление двух неотрицательных целых чисел $x, y$ в системе счисления с основанием $B$. Привести явный алгоритм для вычисления суммы $x + y$, складывающий цифру за цифрой, так чтобы результат снова был представлением в системе счисления с основанием $B$.

(2) Описать арифметику целых чисел со знаком, используя тот факт, что для реализации сложений и вычитаний чисел любых знаков достаточно алгоритма сложения (1) и еще одного алгоритма, а именно, вычитания чисел $x - y$ при условии, что $x \geqslant y \geqslant 0$. (Это означает, что любая задача сложения/вычитания сводится к одной из этих форм, и помимо этого для результата определяется его знак.)

(3) Выписать полные алгоритмы для сложения и вычитания целых чисел в системе счисления с основанием $B$ для чисел с произвольными знаками.

**9.3.** Пусть заданы сбалансированные представления в системе счисления с основанием $B$ двух неотрицательных целых чисел $x, y$. Представить явный алгоритм для вычисления суммы $x + y$, работающий цифра за цифрой, но всегда остающийся в рамках сбалансированного представления по основанию $B$ для суммы. Потом выписать такой же последовательный алгоритм умножения для сбалансированного представления чисел.

**9.4.** Известно, что умножение может быть сведено к одним сложениям, как в примере $3 \cdot 5 = 5 + 5 + 5$. Это простое соображение имеет практический смысл в некоторых ситуациях (а именно, для некоторых машин, особенно старых, где умножение двух слов является дорогостоящей операцией), как видно в последующих заданиях, где мы рассматриваем, как можно использовать приемы работы с памятью, чтобы уменьшить объем вычислений в ходе умножения длинных чисел. Рассмотрим умножение целых чисел, чья запись в системе счисления с основанием $B = 2^b$ имеет $D$ знаков. Эти числа имеют размер $2^n$, т. е. $n \approx bD$. На время следующих заданий назовем операциями с «машинными словами» операции, требующие двух операндов размера $B$ (т. е. имеющих $b$ битов).

(1) Сначала докажите, что стандартный школьный метод умножения, где посредством умножений машинных слов строится параллелограмм, а затем с помощью сложений машинных слов суммируются значения в колонках, требует $O(D^2)$ умножений и $O(D^2)$ сложений машинных слов.

(2) Заметив, что в параллелограмме может быть не более $B$ различных строк, докажите, что все возможные строки могут быть подсчитаны заранее таким образом, что полная операция умножения требует $O(BD)$ умножений и $O(D^2)$ сложений машинных слов.

(3) Теперь докажите, что предварительное вычисление всех строк параллелограмма может быть сделано с помощью последовательных сложений

и без единого умножения, так что умножение может быть сделано за $O(D^2 + BD)$ сложений машинных слов.

(4) Докажите, что подход школьного метода из задания (1) может быть реализован с использованием $O(n)$ битов вспомогательной памяти. Что можно сказать о требованиях к памяти методов (2) и (3)?

Если вы хотите создать проверочную программу, то вот возможное задание: выразить большие числа в системе счисления с основанием $B = 256 = 2^8$ и реализовать с помощью машины задание (2), используя таблицу из 256 предварительно подсчитанных целых чисел, чтобы заполнить ряды параллелограмма. Такая схема может работать медленнее, чем любой другой метод для больших чисел, однако, как мы упоминали, машина с очень медленным умножением машинных слов может выиграть от такого подхода.

**9.5.** Выписать явный алгоритм (или явную программу), который использует соотношение (9.3) для величин $w_n$, чтобы выполнить возведение целого числа большой разрядности в квадрат за время, примерно вдвое меньшее времени умножения двух таких чисел. Обратите внимание, что не требуется явно вычитать член $\delta_n$, если вы предпочтете модифицировать сумму по $i$. Основой для этого метода является то, что ромб из чисел школьного метода можно урезать наполовину. Это упражнение не так тривиально, как может показаться — следует учесть также соображения размера переноса, который может накапливаться в результате сложения огромного количества чисел в одной колонке.

**9.6.** Используя равенство (9.4), написать программу, вычисляющую любое произведение $xy$ для $x, y$, имеющих не более 15 двоичных битов, использующую только поиск в таблицах, сложения, вычитания и сдвиги, и требующую не более $2^{21}$ битов табличной памяти. (Подсказка: это равенство может быть использовано после того, как будет подсчитана определенная таблица.)

**9.7.** Модифицировать алгоритм двоичного деления (9.1.3) таким образом, чтобы он также возвращал значение $x \bmod N$. Заметим, что можно было бы просто применить равенство (9.5), однако имеется способ, использующий только локальные переменные самого алгоритма и избегающий умножения на $N$.

**9.8.** Доказать, что предписание Арази (алгоритм 9.1.4, вычисляющий простое умножение по модулю) и в самом деле возвращает значение $(xy) \bmod N$.

**9.9.** Найти алгоритм, подобный алгоритму 9.1.3, для оснований систем счисления $B = 2^k$, $k > 1$. Может ли это быть сделано без единого явного умножения?

**9.10.** Доказать теорему 9.2.1. Затем доказать ее расширение: разность $y/R - (xR^{-1}) \bmod N$ равняется одному из чисел $\{0, N, 2N, \ldots, (1 + \lfloor x/(RN) \rfloor)N\}$.

**9.11.** Доказать теорему 9.2.4. Затем вывести следствие для возведения в степень, для которого равенство (9.8) было бы частным случаем возведения в куб.

**9.12.** При использовании правил Монтгомери требуется вычислить заранее число $N' = (-N^{-1}) \bmod R$. В случае, если $R = 2^s$, а $N$ нечетно, покажите, что итерационный цикл Ньютона (9.10) при $a$, равном $-N$, с начальным значением $-N \bmod 8$ и при согласованности итерации с модулем $R$ быстро сходится

к $N'$. В частности, покажите, как можно посчитать начальные шаги цикла по модулю меньших степеней двойки, так чтобы вся требуемая работа (в предположении, что используются простейшие методы умножения и возведения в квадрат) могла быть проведена с помощью примерно $4/3$ $s$-битовых умножений и примерно $1/3$ $s$-битовых возведений в квадрат. Поскольку часть каждого произведения уничтожается, когда берется остаток по модулю, покажите, как можно еще сильнее уменьшить выполняемую работу. Сравните этот метод с традиционным методом вычисления обратных величин.

**9.13.** Мы показали, что метод Ньютона, хотя и эффективен, требует хитроумного выбора начальных значений. Для задачи нахождения обратных величин к действительным числам в соответствии с равенством (9.10) опишите строго границы области начальных догадок для заданного положительного действительного числа $a$, для которой метод Ньютона и в самом деле заставит $x$ сходиться к $1/a$.

**9.14.** Мы видели, что с помощью метода Ньютона можно выполнять деление, используя одни умножения. Оказывается, что точно так же можно извлекать квадратный корень. Рассмотрим сдвоенный метод Ньютона

$x = y = 1;$
do {
  $x = x/2 + (1 + a)y/2;$
  $y = 2y - xy^2;$
  $y = 2y - xy^2;$
}

где команда «do» обозначает, что операции, заключенные в фигурные скобки, повторяются какое-то подходящее число раз. Обратите внимание на то, что удвоение шага по $y$ сделано намеренно! Покажите, что эта схема формально порождает разложение в ряд для $\sqrt{1+a}$ по переменной $x$. Сколько верных членов последовательности получается после $k$ итераций цикла do?

Далее, посчитайте таким способом несколько квадратных корней, обратив внимание на то ограничение, что $|a|$ не может быть слишком большим, иначе метод разойдется (формальная правильность результирующего разложения по степеням $a$, разумеется, не означает автоматической сходимости метода).

Зададимся вопросом: можно ли использовать эти идеи, для того чтобы создать алгоритм, извлекающий целочисленные квадратные корни? Это могло бы быть заменой для алгоритма 9.2.11; последний, как мы помним, не содержал ни одного явного деления. По этому вопросу полезным является рассмотрение, для заданного числа $n$, из которого нужно извлечь корень, выражение вроде $\sqrt{n/4^q} = 2^{-q}\sqrt{n}$, позволяющее контролировать сходимость метода.

Кстати, интересно, что обычный, вещественный, метод Ньютона для вычисления обратной величины от квадратного корня сам по себе не содержит ни одного деления, и тем не менее для положительных степеней, похоже, что мы вынуждены использовать варианты сдвоенного алгоритма Ньютона, как это было сделано выше.

**9.15.** Числа Каллена — это числа вида $C_n = n2^n + 1$. Напишите программу возведения в степень Монтгомери, специально приспособленную для поиска составных чисел Каллена с помощью соотношений, таких как $2^{C_n-1} \not\equiv 1 \pmod{C_n}$. Например, в границах алгоритма возведения в степень по модулю $N = C_{245}$ можно взять $R = 2^{253}$, так чтобы было $R > N$. Вы можете заметить, например, что $C_{141}$ — псевдопростое число по основанию 2 (оно на самом деле простое). Примером намного большего простого числа Каллена является число Келлера $C_{18496}$. Также по поводу чисел Каллена смотрите упражнение 1.83.

**9.16.** Пусть мы хотим вычислить $1/3$ с помощью метода обращения Ньютона (в области вещественных чисел, так что результат будет $0.3333\ldots$). Для начальной догадки $x_0 = 1/2$ доказать, что для положительного $n$ $n$-е приближение $x_n$ на самом деле равно

$$x_n = \frac{2^{2^n} - 1}{3 \cdot 2^{2^n}},$$

тем самым показав свойство квадратичной сходимости успешного цикла Ньютона. Тот факт, что для приближений $x_n$ может быть дано явное выражение, интересен сам по себе. Такие выражения редки — можете ли Вы найти другие?

**9.17.** Выразить асимптотическую сложность алгоритма 9.2.8 через сложность умножения чисел размера $N$, в предположении, что выполнены все улучшения сдвигов, упомянутые в тексте. Затем определить асимптотическую сложность составной операции $(xy) \bmod N$ для $0 \leqslant x, y < N$ в случае, если обобщенная обратная величина еще неизвестна. Чему равна сложность $(xy) \bmod N$, если обратная величина известна? (Она должна быть асимптотически той же самой, что и у составной операции Монтгомери $(xy) \bmod N$, если в ней не обращать внимания на предварительные вычисления.) Кстати, в реальных программах, использующих идеи Ньютона—Барретта, внутри процедуры получения остатка от деления можно вставить проверку того, известна ли обратная величина, и если нет, вызывать процедуру вычисления обобщенной обратной величины.

**9.18.** Определить асимптотическую сложность алгоритма 9.2.13 при заданных $x$, $N$, выраженную через количество операций умножения на целые числа $c$ разных размеров. Например, при использовании какого-либо варианта школьного метода умножения битовая сложность операции умножения $yc$ была бы $O(\ln y \ln c)$. Ответьте также на любопытный вопрос: при каком размере $|c|$ (по сравнению с $N = 2^q + c$) взятие остатка для модулей особого вида настолько же плохо, как остальные обычные схемы (такие как деление длинных чисел или метод Ньютона—Барретта)? Кстати, наиболее подходящим для использования этого метода является случай, когда $c$ имеет размер одного машинного слова.

**9.19.** Упростите алгоритм 9.4.2 в случае, если не требуется вычислить расширенное решение $ax + by = g$, а нужна только сама обратная величина. (В этом случае не все операции алгоритма действительно необходимы.)

**9.20.** Реализуйте рекурсивный алгоритм 9.4.6 для НОД (или более новый алгоритм 9.4.7; смотрите следующий абзац). Выберите параметры $lim$ и $prec$, чтобы получить наибольшее быстродействие при вычислении $rgcd(x, y)$ для $x, y$ различных (но примерно равных) размеров. Вы должны увидеть, что $rgcd()$ становится лучше $cgcd()$ в случае, когда аргументы имеют, грубо говоря, тысячи битов в длину. (Заметьте: наша запись алгоритма 9.4.6 сделана в таком виде, что если в языке программирования выполняются обычные правила для глобальных переменных, таких как матрица $G$, и переменных, локальных для процедур, таких как переменные $x, y$ в $hgcd()$, то перевод из нашей записи в работающую программу будет не очень сложным.)

Что касается алгоритма 9.4.7, читатель должен обнаружить, что возникают различные проблемы оптимизации. Например, мы обнаружили, что алгоритм 9.4.6 будет, как правило, выполняться быстрее, если нет быстрого способа обнаружить нули в конце числа и сдвигать гигантские числа. С другой стороны, если такие приемы реализованы эффективно, новый алгоритм 9.4.7 будет господствовать.

**9.21.** Докажите, что алгоритм 9.2.10 работает правильно. Далее, найдите версию алгоритма, которая использовала бы идею разделения сдвигов, содержащуюся в равенстве (9.12) и последующих комментариях. Хорошим источником для построения циклов в этом случае является [Menezes et al. 1997].

Также исследуйте гипотезу [Oki 2003], что можно присваивать $s = 2B(N-1)$ в алгоритме 9.2.10.

**9.22.** Докажите, что алгоритм 9.2.11 работает правильно. Полезно будет наблюдение, что $x$ в ходе цикла строго уменьшается. Докажите затем оценку $O(\ln \ln N)$ для количества шагов, достаточного для завершения цикла. Используя идею об изменении точности на каждом шаге алгоритма, покажите, что битовая сложность правильно настроенного алгоритма может быть уменьшена до $O\left(\ln^2 N\right)$. Многие из этих идей восходят к подходу [Alt 1979].

**9.23.** Насколько произвольной может быть начальная установка $x$ в алгоритме 9.2.11?

**9.24.** Выпишите простой алгоритм, который использовал бы алгоритм 9.2.11, чтобы определить, является ли данное целое число полным квадратом. Заметим, что имеются намного более эффективные методы решения этой задачи, например, предварительная проверка того, что число является полным квадратом по модулю нескольких малых простых чисел [Cohen 2000].

**9.25.** Реализуйте алгоритм 9.2.13 внутри теста простоты Люка—Лемера. С помощью этого докажите или опровергните простоту различных чисел Мерсенна $2^q - 1$. Обратите внимание, что если имеется понижение по модулю особого вида, то в методе Люка—Лемера даже не требуются умножения в общем случае; достаточно одних возведений в квадрат.

**9.26.** Докажите, что алгоритм 9.2.13 работает правильно, т. е. завершается с нужным результатом.

**9.27.** Найти алгоритм для быстрой операции взятия остатка для модулей вида

$$p = 2^a + 2^b + \ldots + 1,$$

где все входящие в выражение показатели степени (позиции двоичных битов) $a, b, \ldots$ расположены редко, т. е. лишь малая часть битов двоичного представления $p$ равна 1. Найти также обобщение этого метода, для которого допускаются знаки «минус», т. е. $p = 2^a \pm 2^b \pm \ldots \pm 1$, где показатели степени тоже редки. В этой связи могут оказаться полезными работы [Solinas 1999] и [Johnson et al. 2001].

**9.28.** Некоторые вычисления, такие как числовое преобразование Ферма (ЧПФ) и прочие теоретико-числовые преобразования требуют умножения на степени двойки. На основании теоремы 9.2.12, найдите алгоритм, который для модуля $N = 2^m + 1$ быстро вычисляет $(x2^r) \bmod N$ при $x \in [0, N-1]$ для любого заданного (положительного или отрицательного) целого числа $r$. Требуется, по сути, алгоритм, который бы быстро выполнял переносы, которые используются в теореме, приводя все биты желаемого остатка в стандартный неотрицательный вид (за исключением, разумеется, случая, когда требуется использовать сбалансированное распределение или любую другую парадигму, разрешающую отрицательные цифры).

**9.29.** Найдите символическое соотношение для возведения в степень типа равенства (9.16), но для алгоритма 9.3.1.

**9.30.** Докажите, что алгоритм 7.2.4 работает правильно. Полезно проследить за работой маленьких примеров, таких как $n = 0011_2$, для которого $m = 1001_2$ (и мы специально дополнили $n$ до длины в 4 бита). Сравните сложность этого простейшего преобразования, подходящего для вычислений на эллиптических кривых, со схемой «слева направо», алгоритм 9.3.1, чтобы выяснить, имеется ли какая-либо выгода от парадигмы «сложения-вычитания».

**9.31.** Для алгоритмов двоичного НОД и расширенного двоичного НОД покажите, каким образом можно улучшить скорость работы, убрав часть операций в случае, если, например, $y$ простое и требуется вычислить $x^{-1} \bmod y$. Ключом к решению является замечание, что после шага [Начальная установка] каждого из алгоритмов знание того, что $y$ нечетное, позволяет убрать часть внутренних переменных. Таким образом, требуется получить алгоритм для обращения $x$ по заданному модулю $y$, использующий лишь четыре внутренних переменных.

**9.32.** Можно ли обобщить алгоритм 9.4.4 на случай составного $p$?

**9.33.** Доказать, что алгоритмы 9.4.4 и 9.4.5 работают правильно. Для последнего из них может быть полезным рассмотреть, каким образом обращается чистая степень двойки по модулю простого числа Мерсенна.

**9.34.** В том же духе, что и алгоритм 9.2.13 для модулей особого вида, сильно зависящий от битовых сдвигов, преобразовать алгоритм 9.4.5, с тем чтобы явно указать битовые сдвиги, используемые на различных шагах. В частности, не только операция взятия модуля, но и умножение на степень двойки чрезвычайно просто по модулю простых чисел Мерсенна; используя эти упрощения, перепишите алгоритм.

**9.35.** Можно ли вычислить НОД двух чисел размера $N$ за полиномиальное время (т. е. время, пропорциональное некоторой степени $\lg^{\alpha} N$), используя полиномиальное количество (т. е. $\lg^{\beta} N$) соединенных параллельно процессоров? По этому поводу можно посмотреть [Cesari 1998], где объясняется, что сейчас неизвестно, возможна ли такая вычислительная схема.

**9.36.** Выпишите явный алгоритм умножения целых чисел, используя метод Карацубы. Убедитесь, что Вы показали рекурсивную природу алгоритма, а также что правильно решили проблему переносов, которая возникает, когда результирующие цифры становятся больше основания системы счисления.

**9.37.** Покажите, что рекурсия в духе метода Карацубы, примененная к методу Тоома—Кука при $D = 3$ (т. е. рекурсия, примененная к алгоритму 9.5.2) дает целочисленное умножение двух чисел размера $N$ со сложностью, заявленной в тексте, т. е. за $O((\ln N)^{\ln 5/\ln 3})$ умножений машинных слов. (Здесь всюду подразумевается, что мы не считаем ни сложений, ни умножений на константы, так как они появлялись бы в каждом рекурсивном проходе шага [Восстановление] алгоритма 9.5.2.)

**9.38.** Преобразовать шаг [Начальная установка] алгоритма 9.5.2, так чтобы $r_i, s_i$ можно было вычислить наиболее эффективно.

**9.39.** Мы видели, что ациклическую свертку длины $N$ можно вычислить за $2N-1$ умножений (не считая умножений на константы; например, член $4x$ может быть вычислен одними сдвигами, без единого умножения). Оказывается, что циклическую свертку можно вычислить за $2N - d(N)$ умножений, где $d$ — стандартная функция делимости (равная количеству делителей числа $n$), в то время как негациклическая свертка может быть вычислена за $2N-1$ умножений. (Все эти результаты получены благодаря в основном Винограду; см. более старую, но все равно превосходную подборку [McClellan and Rader 1979].) Вот несколько явных задач, связанных с этим:

(1) Показать, что два комплексных числа $a + bi, c + di$ можно умножить за три вещественных умножения.

(2) Найти алгоритм, который выполняет негациклическую свертку длины 4 за девять умножений, где все умножения и деления делаются на степень двойки (и потому являются простыми сдвигами). Теоретический минимум, конечно, составляет 7 умножений, но такая версия с 9 умножениями имеет свои преимущества.

(3) Используя идеи метода Тоома—Кука, построить явную схему, вычисляющую свертку длины 4, работающую за 7 умножений.

(4) Можно ли использовать негациклическую свертку длины $D > 2$, чтобы построить метод умножения, подобный методу Карацубы, который был бы асимптотически лучше $O\left((\ln D)^{\ln 3/\ln 2}\right)$?

(5) Показать, как можно использовать преобразования Уолша—Адамара, чтобы вычислить циклическую свертку длины 16 за 43 умножения [Crandall 1996a]. Хотя теоретический минимум для количества умножений для свертки такой длины равен 27, схема Уолша—Адамара не содержит никаких сложно вычислимых коэффициентов. Похоже также, что

схема имеет сложность посередине между сложностью метода Винограда (линейной по $N$) и сложностью методов с использованием преобразований Фурье ($N \ln N$). В самом деле, 43 не так велико, как $16 \lg 16$. Настоящая сложность метода Уолша—Адамара до сих пор неизвестна.

**9.40.** Докажите теорему 9.5.13 с помощью идей свертки так, как это делается ниже. Пусть $N = 2 \cdot 3 \cdot 5 \ldots p_m$ — произведение последовательных простых чисел, определим

$$r_N(n) = \#\{(a,b) \ : \ a+b=n;\ \text{НОД}(a,N) = \text{НОД}(b,N) = 1;\ a,b \in [1, N-1]\},$$

т. е. $r_N(n)$ — количество представлений, которое требуется оценить снизу. Определим сигнал $y$ длины $N$ условиями $y_n = 1$, если НОД $(n, N) = 1$, и $y_n = 0$ иначе. Определим циклическую свертку

$$R_N(n) = (y \times y)_n.$$

Докажите, что для $n \in [0, N-1]$

$$R_N(n) = r_N(n) + r_N(N+n).$$

Другими словами, циклическая свертка является комбинированным представлением чисел $n$ и $N+n$. Далее, заметим, что сумма Рамануджана $Y$, определяемая равенством (9.26), является ДПФ от сигнала $y$, так что верно

$$R_N(n) = \frac{1}{N} \sum_{k=0}^{N-1} Y_k^2 e^{2\pi i k n/N}.$$

Докажите теперь, что функция $R$ мультипликативна, т. е. если $N = N_1 N_2$, где $N_1, N_2$ взаимно просты, то $R_N(n) = R_{N_1}(n) R_{N_2}(n)$. Из этого сделаем вывод, что

$$R_N(n) = \varphi_2(N, n),$$

где $\varphi_2$ определяется в тексте, следующем за теоремой 9.5.13. Итак, имеется явное выражение для $r_N(n) + r_N(N+n)$. Обратите внимание, что $\varphi_2$ положительно, если $n$ четно. Далее, докажите, что если $a+b=n$ (т. е. $n$ представимо), то $2N-n$ также представимо. Выведите следствие, что $r_N(n) > 0$ для всех четных $n \in [N/2+1, N-1]$, таким образом, все достаточно большие четные числа также представимы. Это означает, что остается показать, что для четных $n$ из отрезка $[N/2+1, N-1]$ величина $r_N(n+N)$ сравнительно мала по отношению к $\varphi_2(N,n)$. Наконец, заметим, что из равенства $a+b = N+n$ следует, что $b > n$, и рассмотрим величину

$$\#\{b \in [n, N] : b \not\equiv 0, n \pmod N\}.$$

Оценив ее, сделайте вывод, что для некоторой абсолютной константы $C$ и четного $n \in [N/2+1, N-1]$

$$r_N(n) \geqslant C \frac{n}{(\ln \ln N)^2} - 2^{m+1}.$$

Это доказывает теорему 9.5.13 для достаточно больших произведений $N$, а для малых произведений, таких как $N = 2, 6, 30$ теорема может быть доказана конечным перебором $n < 2N$.

Заметим, что теорему можно доказать напрямую применением просеивания. Другой способ заключается в том, чтобы использовать китайскую теорему об остатках и некоторые комбинаторные соотношения, чтобы выразить $R_N$ как функцию от $\varphi_2$. Любопытен вопрос: можно ли полностью избежать приведенных доводов (для оценки $r_N(N+n)$), которые, по общему признанию, являются в некотором роде аргументами метода решета, используя вместо них алгебраические операции над негациклической сверткой $y \times_- y$? Как мы отмечали в основном тексте, это потребовало бы анализа некоторых любопытных тригонометрических сумм. Нам неизвестны никакие удобные явные выражения для негациклической свертки, но если бы они могли быть получены, то точное количество представлений $n = a+b$ было бы точно так же представимо в виде явной формулы.

**9.41.** Интересные точные результаты, касающиеся сумм квадратов, можно получить с помощью аккуратного применения принципов свертки. Основная идея заключается в том, чтобы рассмотреть сигнал с элементами $x_{n^2}$, равными 1, и нулевыми остальными элементами. Пусть $p$ — нечетное простое число, определим

$$\hat{x}_k = \sum_{j=0}^{(p-1)/2} \left(1 - \frac{\delta_{0j}}{2}\right) e^{-2\pi i j^2 k/p},$$

где $\delta_{ij} = 1$, если $i = j$, и 0 в противном случае. Покажите, что $\hat{x}_0 = p/2$, в то время как для $k \in [1, p-1]$ верно

$$\hat{x}_k = \frac{\omega_k}{2}\sqrt{p},$$

где $\omega_k = \left(\frac{k}{p}\right), -i\left(\frac{k}{p}\right)$ соответственно, если $p \equiv 1, 3 \pmod 4$. Идея заключается в том, чтобы показать, что все это является следствием теоремы 2.3.7. (Обратите внимание, что теория сумм Гаусса более общего характера связана с методами проверки простоты, например, в нашей лемме 4.4.1 и следующих за ней рассмотрениях.) Теперь для $n \in [0, p-1]$ определим $R_m(n)$ как количество представлений числа $n$ в виде суммы $m$ квадратов

$$a_1^2 + a_2^2 + \ldots + a_m^2 \equiv n \pmod p$$

для целых $a_j \in [0, (p-1)/2]$, за исключением того, что представлению сопоставляется весовой сомножитель $1/2$ за каждую нулевую компоненту $a_j$. Например, представлению $0^2 + 3^2 + 0^2$ сопоставляется весовой коэффициент $1/4$. Рассматривая соответствующую $m$-кратную свертку определенного сигнала самого с собой, покажите, что

$$\begin{aligned} R_2(n) &= \frac{1}{4}\left(p + \left(\frac{-1}{p}\right)(p\delta_{0n} - 1)\right), \\ R_3(n) &= \frac{1}{8}\left(p^2 + \left(\frac{-n}{p}\right)p\right), \\ R_4(n) &= \frac{1}{16}\left(p^3 + p^2\delta_{0n} - p\right). \end{aligned}$$

(Типичный вариант, который можно посчитать вручную, имеет место для $p = 23$: $R_4(0) = 12673/16$, и для любого $n \not\equiv 0 \pmod p$, $R_4(n) = 759$.)

Далее, из этих точных соотношений сделайте вывод:

(1) Любое простое число $p \equiv 1 \pmod 4$ является суммой двух квадратов, в то время как никакое из $p \equiv 3 \pmod 4$ не может так представляться (см. также упражнение 5.16).
(2) Существует число $0 < m < p$, такое что $mp = a^2 + b^2 + c^2 + d^2$.

Результат (2) быстро приводит к классической теореме Лагранжа, утверждающей, что каждое неотрицательное целое число является суммой четырех квадратов. Можно было бы использовать малую теорему Лагранжа, чтобы доказать, что наименьшее $m$, удовлетворяющее утверждению (2), равно 1, так что любое *простое* число является суммой четырех квадратов. Последним шагом, таким образом, становится доказательство того, что если два целых числа $a, b$ представимы в виде четырех квадратов, то их произведение $ab$ также представимо. Эти завершающие подробности можно найти в [Hardy and Wright 1979].

Что можно сказать о сумме трех квадратов? Интересной задачей было бы использование свертки, для того чтобы доказать довольно непростую знаменитую теорему Гаусса «$num = \Delta + \Delta + \Delta$», означающую, что любое неотрицательное целое число является суммой трех треугольных чисел, т. е. чисел вида $k(k+1)/2, k \geqslant 0$. Это равносильно утверждению, что любое целое число, сравнимое с 3 по модулю 8, является суммой трех квадратов. (На самом деле, все числа, не допускающие такого представления в виде суммы трех квадратов, имеют вид $4^a(8b + 7)$.) Не ясно, как браться за такую задачу; с другой стороны, из соотношения для $R_2$ выше следует, что любое $p \equiv 7 \pmod 8$ удовлетворяет достаточно странному равенству $mp = a^2 + b^2 + c^2$ для некоторого $m < p$. (Например, 7 не является суммой трех квадратов, но 14 является.)

**9.42.** Покажите, что циклическая свертка двух сигналов длины $D$ равносильна умножению двух многочленов, т. е.

$$x \times y \equiv x(t)y(t) \pmod{t^D - 1},$$

где «$\equiv$» означает, что элементы сигнала слева соответствуют коэффициентам многочлена справа. Затем покажите, что негациклическая свертка $x \times_- y$ равносильна умножению по модулю $\pmod{t^D + 1}$. Используя китайскую теорему об остатках, воспользуйтесь этими фактами для доказательства равенства (9.36), используемого в свертке Нюссбаумера.

**9.43.** В том же духе, что и в упражнении 9.42, дайте полиномиальное описание общей взвешенной свертки $x \times_a y$, где $a = \left(A^j\right)$ для некоторого порождающего элемента $A$.

**9.44.** Реализуйте алгоритм 9.5.19 в целях доказательства того, что $p = 2^{521} - 1$ — простое, с использованием теста простоты Люка—Лемера. Идея заключается в том, чтобы хранить особое представление по переменному основанию всех величин в течение всего теста простоты. (Другими словами, результат работы алгоритма 9.5.19 готов к использованию в качестве входа для последу-

ющего вызова алгоритма.) Для больших простых чисел, таких как новые найденные числа Мерсенна, исследователи использовали такие длины, что число $q/D$, т. е. типичное количество битов цифры в переменном основании, имело 16 битов или меньше. Кроме того, это использовалось для того, чтобы подавить возможные ошибки арифметики чисел с плавающей точкой.

**9.45.** Реализуйте алгоритм 9.5.17, чтобы установить природу различных чисел Ферма, используя тест простоты Пепена, утверждающий, что $F_n$ — простое тогда и только тогда, когда $3^{(F_n-1)/2} \equiv -1 \pmod{F_n}$. Как вариант, можно использовать тот же алгоритм для задач разложения на множители [Brent et al. 2000]. (Заметим, что метод уменьшения погрешности с помощью использования сбалансированного представления, упомянутый в упражнении 9.55, также применим к вычислениям с числами Ферма в этом алгоритме.) Этот метод также был использован для установления характера $F_{22}$ в 1993 г. [Crandall et al. 1995] и $F_{24}$ [Crandall et al. 1999].

**9.46.** Реализуйте алгоритм 9.5.20, для того чтобы выполнять умножение больших чисел с помощью циклической свертки сигналов, дополненных нулями. Можно ли использовать методы ВДП для вычисления негациклической целочисленной свертки с помощью китайской теоремы об остатках на подходящем наборе простых чисел?

**9.47.** Покажите, что если в арифметическом поле имеется кубический корень из единицы, то для $D = 3 \cdot 2^k$ можно выполнить циклическую свертку длины $D$ с помощью комбинирования трех отдельных сверток длины $2^k$. (См. упражнение 9.43; рассмотрите символическое разложение многочлена $t^D - 1$ для такого $D$.) Этот подход был в самом деле использован Вольтманом для обнаружения новых простых чисел Мерсенна (он использовал ВДПИО длины $3 \cdot 2^k$).

**9.48.** Реализуйте идеи [Percival 2003], обобщающие алгоритм 9.5.19 на случай вычислений по модулю чисел Прота $k \cdot 2^n \pm 1$. Основная идея заключается в том, что вычисления по модулю чисел $a \pm b$ можно выполнять с хорошим контролем над ошибками все время, пока произведение простых чисел $\prod_{p|ab} p$ остается достаточно малым. Согласно подходу Персиваля, выполняется обобщение представления с переменным основанием из теоремы 9.5.18, приводящее к виду, содержащему произведения

$$x = \sum_{j=0}^{D-1} x_j \prod_{p^k \| a} p^{\lceil kj/D \rceil} \prod_{q^m \| b} q^{\lceil -mj/D \rceil + mj/D},$$

обеспечивающее быстрые вычисления по модулю $a - b$.

Заметим, что соединение этого подхода с быстрой операцией mod алгоритма 9.2.14 дало бы в результате оптимальный метод для вычислений с числами, отличными от привычных чисел Мерсенна/Ферма. В самом деле, как показано в исследованиях обобщенных чисел Ферма [Dubner and Gallot 2002], наилучшие результаты уже достигнуты.

**9.49.** В литературе, посвященной БПФ, существует особо эффективное преобразование вещественных сигналов, называемое БПФ Сёренсона. Это преоб-

разование с разделенным основанием, использующее $\sqrt{2}$ и схему понижения, чтобы получить БПФ с наименьшей известной сложностью для вещественных сигналов; однако в настоящее время вопросы, связанные с использованием памяти, кэша и способностей процессоров настолько важнее, что просто количество операций стало играть куда меньшую роль. Итак, для кольца $\mathbf{Z}_n$, где $n = 2^m + 1$, а $m$ делится на 4, покажите, что корень из двух равен

$$\sqrt{2} = 2^{3m/4} - 2^{m/4}.$$

Определите, можно ли выполнить преобразование Сёренсона по модулю $n$ простым использованием стандартной процедуры Сёренсона, где вместо $\sqrt{2}$ используется подстановка предыдущей формулы. (Подробный текст для преобразования БПФ Сёренсона вещественных сигналов можно найти в [Crandall 1994b].)

**9.50.** Рассмотрим преобразование, имеющее вид обычного ДПФ

$$X_k = \sum_{j=0}^{N-1} x_j h^{-jk},$$

за исключением того, что элементы сигнала $x_j$ и корень $h$ степени $N$ берутся из поля $\mathbf{Q}\left(\sqrt{5}\right)$. Это преобразование называется теоретико-числовым преобразованием (ТЧП) над «квадратичным полем золотого сечения», из-за того что золотое сечение $\phi = \left(\sqrt{5} - 1\right)/2$ принадлежит этому полю. Далее предполагается, что все вычисления проводятся в поле $\mathbf{Z}[\phi]$, так что элементы сигнала и корень $h$ имеют вид $a + b\phi$, где $a, b$ — целые. Докажите, что умножение в этом поле можно выполнить за три целочисленных умножения. Затем рассмотрите поле $\mathbf{F}_p\left(\sqrt{5}\right)$ и разработайте теорию для возможной длины таких преобразований над этим полем, если в качестве корня выбирается степень числа $\phi$. Рассмотрите преобразование (при четном $N$)

$$\mathbf{X}_k = \sum_{j=0}^{N/2-1} \mathbf{H}^{-jk} \mathbf{x}_j,$$

где новый сигнал содержит векторные элементы $\mathbf{x}_j = (a_j, b_j)$, а компоненты исходного сигнала представлялись в виде $x_j = a_j + b_j\phi$ в этом поле. Матрица $\mathbf{H}$ здесь имеет вид:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

Опишите, в каком смысле это матричное преобразование равносильно предшествующему определению ДПФ, при условии что степени матрицы $\mathbf{H}$ удобно описываются с помощью чисел Фибоначчи

$$\mathbf{H}^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix},$$

и $n$-я степень матрицы может быть подсчитана с помощью метода «разделяй и властвуй» за $O(\ln n)$ матричных умножений. В заключение, установите сложность этого теоретико-числового преобразования с использованием матриц.

Этот пример необычного преобразования, напоминающего о дискретном преобразовании Галуа (ДПГ) из основного текста главы, появился в работах [Dimitrov et al. 1995], [Dimitrov et al. 1998] и был на самом деле предложен как способ получения содержательных спектров — в дискретном пространстве (!) — для вещественных данных из реального мира.

**9.51.** Следуя алгоритму 9.5.22 для циклической свертки, предложите похожий алгоритм для негациклической свертки с помощью скомбинированного метода ДПГ/ВДП с половинной длиной, что означает, что вам требуется свернуть два вещественных целочисленных сигнала длины $D$ с помощью комплексного преобразования ДПГ длины $D/2$. Вам потребуется найти для взвешенного преобразования ВДП длины $D/2$ корень степени $D/2$ из $i$ в поле $\mathbf{F}_{p^2}$, где $p$ — простое число Мерсенна. Подробности, которые могут помочь в такой реализации, можно найти в [Crandall 1997b].

**9.52.** Изучим так называемое числовое преобразование Ферма (ЧПФ), определяемое равенством

$$X_k = \sum_{j=0}^{D-1} x_j g^{-jk} \pmod{f_n},$$

где $f_n = 2^n + 1$, а $g$ имеет мультипликативную степень $D$ в кольце $\mathbf{Z}_n$. Полезно выбрать в качестве $g$ степень двойки; какие длины сигналов $D$ допустимы в таком случае? Преобразование ЧПФ имеет преимущество, что внутренние «бабочки» быстрой реализации метода используют вычисления, свободные от умножений, но и отчетливый недостаток, связанный с ограничением на допустимые длины сигналов. Имеются ли другие полезные приложения ЧПФ в вычислительной теории чисел, кроме того, что ЧПФ встречается в алгоритме Шёнхаге 9.5.23?

**9.53.** В таких случаях, как в алгоритме 9.5.7, хочется использовать оптимальные способы транспонирования матриц. Это несложно сделать, если матрица квадратная, но в остальных случаях проблема нетривиальна. Обратите внимание, что проблема снова оказывается простой для любой матрицы, если разрешить сначала скопировать исходную матрицу, и затем записать ее обратно в транспонированном порядке. Однако это может потребовать длинных скачков в памяти, которые на самом деле не нужны, так же как и вспомогательная память для копии.

Итак, предложите алгоритм для транспонирования матрицы в общем случае, используя только ту память, где находится исходная матрица; это означает, что создание копии недопустимо, и требуется использовать по возможности минимальные скачки в памяти. Ссылки по этой теме: [Van Loan 1992], [Bailey 1990].

**9.54.** Анализируя соответствующие сложности шагов алгоритма 9.5.8, (1) покажите, что утверждение о сложности в тексте верно для вычисления величин $X'_k$; (2) дайте более точную информацию о константе, скрытой в $O$ большом в оценке 9.24; (3) докажите неравенство (9.25) и (4) объясните, как это неравенство приводит к утверждаемой оценке сложности алгоритма.

Заинтересованный читатель может исследовать/улучшить содержательную часть исходного метода Датта—Рохлина, заключающуюся в том, чтобы разложить колебание $e^{\imath cz}$ в гауссову последовательность [Dutt and Rokhlin 1993].

**9.55.** Перепишите алгоритм 9.5.12 так, чтобы он использовал сбалансированное представление цифр (определение 9.1.2). Обратите внимание, что важные изменения сосредоточены на шаге выполнения переносов. Изучите явление, упомянутое в тексте перед алгоритмом, а именно, что погрешность при использовании сбалансированного представления уменьшается. Имеются несколько численных исследований этого явления, вместе с несколькими теоретическими гипотезами (см. [Crandall and Fagin 1994], [Crandall et al. 1999] и содержащиеся в них ссылки), однако об оценках погрешности, одновременно строгих и при этом полезных, известно очень мало.

**9.56.** Покажите, что если $p = 2^q - 1$, где $q$ — простое, и $x \in \{0, \ldots, p-1\}$, то $x^2 \bmod p$ можно найти, используя два умножения чисел размера $(q/2)$. Подсказка: представьте $x$ в виде $x = a + b2^{(q+1)/2}$, и свяжите задачу возведения $x$ в квадрат с числами

$$(a+b)(a+2b) \text{ и } (a-b)(a-2b).$$

Эта интересная процедура не дает ничего действительно нового — потому что мы уже знаем, что возведение в квадрат (в диапазоне чисел, где оптимальным является школьный метод) примерно в два раза проще, чем умножение — однако это другой метод получения увеличения скорости вдвое, более того, он не требует детального вмешательства в структуру циклов, как это было необходимо для равенства (9.3).

**9.57.** Всегда ли существует набор простых чисел $p_1, \ldots, p_r$, требуемый в алгоритме 9.5.20, и если да, то как его найти?

**9.58.** Докажите, что как и утверждает алгоритм 9.5.20, любой элемент свертки $x \times y$ в этом алгоритме и в самом деле ограничен величиной $NM^2$. Можно ли в приложении к умножению целых чисел большой разрядности использовать идеи сбалансированного представления, т. е. считать, что число по модулю $p$ лежит в отрезке $[-(p+1)/2, (p-1)/2]$, чтобы понизить эту оценку, тем самым уменьшив набор требуемых простых чисел для китайской теоремы об остатках?

**9.59.** Для дискретного преобразования (9.33), основанного на простых числах, в случае, если $g$ имеет квадратный корень $h$, $h^2 = g$, ответьте на вопрос: какова явная формула для элемента преобразованного сигнала $X_k$, если входной сигнал задан равенством $x = \left(h^{j^2}\right)$, $j = 0, \ldots, p-1$? Обратив внимание на определенную простоту сигнала $X_k$, найдите аналогичный сигнал $x$, имеющий $N$ элементов в случае комплексных чисел, для которого обычное комплекснозначное БПФ обладает удобным свойством для абсолютных величин $|X_k|$. (Такие сигналы называют «чирикающими», они имеют большое значение в проверке процедур, вычисляющих БПФ, которые должны, конечно же, продемонстрировать проявление упомянутого соотношения для абсолютных величин элементов сигнала.)

**9.60.** Для простого числа Мерсенна $p = 2^{127} - 1$ укажите явно примитивный корень $a + bi$ 64-й степени из единицы в группе $\mathbf{F}_{p^2}^*$.

**9.61.** Покажите, что если $a+bi$ — примитивный корень максимального порядка $p^2-1$ в группе $\mathbf{F}_{p^2}^*$ (где $p \equiv 3 \pmod 4$, так что существует «$i$»), то число $a^2+b^2$ должно быть примитивным корнем максимального порядка $p-1$ в группе $\mathbf{F}_p^*$. Верно ли обратное?

Найдите несколько простых чисел Мерсенна $p = 2^q - 1$, для которых $6 + i$ является примитивным корнем в группе $\mathbf{F}_{p^2}^*$.

**9.62.** Докажите, что алгоритм 9.5.22 для целочисленной свертки с использованием дискретного преобразования Галуа (ДПГ) работает правильно.

**9.63.** Насколько большими могут быть $x, y$, если в алгоритме 9.5.22 для умножения целых чисел большой разрядности с предварительным дополнением нулями используется целочисленная свертка и ДПГ, в качестве простого числа Мерсенна выбрано $p = 2^{89} - 1$, а элементы сигналов $x, y$ понимаются как цифры по основанию системы счисления $B = 2^{16}$? Что будет, если используется сбалансированное представление цифр (и каждая цифра лежит в интервале $[-2^{15}, 2^{15} - 1]$)?

**9.64.** Опишите, как можно использовать алгоритм 9.5.22 с набором простых чисел Мерсенна, чтобы вычислить целочисленную свертку с помощью реконструкции по китайской теореме об остатках; опишите также и процедуру реконструкции. (Кстати, для случая простых чисел Мерсенна процедура реконструкции особо прямолинейна.)

**9.65.** Проанализируйте сложность алгоритма 9.5.22, учитывая рекурсию по типу алгоритма Шёнхаге 9.5.23, и объясните, как это сопоставляется со сложностями из таблицы 9.1.

**9.66.** Объясните, как можно использовать идеи ВДП для того, чтобы избежать необходимости дополнять сигналы нулями в алгоритме 9.5.25. А именно, покажите, как можно использовать не циклическую свертку длины $2m$, а одну циклическую и одну негациклическую свертку длины $m$. Это возможно, поскольку у нас есть примитивный корень степени $m$ из $-1$, так что можно использовать ВДП для вычисления негациклической свертки. Заметим, что это не изменяет существенно сложности вычислений, но на практике уменьшает объем используемой памяти.

**9.67.** Докажите оценку сложности в $O(D \ln D)$ операций, следующую за алгоритмом Нюссбаумера 9.5.25. Затем проанализируйте довольно запутанную задачу оценки битовой сложности алгоритма. Один из способов начать такой анализ сложности заключается в том, чтобы выбрать оптимальное основание системы счисления $B$, как указано в таблице сложностей в разделе 9.5.8.

**9.68.** Для нечетных простых $p$ алгоритм Нюссбаумера 9.5.25 позволяет вычислять циклические и негациклические свертки по модулю $p$, т. е. в кольце $R$, совпадающем с $\mathbf{F}_p$. Все, что требуется — выполнять все операции с элементами $R$ по модулю $p$, так что структура алгоритма не меняется. Используя такую реализацию алгоритма Нюссбаумера, докажите великую теорему Ферма для

некоторых больших показателей $p$, вызывая свертку, чтобы вычислить ДПФ Шокроллахи. Имеются различные способы преобразования ДПФ в свертки. Один из способов — использовать прием Блюштейна с переиндексацией, другой — считать ДПФ проблемой вычисления многочлена во многих точках, еще один — прием Радера (который работает в случае, если длина сигнала является степенью простого числа). Далее, свертки длин, которые не являются степенями двойки, можно заключить в более длинные, но и более удобные свертки (обсуждение того, как осуществляются переходы от преобразований к сверткам и обратно, смотрите в [Crandall 1996a]). Нужно будет применить теорему 9.5.14, заметив сначала, что длина ДПФ может быть уменьшена до $(p-1)/2$. Вычислите затем ДПФ с помощью циклической свертки длины степени двойки, используя метод Нюссбаумера по модулю $p$. Помимо недавнего захватывающего теоретического успеха Э. Уайлса, доказавшего великую теорему Ферма, на практике были проверены все показатели степени $p < 12000000$. [Buhler et al. 2000]. Кстати, наибольшим простым числом, регулярность которого была доказана с помощью критерия Шокроллахи, является $p = 671008859$ [Crandall 1996a].

**9.69.** Реализуйте алгоритм 9.6.1 для умножения многочленов, коэффициенты которых берутся по модулю $p$. Такая реализация полезна, например, в алгоритме Шуфа для работы с точками на эллиптических кривых, поскольку в этом методе требуется не только умножать большие многочлены, но и создавать схемы возведения в степень, опирающиеся на умножение многочленов очень большой степени.

**9.70.** Докажите оба утверждения о сложности, следующие в тексте главы после алгоритма 9.6.1. Опишите, в каких условиях, например, при каких значениях $D, p$, или при каких условиях на память, и т.д., какой из методов — свертки Нюссбаумера или двоичного разбиения — будет практически более полезным.

Для дальнейшего анализа можно рассмотреть метод Шупа умножения многочленов [Shoup 1995], который основан на использовании китайской теоремы об остатках и имеет свою собственную формулу сложности. На который из двух перечисленных методов метод Шупа более похож, если сравнивать их сложности?

**9.71.** Пусть коэффициенты многочленов $x(t), y(t)$ рассматриваются по модулю $p$, а степени этих многочленов $\approx N$. Чему равна асимптотическая битовая сложность операции взятия остатка от деления многочленов $x \bmod y$ для алгоритма 9.6.4, вызывающего алгоритм 9.6.2, в терминах $p$ и $N$? (Потребуется сделать предположение о сложности целочисленного умножения для коэффициентов многочленов.) Какова она будет, если, как и требуется во многих случаях вычисления целочисленной операции mod, выполнять много раз операцию взятия остатка от деления многочленов на один и тот же многочлен $y(t)$, так что требуется вычислить обратную величину $R[y, \ ]$ только один раз?

**9.72.** Рассмотрим еще одно соотношение для чисел Бернулли по модулю $p$. Докажите теорему, утверждающую, что если $p \geqslant 5$ — простое, $a$ — взаимно просто с $p$, и $d$ определено как $d = -p^{-1} \bmod a$, то для четных $m$ из отрезка

$[2, p-3]$ верно:

$$\frac{B_m}{m}(a^m - 1) \equiv \sum_{j=0}^{p-1} j^{m-1}(dj \bmod a) \pmod{p}.$$

Выведите следствие, что

$$\frac{B_m}{m}(2^{-m} - 1) \equiv \frac{1}{2} \sum_{j=1}^{(p-1)/2} j^{m-1} \pmod{p}.$$

Получите интересное следствие, что если $p \equiv 3 \pmod 4$, то $B_{(p+1)/2}$ не может равняться нулю по модулю $p$.

Такая формула для суммы имеет практическое значение, однако существует ее вычислительно более эффективная форма, где индексы суммирования должны пробегать лишь часть целых чисел из интервала $[0, p-1]$, см. [Wagstaff 1978], [Tanner and Wagstaff 1987].

**9.73.** Докажите, что алгоритм 9.6.5 работает правильно. Модифицируйте его для решения несколько другой задачи, а именно, для вычисления многочлена, заданного в виде

$$x(t) = t(t+d)(t+2d)\dots(t+(n-1)d),$$

в одной заданной точке $t_0$. Идея заключается в том, чтобы выбрать некоторое оптимальное $G < n$ и рассмотреть цикл

$$\mathsf{for}(0 \leqslant j < G) \quad a_j = \textstyle\prod_{q=0}^{G-1}(t_0 + (q + Gj)d);$$

Получите таким способом алгоритм, требующий $O(G^2 + n/G)$ умножений и $O(n+G^2)$ сложений для вычисления $x(t_0)$. Покажите, что с помощью рекурсии частичное произведение в цикле for() (где это частичное произведение снова имеет тот же тип, что и обрабатываемое всем алгоритмом произведение) можно найти $x(t_0)$ за $O(n^{\phi+\epsilon})$ умножений, где $\phi = \left(\sqrt{5}-1\right)/2$ — золотое сечение. Каково в этом случае количество сложений?

Используйте алгоритм такого типа, чтобы вычислить факториалы нескольких больших чисел, например, чтобы проверить простоту некоторого большого числа $p$, проверив, верно ли $(p-1)! \equiv -1 \pmod p$. Основная идея заключается в том, что вычисление произведения многочленов вида

$$(t+1)(t+2)\dots(t+m)$$

в точках $\{0, m, 2m, \dots, (m-1)m\}$ дает $\left(\left(m^2\right)!\right)$. При поиске простых чисел Вильсона использовали этот прием, проводя все вычисления по модулю $p^2$ [Crandall et al. 1997].

**9.74.** Пусть многочлен $x(t)$ дан в виде произведения

$$x(t) = \prod_{k=0}^{D-1}(t - t_k),$$

где все точки $t_k$ заданы. Рассмотрев накопление попарных произведений этих точек, покажите, что $x$ можно представить в обычном виде $x(t) = x_0 + x_1 t + \ldots + x_{D-1} t^{D-1}$ за $O\left(D \ln^2 D\right)$ операций в поле.

**9.75.** Докажите, что алгоритм 9.6.7 работает верно, и докажите оценку сложности (выраженную в терминах операций в кольце), если частичные многочлены и деление многочленов с остатком выполняются «школьным» методом. Какова оценка сложности, если частичные многочлены порождаются с помощью быстрого умножения (см. упражнение 9.74), но деление с остатком до сих пор классическое? Какая сложность возникает, если действует и быстрое умножение многочленов, и деление с остатком (как в алгоритме 9.6.4)?

Как продолжение этой задачи исследуйте некоторые поразительные результаты в области деревьев остатков (см. разд. 3.3) и результаты Бернштейна об увеличенных остаточных деревьях для многочленов [Bernstein 2004a]. Такие методы, вместе с соответственно преобразованным алгоритмом 9.6.7, могут привести к оценке, столь же хорошей, как $O\left(D \ln^{2+o(1)} D\right)$, с понижением скрытой в $O$ константы, возможным благодаря хранению главных операндов БПФ во время умножения больших чисел.

**9.76.** Изучите возможность ослабления того условия, что в алгоритме 9.6.7 $D$ должно быть степенью двойки. Одним из способов, конечно же, является допущение, что изначальные многочлены имеют запас из нулевых коэффициентов (и что набор точек для вычисления значений многочлена $T$ имеет длину, равную степени двойки), и потому можно считать, что степень многочлена $x$ на один меньше, чем степень двойки. Другой способ заключается в том, чтобы изменить шаг [Проверить границу рекурсии] так, чтобы он проверял, нечетна ли длина набора $T$. Проверка такого типа дает гарантию, что в ходе рекурсии длина сигнала уменьшится вдвое.

## 9.8. Проблемы для исследования

**9.77.** Как мы упомянули, способы улучшения схем возведения в степень могут быть сложными, и во многих отношениях не исследованными. В этом упражнении мы рассматриваем некоторые интересные проблемы, связанные с такими улучшениями.

Если задан алгоритм для поиска обратной величины (как вариант, если возможно обращение точек в алгебре точек на эллиптической кривой), то выбор метода возведения в степень с помощью сложения/вычитания делает ситуацию более интересной. Алгоритм 7.2.4 возведения в степень с помощью сложения/вычитания, например, имеет следующую интересную «стохастическую» интерпретацию. Пусть $x$ обозначает действительное число из интервала $(0, 1)$, а $y$ — дробную часть числа $3x$, т. е. $y = 3x - \lfloor 3x \rfloor$. Обозначим через $z$ результат применения «исключающего или» к $x, y$:

$$z = x \wedge y,$$

что означает, что $z$ является побитовым «исключающим или» двоичных пред-

ставлений $x$ и $y$. Рассмотрим теперь следующее предположение: если $x, y$ выбираются наугад, то с вероятностью 1 одна треть всех двоичных битов $z$ равна единице. Если это и в самом деле так, то это предложение означает, что если имеется операция возведения в квадрат, работающая за время $S$, и операция умножения, работающая за время $M$, то алгоритм 7.2.4 работает за время, примерно равное $(S+M/3)b$, если рассматриваемые операнды имеют по $b$ двоичных битов. Как это соотносится со стандартными двоичными схемами возведения в степень, см. алгоритмы 9.3.1 и 9.3.2? Как это соотносится со случаем основания системы счисления $B = 3$ обобщенного алгоритма просмотровой схемы (алгоритм 9.3.3)? (Для ответа на этот вопрос Вам потребуется установить, равносильна ли схема сложения/вычитания некоторой просмотровой схеме, или нет.)

Далее, установите точные оценки количества операций возведения в квадрат и сложения в реальных схемах возведения в степень. Например, для двоичной схемы возведения в степень типа «слева направо» более точная оценка сложности составляет

$$C \sim (b(y)-1)S + (o(y)-1)M,$$

где запись показателя $y$ имеет длину $b(y)$ битов, из которых $o(y)$ равны 1. Расширьте эту теорию на случай просмотровых схем, учитывая также шаг предварительных вычислений. Таким образом опишите количественно, какая из схем будет наилучшей для типичного приложения в криптографии, а именно, для случая, если $x, y$ имеют по 192 бита и требуется вычислить $x^y$ по модулю некоторого 192-битового простого числа.

Далее, реализуйте схему умножения на эллиптических кривых по основанию системы счисления $B = 16$, что означает, что как и в алгоритме 9.3.3, одновременно обрабатываются четыре бита показателя степени. Обратите внимание, что, как объясняется в тексте после алгоритма просмотровой схемы возведения в степень, достаточно подсчитать только точки, кратные следующим: $P, 3P, 5P, 7P$. Конечно, все эти умножения следует подсчитать заранее каким-либо оптимальным способом.

Далее, изучите выбор других схем возведения в степень (дополнения к упражнениям такого типа показывают, насколько запутана эта область исследований), как описывается в [Müller 1997], [De Win et al. 1998], [Crandall 1999b] и содержащихся в них ссылках. Один из примеров предпринимаемых попыток улучшения метода: некоторые исследователи рассматривали такие показатели степени, в которых было какое-то заранее известное число нулей между другими цифрами. Также имеется особое преимущество, возникающее в методах, где показатель степени может быть сильно сжат [Yacobi 1999], такие исследования еще более запутываются возможностью упаковки, зависящей от выбора основания системы счисления. Интересная исследовательская задача — установление точной связи между сжимаемостью показателя и оптимальной сложностью возведения в такую степень.

**9.78.** В связи с результатами для сложности, такими как в упражнении 9.37, могло показаться, что версия алгоритма Тоома—Кука для больших $D$ могла

бы с помощью рекурсии быть сведена к идеальной оценке битовой сложности $O\left(\ln^{1+\epsilon} N\right)$. Однако, как мы упомянули, количество сложений при этом растет стремительно. Оцените количество сложений в методе Тоома—Кука и обсудите дисбаланс между очень низким количеством умножений и невероятным количеством сложений. Обратите внимание также на существование оптимизации количества сложений, как это показано в упражнении 9.38.

Обсуждаемый вопрос сложен, однако он имеет очевидное практическое значение. Например, нам *a priori* ничто не мешает использовать разные, перемежающиеся методы Тоома—Кука с одним большим рекурсивным умножением. Ясно, что для оптимизации такой большой смешанной схемы требуется что-то знать о взаимодействии количества умножений и сложений, также как и о других проявлениях накладных расходов. Еще один такой аспект включает в себя сдвиги и перестановку данных, необходимые для того, чтобы перевести целое число к виду коэффициентов из метода Тоома—Кука.

**9.79.** Насколько далеко надо численно уметь проверять гипотезу Гольдбаха, рассматривая ациклическую свертку сигнала

$$G = (1, 1, 1, 0, 1, 1, 0, 1, 1, 0, \ldots)$$

самого с собой? (В этом примере, как и в примере из основного текста, элемент сигнала $G_n$ равен 1 тогда и только тогда, когда $2n + 3$ — простое.) Чему равна вычислительная сложность подхода, основанного на использовании сверток, для того чтобы доказать гипотезу Гольдбаха для всех четных чисел, не превосходящих $x$? Заметим, что она была установлена для чисел вплоть до $x = 4 \cdot 10^{14}$ [Richstein 2001]. Отметим также, что явные вычисления, основанные на БПФ, для чисел до $10^8$ и в самом деле были проведены в [Lavenier and Saouter 1998]. Любопытный вопрос: можно ли найти представления Гольдбаха с помощью только целочисленных сверток массивов $b$-битовых целых чисел (например, при $b = 16$ или 32), где единицами были бы отмечены места простых чисел, и заранее известно, что наличие двух битов простых чисел в одном целом числе — довольно редкое событие?

**9.80.** Можно использовать идеи свертки для анализа определенных аддитивных проблем более высокого порядка в кольцах $\mathbf{Z}_N$, и, вероятно, в более сложных условиях, ведущих в интересные области исследований. Обратите внимание, что упражнение 9.41 имеет дело с суммами квадратов. Когда используются более высокие степени, использование сверток и манипуляций со спектральными разложениями становится проблематичным.

Чтобы прочувствовать суть изложенного здесь исследовательского подхода, рассмотрите вначале тригонометрическую сумму $k$-х степеней (квадратичный и кубический варианты которой появлялись в упражнении 1.66), а именно, сумму

$$U_k(a) = \sum_{x=0}^{N-1} e^{2\pi i a x^k / N}.$$

Обозначим через $r_s(n)$ количество представлений $n$ в виде суммы $s$ $k$-х степе-

ней в кольце $\mathbf{Z}_N$. Докажите, учитывая равенство

$$\sum_{n=0}^{N-1} r_s(n) = N^s,$$

что также выполнено равенство

$$\sum_{n=0}^{N-1} r_s(n)^2 = \frac{1}{N} \sum_{a=0}^{N-1} |U_k(a)|^{2s}.$$

Именно последнее соотношение ведет к нескольким интересным оценкам и выводам. В самом деле, спектральная сумма степеней $|U|^{2s}$, если она оценена сверху, позволяет установить нижние оценки количества представимых элементов $\mathbf{Z}_N$. Другими словами, верхние оценки спектральной амплитуды $|U|$ явно управляют количеством представлений во всем кольце, что выгодно с аналитической точки зрения.

Далее, в качестве начального подступа к открывающимся исследованиям, используйте идеи и результаты упражнений 1.44, 1.66, чтобы показать, что существует положительная константа $c$, такая что для всякого простого $p$ доля элементов $\mathbf{Z}_p$, являющихся суммой двух кубов, составляет не меньше, чем $c$. Конечно, мы видели, что теория эллиптических кривых полностью разрешает вопрос двух кубов — даже для колец $\mathbf{Z}_N$, где $N$ — составное — способом, описанным в упражнении 7.20, однако идея данного упражнения заключается в том, чтобы использовать только свертку и представление о спектре. Насколько большой можно сделать константу $c$, например, для существенно больших $p$? Один из способов состоит в том, чтобы вывести из оценки вида «$p^{3/4}$» упражнения 1.66, что всякий элемент $\mathbf{Z}_p$ является суммой 5 кубов, затем получить более точные результаты, используя наилучшую возможную оценку вида «$p^{1/2}$». Что дает такой спектральный подход для составного $N$? В этом случае можно применить для соответствующих индексов Фурье $a$ оценку «$N^{2/3}$» (пример см. в [Vaughan 1997, Theorem 4.2]).

Попробуйте найти простое доказательство следующей теоремы: если $N$ простое, то для любого $k$ существуют положительные константы $c_k, \varepsilon_k$, такие что для всякого $a \not\equiv 0 \pmod N$ верно

$$|U_k(a)| < c_k N^{1-\varepsilon_k}.$$

Выведите из этой теоремы, что для любого $k$ имеется фиксированное число $s$ (не зависящее ни от чего, кроме $k$), такое что каждый элемент $\mathbf{Z}_N$ (где $N$ — простое) является суммой $s$ $k$-х степеней. Такие границы, как верхняя граница для $|U|$, не очень сложно установить, используя рекурсию приема Вейля, как это сделано в упражнении 1.66. (Некоторые ссылки далее объясняют, как работать дальше, чтобы получить, что на самом деле $\varepsilon_k \approx 1/k$.)

Можете ли Вы доказать существование фиксированного $s$ для составного $N$? Можете ли Вы установить точные значения $s$ для разных $k$? (Вспомните о проблеме выбора «4 или 5» в кубическом случае.) В таких исследованиях Вам бы пришлось найти верхние оценки для суммы $U$ в общем случае, и они в самом деле были получены; см. [Vinogradov 1985], [Ellison and Ellison 1985], [Nathanson

1996], [Vaughan 1997]. Однако наиболее сложная часть задачи — установить явные значения $s$, что означает, что при оценках придется отслеживать явные значения констант.

Одной из наиболее удивительных составляющих этой области исследований является смешение теоретических выкладок и прямых вычислений. Это значит, если у Вас есть параметры оценки $c_k, \varepsilon_k$ для задачи $k$-ой степени, то вероятно, что Вы окажетесь в ситуации, когда теория работает для «достаточно больших» $N$, в то время как Вам нужно, чтобы вычисления работали для всех значений $N$ от единицы и до этого теоретического порога. Вычисления начинают играть еще большую роль, когда константа $c_k$ велика, или, в меньшей степени, когда $\varepsilon_k$ мало. В свете этого, огромные теоретические усилия XX века по установлению общих оценок для тригонометрических сумм теперь можно рассматривать с вычислительной точки зрения.

Эти исследования, разумеется, напоминают о литературе по поводу знаменитой проблемы Варинга, которая утверждает о представимости любого неотрицательного числа в виде фиксированного количества $s$ $k$-х степеней (например, теорема Лагранжа о представимости любого числа в виде суммы четырех квадратов из упражнения 9.41 доказывает частный случай для $k = 2,\ s = 4$). Вопросы в полном случае Варинга другие, поскольку, во-первых, тригонометрические суммы берутся не по всем элементам кольца, а лишь до номера $x \approx \lfloor N^{1/k} \rfloor$, соответственно, и процедуры построения оценок более сложные. Несмотря на такие препятствия, хорошим исследованием было бы выяснение оценок для $s$ при заданном $k$ в классическом случае Варинга — все такие оценки исторически включают в себя непрерывные интегралы — используя исключительно методы дискретных сверток. (В 1909 г. Гильберт решил проблему Варинга с помощью искусного комбинаторного подхода, в то время как во многих работах используются точные и мощные методы математического анализа, например, [Hardy 1966], [Nathanson 1996], [Vaughan 1997].) Кстати, многие вопросы того же типа, что и проблема Варинга, были полностью решены; см., например, [Winterhof 1998].

**9.81.** Существует ли способ обработки больших сверток без использования ДПФ, с использованием идей вроде матричного метода, лежащего в основе алгоритма 9.5.7? Т. е. предлагается вычислять свертку маленькими частями, используя обычный подход: сворачиваемые сигналы можно хранить на вместительном носителе (например, жестком диске), в то время как вычисления требуют сравнительно мало памяти (около размера строки или столбца таблицы).

Следуя этому принципу, разработайте стандартный метод с тремя преобразованиями БПФ для свертки произвольных сигналов, за исключением того, что метод должен работать в матричной форме, подобно тому, как это делается в алгоритме 9.5.7, и при этом *не делать* излишних транспонирований. Подсказка: организуйте первое преобразование БПФ, чтобы оно оставляло данные в таком виде, что после применения обычного покомпонентного умножения обратное преобразование Фурье могло сразу начинать работу с БПФ строк.

Кстати, Майер построил схемы для БПФ, не требующие вообще никаких транспозиций; вместо этого используются преобразования Фурье столбцов, которые избегают типичных проблем с памятью. По поводу обсуждения метода Майера см. [Crandall et al. 1999].

**9.82.** Одно простое число, предложенное в [Craig-Wood 1998], а именно,

$$p = 2^{64} - 2^{32} + 1,$$

имеет свойства, полезные в методе вычисления свертки, основанном на использовании китайской теоремы об остатках. Изучите некоторые из этих свойств, например, выяснив, каковы допустимые длины сигналов для теоретико-числовых преобразований по модулю $p$, указав элемент порядка 64 с малой амплитудой (такие элементы могут появляться в некоторых алгоритмах БПФ), и т.д.

**9.83.** Вот интересный результат: циклическую свертку для сигналов длины 8 по модулю простого числа Мерсенна можно вычислить всего за одиннадцать умножений. Это неожиданно, потому что оценка Винограда дает как раз $2 \cdot 8 - 4 = 12$ умножений, как было показано в упражнении 9.39. Конечно, разгадка заключается в том, что выбор в качестве модуля числа Мерсенна слегка изменяет задачу.

Чтобы объяснить это явление, докажите существование корня восьмой степени из единицы в поле $\mathbf{F}_{p^2}$, где $p$ — простое число Мерсенна, а корень записывается достаточно простой формулой, чтобы можно было выполнять преобразования Галуа (ДПГ) без собственно целочисленного умножения. Затем рассмотрите преобразование Галуа длины 8, использованное для вычисления циклической свертки $x$, $y$. Далее, докажите, что сигналы $X, Y$ имеют достаточную степень симметрии, так что покомпонентное произведение $X * Y$ на самом деле требует двух вещественных и трех комплексных умножений. Это как раз дает заявленные 11 умножений.

Открытым вопросом является вопрос, имеются ли подобные нарушения оценки Винограда для каких-либо длин, больших восьми.

**9.84.** Изучите интересные результаты работы [Yagle 1995], в которой отмечено, что матричное умножение для матриц размера $n \times n$ можно выполнить с помощью свертки длины $n^3$. Это не очень полезно, поскольку мы не можем выполнять произвольную свертку длины $n$ быстрее, чем за $O(n \ln n)$ операций. Однако Ягль заметил, что рассматриваемая свертка является *разреженной*, что ведет к интересным исследованиям, затрагивающим в том числе и теоретико-числовые преобразования.

**Приложение**

# ПСЕВДОКОД КНИГИ

Все алгоритмы в этой книге написаны на особом псевдокоде, который можно описать, пожалуй, как смешение естественного языка и C. Причины такого выбора языка были изложены в предисловии, где мы выразили надежду, что такое «смешение» языков позволит всем читателям понимать алгоритмы, а программистам — их реализовывать. Также в предисловии мы указали адреса в сети Интернет, где можно найти реализации алгоритмов книги на языке *Mathematica*.

С учетом сказанного, назначение этого приложения заключается не в том, чтобы предоставить строгий справочник определений допустимых инструкций, поскольку это потребовало бы чего-то вроде полного собрания синтаксических правил, как это можно было ожидать от типичного справочника по C. Вместо этого мы далее приводим несколько явных примеров того, как следует понимать определенные высказывания нашего псевдокода.

### Естественный язык и комментарии

Для наиболее сложных математических преобразований внутри псевдокода мы используем описание на естественном языке. Основная причина, по которой мы это делаем, хорошо объясняется следующим примером. Высказывание на языке C

`if((`$n$`== floor(`$n$`)) && (`$j$` == floor(sqrt(`$j$`))*floor(sqrt(`$j$`)))) ...,`

на самом деле означающее: «если $n$ — целое число, а $j$ — полный квадрат», можно на нашем языке передать как

$\mathsf{if}(n, \sqrt{j} \in \mathbf{Z})\ \ldots$

То есть мы попытались поместить обычные математические обозначения внутрь условия. Мы также приняли определенные правила формирования отступов. Если бы мы позволили (чего мы на самом деле не сделали) такие высказывания на естественном языке, как:

> Для всех псевдопростых чисел в $S$ применить равенство (X); Применить равенство (Y);

то начинающему программисту могло бы быть неясно, должно ли быть равенство (Y) применено ко всем псевдопростым числам, или лишь один раз, после выполнения цикла для равенства (X). То, как мы хотим, чтобы естественный язык использовался на самом деле для случая, если равенство (Y) нужно применить лишь один раз после завершения цикла, записывается так:

Для всех псевдопростых чисел в $S$ применить равенство (X);
Применить равенство (Y);

Благодаря требованию, что каждая инструкция должна занимать свою строку, интерпретация, что равенство (Y) выполняется лишь один раз, после выполнения цикла по псевдопростым числам, становится очевидной. Соответственно, когда текст инструкции на естественном языке настолько длинен, что переносится на следующую строку, мы используем отступ, например:

Найти случайным образом число $t \in [0, p-1]$, такое что $t^2 - a$ не является квадратичным вычетом по модулю $p$, с помощью алгоритма 2.3.5;
$x = (t + \sqrt{t^2 - a})^{(p+1)/2}$;
. . .;

В последнем примере следует последовательно выбирать случайные целые числа $t$ в указанном промежутке до тех пор, пока не будет найдено число, удовлетворяющее заданным условиям, и затем перейти к следующему шагу, который выполняет вычисление и присваивание переменной $x$ результата вычисления.

Также на естественном языке написаны все комментарии псевдокода книги. Они имеют следующий вид (также они выровнены по правому краю, в отличие от самого псевдокода):

$x = (t + \sqrt{t^2 - a})^{(p+1)/2}$; // Использовать вычисления в поле $\mathbf{F}_{p^2}$.

Дело в том, что комментарии, начинающиеся с «//» *не следует* выполнять как псевдокод. Так, комментарий, приведенный выше, дает полезный совет, показывающий, что для выполнения этой операции следует сначала написать процедуру для вычислений в поле $\mathbf{F}_{p^2}$. Другие комментарии поясняют обозначения псевдокода, или дают дополнительную информацию о том, как именно выполнять текущую инструкцию.

### Присваивание и проверка условий

Мы решили *не использовать* в чем-то популярный синтаксис присваивания $x := y$, вместо этого мы делаем $x$ равным $y$ с помощью простого знака равенства $x = y$. (Отметим, что при таком обозначении для присваивания, символ «=» не означает симметричного отношения: присваиваение $x = y$ — это не та же операция, что присваивание $y = x$.) Поскольку присваивание выглядит как равенство, то знак условного равенства $x == y$ означает, что мы не выполняем присваивания, а только проверяем, являются ли $x$ и $y$ равными. (В случае проверки равенства символ «==» в самом деле симметричен.) Вот примеры типичных присваиваний:

$x = 2$; // Переменная $x$ получает значение 2.
$x = y = 2$; // Обе переменные $x$ и $y$ получают значение 2.
$F = \{\}$; // $F$ становится пустым множеством.
$(a, b, c) = (3, 4, 5)$; // Переменная $a$ становится равной 3, $b$ — 4, $c$ — 5.

Обратите внимание на важное правило, что множественные (векторные) присваивания предполагают, что сначала выполняются все вычисления вектора в

правой части присваивания, после чего заменяются значения величин слева. Например, присваивание

$$(x, y) = (y^2, 2x);$$

означает, что вычисляются все компоненты вектора справа, и затем изменяются значения всех компонент вектора слева. Таким образом, этот пример равносилен последовательности (мы в данном случае предполагаем, что ни $x$, ни $y$ не вызывают скрытых функций)

```
t = x;                    // t в данном случае — временная переменная.
x = y^2;
y = 2t;
```

и вполне очевидно, насколько наглядно эффективно однострочное векторное присваивание. Обратите также внимание, что составные операции

```
x = y^2;
y = 2x;
```

и

```
y = 2x;
x = y^2;
```

обе отличаются от векторного присваивания, а также различны между собой.

Поскольку мы придерживаемся правила, что упорядоченные последовательности обозначаются круглыми скобками (например, $(x_n)$), в то время как для множеств используются фигурные скобки (например, $\{X, a, \alpha\}$), мы выполняем присваивания последовательностей, векторов и т.д., в стиле, согласовывающемся с текстом; так, $\vec{v} = (0, 1, 0)$ — это упорядоченное (векторное) присваивание, в то время как присваивание множества из трех многочленов можно записать как $S = \{x^2 + 1, x, x^3 - x\}$, и порядок многочленов при этом не важен. Более того, $S = \{x^2 + 1, x, x, x^3 - x\}$ — это то же самое присваивание, поскольку запись множеств не учитывает кратности вхождения элементов. Отметим, что различие между присваиваниями векторов и множеств важно, с учетом вольного обращения со скобками в новых языках. В языке *Mathematica* фигурные скобки означают «списки», с которыми можно обращаться, как с векторами (последовательностями) или множествами, при этом доступны линейные векторные операции (такие как умножение матрицы на вектор) и теоретико-множественные операции (такие как объединение и пересечение). Подобным образом, язык C позволяет выполнять присваивания записей данных с помощью фигурных скобок, как в «float $x[3] = \{1.1, 2.2, 3.3\}$;», что заполняет $x$ по векторному правилу. В последнем случае в нашем псевдокоде запись бы выглядела как $x = (1.1, 2.2, 3.3)$.

Условия внутри операторов if() часто используют классические математические обозначаения, однако не всегда. Давайте приведем несколько примеров синтаксиса условных выражений:

```
if(x == y) task();        // Проверить, равны ли числа x, y, не изменяя их.
```

```
if(x ⩾ y) task();                    // Проверить, верно ли, что x не меньше y.
if(x|y) task();                      // Проверить, верно ли, что x делит y.
if(x ≡ y (mod p)) task();            // Проверить, сравнимы ли x, y по модулю p.
```

Отметим, что сравнение по модулю не требует записи вида $x \equiv\equiv y \pmod p$, поскольку в таких случаях не может возникнуть никакой путаницы с присваиванием. Однако допустима конструкция $x == y \bmod p$, поскольку, как объясняется в тексте книги, запись $y \bmod p$ обозначает целое число $y - p\lfloor y/p \rfloor$. В таком случае может оказаться так, что $x$ равно этому числу, или может оказаться так, что мы хотим присвоить $x$ это значение (в последнем случае мы бы написали $x = y \bmod p$).

Другой вид условия дает цикл while(), такой как

```
while(x ≠ 0) {
        task1();
        task2();
        ...;
}
```

который означает, что значение переменной $x$ проверяется на равенство нулю при входе в цикл, затем все стоящие внутри инструкции должны выполняться до тех пор, пока после некоторого полного прохода не обнаружится, что $x$ равно нулю, и это завершает выполнение цикла while().

Операции, изменяющие значение одной переменной, включают следующие:

```
x = x + c;                           // x увеличивается на c.
x = cx;                              // x умножается на c.
x = x << 3;
     // Целочисленный сдвиг x влево на 3 бита; то же самое, что x = 8x.
x = x >> 3;                  // Сдвиг вправо; то же самое, что x = ⌊x/8⌋.
x = x ∧ 37;      // Побитовое исключающее «или» чисел x и 0...0100101.
x = x & 37;                  // Побитовое «и» чисел x и 0...0100101.
```

### Циклы for()

Циклы for() присутствуют всюду в этой книге, так как они являются нашим основным инструментом выполнения заданий многократно. Снова обойдемся набором примеров и не будем пытаться подробно перечислить все возможные формы допустимых циклов на нашем языке; вместо этого рассмотрим некоторые наиболее частые формы появления их в книге.

```
for(a ⩽ x < b) task();  // Для всех целых x ∈ [a, b), в порядке возрастания.
for(a ⩾ x ⩾ b) task();      // Для всех целых x ∈ [b, a], в порядке убывания.
for(x ∈ [a, b)) task();  // Для всех целых x ∈ [a, b), в порядке возрастания.
```

Отметим, что предполагается, что числа $a$, $b$ в предыдущих примерах соотносятся правильно, чтобы приводить к возрастающему или убывающему порядку; например, если цикл выглядит как for($a \geqslant \ldots$), то $b$ не должно превосходить $a$ (в противном случае цикл считается пустым). Заметим также, что первый

и третий примеры цикла for() эквивалентны; мы просто хотим сказать, что третья форма допустима по нашим правилам. Отметим далее, что ни $a$, ни $b$ не обязательно целые числа. Вот почему мы не могли поместить в качестве комментария к первому циклу for() в приведенном примере следующее: «Для $x = a, a+1, a+2, \ldots, b-1$» — хотя такой комментарий допустим, если оба числа $a, b$ — целые, и $a < b$. В соответствии с этим, вот пример того, как преобразуются традиционные математические обозначения в условии цикла for():

```
for(1 ⩽ a и a^2 ⩽ m) task();      // Выполнить задачу для a = 1, 2, ... ⌊√m⌋.
```

примером которого является алгоритм 7.5.8. Другие примеры смешанного естественного языка и C такие:

```
for(простые p|F) task();
    // Выполнить для всех простых p, делящих F.
    // Отметим: значения p идут в возрастающем порядке, если не указано
       обратное.
for(p ∈ 𝒫, p ∈ 𝒮) task();
                    // Выполнить для всех простых p ∈ 𝒮, соблюдая порядок.
for(нечетные j ∈ [1, C]) task();
                        // Выполнить для j = 1, 3, 5, ..., не превосходящих C.
```

Алгоритмы 3.2.1, 4.1.7, 7.4.4 используют такие конструкции цикла for(), как в приведенных примерах. Для более общих условий циклов мы решили использовать стандартный синтаксис C, особенно когда управляющая переменная должна изменяться на нетривиальные величины. В качестве примера такого общего цикла приведем

```
for(j = q; j < B; j = j + p) task();                    // Цикл в стиле C.
```

Если предположить, что $q$ — целое, то этот цикл означает, что $j$ принимает значения $q,\ q+p, q+2p, \ldots, q+kp$, где $k$ — наибольшее целое число, строго меньшее $(B-q)/p$. Алгоритм 3.2.1 является примером использования такого более общего цикла for(). Кстати, для не программистов хорошим правилом поведения является развеяние путаницы в следующем вопросе: когда именно выполняются операции из тела цикла? Глядя на этот цикл for(), мы можем сформулировать правило таким образом: task() *никогда* не будет выполнен, если среднее условие ложно; то есть если $j \geqslant B$, то операции цикла не будут выполнены, и цикл завершится. Другое правило заключается в следующем: увеличение $j = j + p$ происходит *после* выполнения операций цикла (везде в нашем псевдокоде мы предполагаем, что операции не изменяют управляющей переменной цикла). Так что можно представлять, что после каждого выполнения тела цикла $j$ увеличивается на $p$, а затем проверяется среднее условие.

### Управление программой

Наш псевдокод должен исполняться, начиная сверху, хотя в некоторых случаях мы помещаем туда вызываемые функции и процедуры. В таких случаях

порядок размещения не важен, и исполнение на самом деле начинается с первой метки, следующей после того как определены функции и процедуры. В любом случае мы оформляем с помощью отступа инструкции, следующие за метками, оформленными в квадратные скобки [ ], например:

```
3. [Проверить простоту p]
       Инструкция с отступом;
       Инструкция с отступом;
       ...;
```

где инструкции выполняются последовательно, сверху вниз (за исключением, конечно, случаев, когда встречается операция goto [Другая метка]; см. ниже по поводу операции goto). Важно отметить, что на такой метке, как [Проверить простоту $p$] в примере выше, мы не предполагаем выполнения инструкций непосредственно на метке. Метка никогда не является выполнимой инструкцией. (Во многом так же, как и комментарии, начинающиеся с «//», в которых задания описываются, а не исполняются.) В примере выше мы ожидаем, что проверка простоты проводится где-то ниже самой метки, посредством собственно инструкций с отступом.

Таким образом, мы пишем метки на естественном языке, с тем чтобы они описывали идею следующих за ними инструкций, вплоть до следующей метки. Этот последовательный, идущий сверху вниз, порядок абсолютен. Например, метку из нашего примера или любую другую метку такого же рода можно истолковать как [Далее, проверить простоту $p$]; в случае определений функций и процедур метка означает [Далее, определим функцию или процедуру].

В некоторых случаях псевдокод был упрощен с помощью использования операций «goto», как в высказывании «goto [Проверить простоту $p$]», которая направляет нас к указанной метке, и начиная с нее мы продолжаем выполнять инструкции, следуя сверху вниз.

Все наши циклы в псевдокоде используют фигурные скобки { и }, чтобы отметить начало и конец операций цикла. Такое использование фигурных скобок не связано с их использованием для обозначения множеств. Также использование фигурных скобок для обозначения блока для функции или процедуры (см. следующий раздел) не зависит от обозначения множеств.

### Функции и значения возврата/отчета

Типично новые функции в нашем псевдокоде имеют вид:

```
func(x) {
        ...;
        ...;
        return y;
}
```

и используемая идея — та же самая, что и в большинстве современных языков: вызов функции $func(x)$ выполняется точно так же, как вызов тригонометрической функции или квадратного корня, и дает значение $y$. Процедуры (как

противопоставление функциям) имеют тот же самый синтаксис, но не возвращают никакого значения, хотя некоторые переменные обычно изменяются в течение исполнения процедуры. Помимо этого, инструкция return является инструкцией выхода; например, последовательность операций

```
if(x ≠ y) return x^3;
return x^4;
```

не требует использования конструкции «else» для случая $x^4$, поскольку мы всегда предполагаем, что текущая процедура или функция *завершается мгновенно* по любому требованию, в данном случае благодаря инструкции if(). Подобным образом, инструкция return, когда выполняется, мгновенно производит выход из любого цикла while() или for().

Наконец, мы используем инструкцию report следующим образом. Вместо того чтобы возвращать значение из функции/процедуры, оператор report просто сообщает о нем — например, печатает или отчитывается о нем другой программе — на лету сообщает, каким оно было. Так, следующая функция дает пример использования report/return (код функции предполагает наличие подпрограммы, вычисляющей количество делителей $d(n)$):

```
mycustomπ(x) {
// Сообщить обо всех простых числах, не превосходящих x,
// и подсчитать их количество.
    c = 0;                          // c будет количеством простых чисел.
    for(2 ⩽ n ⩽ x) {
        if(d(n) == 2) {
            c = c + 1;
            report n;               // Как если бы было написано «print n».
        }
    }
    return c;
}
```

О простых числах будет сообщено в порядке возрастания, а значение, возвращаемое функцией $mycustom\pi(x)$, будет равно классическому $\pi(x)$.

# Список литературы


[Adleman and Huang 1992] L. Adleman and M.-D. Huang. *Primality testing and abelian varieties over finite fields*, volume 1512 of *Lecture Notes in Mathematics*. Springer–Verlag, 1992.

[Adleman 1994] L. Adleman. The function field sieve. In L. Adleman and M.-D. Huang, editors, *Algorithmic Number Theory: Proc. ANTS-I, Ithaca, NY*, volume 877 of *Lecture Notes in Computer Science*, pages 108–121. Springer–Verlag, 1994.

[Adleman and Lenstra] L. Adleman and H. Lenstra, Jr. Finding irreducible polynomials over finite fields. In *Proc. 18th Annual ACM Symposium on the Theory of Computing*, pages 350–355, 1986.

[Adleman et al. 1983] L. Adleman, C. Pomerance, and R. Rumely. On distinguishing prime numbers from composite numbers. *Ann. of Math.*, 117:173–206, 1983.

[Agarwal and Cooley 1986] R. Agarwal and J. Cooley. Fourier transform and convolution subroutines for the IBM 3090 vector facility. *IBM Journal of Research and Development*, 30:145–162, 1986.

[Agrawal 2003] M. Agrawal. PRIMES is in P. `http://www.fields.utoronto.ca/audio/02-03/agrawal/agrawal/`.

[Agrawal et al. 2002] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. `http://www.cse.iitk.ac.in/news/primality.html`.

[Agrawal et al. 2004] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. *Ann. of Math.*, 160:781–793, 2004.

**[Aho et al. 1974]** A. Aho, J. Hopcroft, and J. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974[1].

[Alford et al. 1994a] W. Alford, A. Granville, and C. Pomerance. There are infinitely many Carmichael numbers. *Ann. of Math.*, 139:703–722, 1994.

[Alford et al. 1994b] W. Alford, A. Granville, and C. Pomerance. On the difficulty of finding reliable witnesses. In L. Adleman and M.-D. Huang, editors, *Algorithmic Number Theory: Proc. ANTS-I, Ithaca, NY*, volume 877 of *Lecture Notes in Computer Science*, pages 1–16. Springer–Verlag, 1994.

[Alford and Pomerance 1995] W. Alford and C. Pomerance. Implementing the self-initializing quadratic sieve on a distributed network. In *Number-theoretic and algebraic methods in computer science (Moscow, 1993)*, pages 163–174. World Scientific, 1995.

[Alt 1979] H. Alt. Square rooting is as difficult as multiplication. *Computing*, 21:221–232, 1979.


[1] Полужирным шрифтом выделены работы, имеющиеся на русском языке. См. следующий раздел. — *Прим. ред.*

[Apostol 1986] T. Apostol. *Introduction to Analytic Number Theory,* 3rd printing. Springer–Verlag, 1986.

[Arazi 1994] B. Arazi. On primality testing using purely divisionless operations. *The Computer Journal*, 37:219–222, 1994.

[Archibald 1949] R. Archibald. Outline of the history of mathematics. *Amer. Math. Monthly*, 56, 1949. The second Herbert Ellsworth Slaught Memorial Paper: supplement to no. 1 issue, 114 pp.

[Ares and Castro 2004] S. Ares and M. Castro. Hidden structure in the randomness in the prime number sequence. *Condensed Matter Abstracts*, 2004. `http://arxiv.org/abs/cond-mat/0310148`.

[Arney and Bender 1982] J. Arney and E. Bender. Random mappings with constraints on coalescence and number of origins. *Pacific J. Math.* 103:269–294, 1982.

[Artjuhov 1966/67] M. Artjuhov. Certain criteria for the primality of numbers connected with the little Fermat theorem (Russian). *Acta Arith.*, 12:355–364, 1966/67.

[Ashworth and Lyne 1988] M. Ashworth and A. Lyne. A segmented FFT algorithm for vector computers. *Parallel Computing*, 6:217–224, 1988.

[Atkin 1986] A. Atkin. Schoof's algorithm. Unpublished manuscript, 1986.

[Atkin 1988] A. Atkin. The number of points on an elliptic curve modulo a prime (i). Unpublished manuscript, 1988.

[Atkin 1992] A. Atkin. The number of points on an elliptic curve modulo a prime (ii). Unpublished manuscript, 1992.

[Atkin and Bernstein 2004] A. Atkin and D. Bernstein. Prime sieves using binary quadratic forms. *Math. Comp.*, 73:1023–1030, 2004.

[Atkin and Morain 1993a] A. Atkin and F. Morain. Finding suitable curves for the elliptic curve method of factorization. *Math. Comp.*, 60:399–405, 1993.

[Atkin and Morain 1993b] A. Atkin and F. Morain. Elliptic curves and primality proving. *Math. Comp.*, 61:29–68, 1993.

[Bach 1985] E. Bach. *Analytic Methods in the Analysis and Design of Number-Theoretic Algorithms.* A 1984 ACM Distinguished Dissertation. The MIT Press, 1985.

[Bach 1990] E. Bach. Explicit bounds for primality testing and related problems. *Math. Comp.*, pages 355–380, 1990.

[Bach 1991] E. Bach. Toward a theory of Pollard's rho method. *Inform. and Comput.*, 90:139–155, 1991.

[Bach 1997a] E. Bach. The complexity of number-theoretic constants. *Inform. Process. Lett.*, 62:145–152, 1997.

[Bach 1997b] E. Bach. Comments on search procedures for primitive roots. *Math. Comp.*, 66(220):1719–1727, 1997.

[Bach and Shallit 1996] E. Bach and J. Shallit. *Algorithmic Number Theory*, volume I. MIT Press, 1996.

[Baillie and Wagstaff 1980] R. Baillie and S. Wagstaff, Jr. Lucas pseudoprimes. *Math. Comp.*, 35:1391–1417, 1980.

[Bailey 1990] D. Bailey. FFTs in external or hierarchical memory. *J. Supercomp.*, 4:23–35, 1990.

[Bailey and Crandall 2001] D. Bailey and R. Crandall. On the random character of fundamental constant expansions, *Experiment. Math.*, 10:175–190, 2001.

[Bailey and Crandall 2002] D. Bailey and R. Crandall. Random generators and normal numbers. *Experiment. Math.*, 11:527–546, 2002.

[Bailey et al. 2004] D. Bailey, J. Borwein, R. Crandall, and C. Pomerance. On the binary expansions of algebraic numbers. *Journal de Theorie des Nombres, Bordeaux*, 16:487–518, 2004.

[Balasubramanian and Nagaraj 1997] R. Balasubramanian and S. Nagaraj. Density of Carmichael numbers with three prime factors. *Math. Comp.*, 66:1705–1708, 1997.

[Balazard et al. 1999] M. Balazard, E. Saias, and M. Yor. Notes sur la fonction $\zeta$ de Riemann. II. *Adv. Math.*, 143:284–287, 1999.

[Balog 1989] A. Balog. On a variant of the Piatetski-Shapiro prime number theorem. In *Groupe de travail en théorie analytique et élementaire des nombres, 1987–1988*, volume 89-01 of *Publ. Math. Orsay*, pages 3–11. Univ. Paris XI, Orsay, 1989.

[Barrett 1987] P. Barrett. Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In A. Odlyzko, editor, *Advances in Cryptology, Proc. Crypto '86*, volume 263 of *Lecture Notes in Computer Science*, pages 311–323. Springer–Verlag, 1987.

[Bateman et al. 1989] P. Bateman, J. Selfridge, and S. Wagstaff, Jr. The new Mersenne conjecture. *Amer. Math. Monthly*, 96:125–128, 1980.

[Bays and Hudson 2000a] C. Bays and R. Hudson. Zeroes of Dirichlet $L$-functions and irregularities in the distibution of primes. *Math. Comp.*, 69:861–866, 2000.

[Bays and Hudson 2000b] C. Bays and R. Hudson. A new bound for the smallest $x$ with $\pi(x) > \mathrm{li}\,(x)$. *Math. Comp.*, 69:1285–1296, 2000.

[Bernstein 1997] D. Bernstein. Multidigit multiplication for mathematicians, 1997. `http://cr.yp.to/arith.html#m3`.

[Bernstein 1998] D. Bernstein. Bounding smooth integers (extended abstract). In [Buhler 1998], pages 128–130.

[Bernstein 2003] D. Bernstein. Proving primality in essentially quartic time. `http://cr.yp.to/ntheory.html#quartic`.

[Bernstein 2004a] D. Bernstein. Scaled remainder trees. `http://cr.yp.to/papers.html#scaledmod`.

[Bernstein 2004b] D. Bernstein. Factorization myths. `http://cr.yp.to/talks.html#2004.06.14`.

[Bernstein 2004c] D. Bernstein. Doubly focused enumeration of locally square polynomial values. In *High primes and misdemeanours: lectures in honour of the 60th birthday of Hugh Cowie Williams*, volume 41 of Fields Inst. Commun., pages 69–76. Amer. Math. Soc., 2004.

[Bernstein 2004d] D Bernstein. How to find smooth parts of integers. `http://cr.yp.to/papers.html#smoothparts`.

[Bernstein 2004e] D. Bernstein. Fast multiplication and its applications. In J. Buhler and P. Stevenhagen, editors *Cornerstones in algorithmic number theory* (tentative title), a Mathematical Sciences Research Institute Publication. Cambridge University Press, to appear.

[Berrizbeitia 2002] P. Berrizbeitia. Sharpening "PRIMES is in P" for a large family of numbers. `http://arxiv.org/find/grp_math/1/au:+Berrizbeitia/0/1/0/all/0/1`.

[Berry 1997] M. Berry. Quantum chaology. *Proc. Roy. Soc. London Ser. A*, 413:183–198, 1987.

[Berta and Mann 2002] I. Berta and Z. Mann. Implementing elliptic-curve cryptography on PC and Smart Card. *Periodica Polytechnica, Series Electrical Engineering*, 46:47–73, 2002.

[Beukers 2004] F. Beukers. The diophantine equation $Ax^p + By^q = Cz^r$. `http://www.math.uu.nl/people/beukers/Fermatlectures.pdf`.

[Blackburn and Teske 1999] S. Blackburn and E. Teske. Baby-step giant-step algorithms for non-uniform distributions. Unpublished manuscript, 1999.

[Bleichenbacher 1996] D. Bleichenbacher. *Efficiency and security of cryptosystems based on number theory.* PhD thesis, Swiss Federal Institute of Technology Zürich, 1996.

[Blum et al. 1986] L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudorandom number generator. *SIAM J. Comput.*, 15:364–383, 1986.

[Bombieri and Iwaniec 1986] E. Bombieri and H. Iwaniec. On the order of $\zeta(1/2+it)$. *Ann. Scuola Norm. Sup. Pisa Cl. Sci. (4)*, 13:449–472, 1986.

[Bombieri and Lagarias 1999] E. Bombieri and J. Lagarias. Complements to Li's criterion for the Riemann hypothesis. *J. Number Theory*, 77:274–287, 1999.

[Boneh 1999] D. Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices Amer. Math. Soc.*, 46:203–213, 1999.

[Boneh and Venkatesan 1998] D. Boneh and R. Venkatesan. Breaking RSA may not be equivalent to factoring. In *Advances in Cryptology, Proc. Eurocrypt '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 25–34. Springer–Verlag, 1998.

[Borwein 1991] P. Borwein. On the irrationality of $\sum(1/(q^n + r))$. *J. Number Theory*, 37:253–259, 1991.

[Borwein and Borwein 1987] J. Borwein and P. Borwein. *Pi and the AGM: A Study in Analytic Number Theory and Computational Complexity.* John Wiley and Sons, 1987.

[Borwein et al. 2000] J. Borwein, D. Bradley, and R. Crandall. Computational strategies for the Riemann zeta function. *J. Comp. App. Math.*, 121:247–296, 2000.

[Bosma and van der Hulst 1990] W. Bosma and M.-P. van der Hulst. *Primality proving with cyclotomy.* PhD thesis, University of Amsterdam, 1990.

[Bosselaers et al. 1994] A. Bossalaers, R. Govaerts, and J. Vandewalle. Comparison of three modular reduction functions. In D. Stinson, editor, *Advances in Cryptology, Proc. Crypto '93*, volume 773 in Lecture Notes in Computer Science, pages 175–186. Springer–Verlag, 1994.

[Boyle et al. 1995] P. Boyle, M. Broadie, and P. Glasserman. Monte Carlo methods for security pricing. Unpublished manuscript, June 1995.

[Bratley and Fox 1988] P. Bratley and B. Fox. ALGORITHM 659: Implementing Sobol's quasirandom sequence generator. *ACM Trans. Math. Soft.*, 14:88–100, 1988.

[Bredihin 1963] B. Bredihin. Applications of the dispersion method in binary additive problems. *Dokl. Akad. Nauk. SSSR*, 149:9–11, 1963.

[Brent 1979] R. Brent. On the zeros of the Riemann zeta function in the critical strip. *Math. Comp.*, 33:1361–1372, 1979.

[Brent 1994] R. Brent. On the period of generalized Fibonacci recurrences. *Math. Comp.*, 63:389–401, 1994.

[Brent 1999] R. Brent. Factorization of the tenth Fermat number. *Math. Comp.*, 68:429–451, 1999.

[Brent et al. 1993] R. Brent, G. Cohen, and H. te Riele. Improved techniques for lower bounds for odd perfect numbers. *Math. Comp.*, 61:857–868, 1993.

[Brent et al. 2000] R. Brent, R. Crandall, K. Dilcher, and C. van Halewyn. Three new factors of Fermat numbers. *Math. Comp.*, 69: 1297–1304, 2000.

[Brent and Pollard 1981] R. Brent and J. Pollard. Factorization of the eighth Fermat number. *Math. Comp.*, 36:627–630, 1981.

[Bressoud and Wagon 2000] D. Bressoud and S. Wagon. *A Course in Computational Number Theory.* Key College Publishing, 2000.

[Brillhart et al. 1981] J. Brillhart, M. Filaseta, and A. Odlyzko. On an irreducibility theorem of A. Cohn. *Canad. J. Math.*, 33:1055–1059, 1981.

[Brillhart et al. 1988] J. Brillhart, D. Lehmer, J. Selfridge, B. Tuckerman, and S. Wagstaff, Jr. *Factorizations of $b^n \pm 1$, $b = 2, 3, 5, 6, 7, 10, 11, 12$ up to high powers.* Second edition, volume 22 of *Contemporary Mathematics.* Amer. Math. Soc., 1988.

[Bruin 2005] N. Bruin. The primitive solutions to $x^3+y^9 = z^2$. *J. Number Theory*, 111: 179–189, 2005.
`http://arxiv.org/find/math/1/au:+Bruin_N/0/1/0/all/0/1`.

[Buchmann et al. 1997] J. Buchmann, M. Jacobson, Jr., and E. Teske. On some computational problems in finite groups. *Math. Comp.*, 66:1663–1687, 1997.

[Buell and Young 1988] D. Buell and J. Young. The twentieth Fermat number is composite. *Math. Comp.*, 50:261–263, 1988.

[Buhler 1991] J. Buhler, 1991. Private communication.

[Buhler 1998] J. Buhler, editor. *Algorithmic Number Theory: Proc. ANTS-III, Portland, OR*, volume 1423 of *Lecture Notes in Computer Science.* Springer–Verlag, 1998.

[Buhler 2000] J. Buhler, R. Crandall, R. Ernvall, T. Metsänkylä, and M. Shokrollahi. Irregular primes and cyclotomic invariants to 12 million. *J. Symbolic Comput.*, 11:1–8, 2000.

[Buhler et al. 1993] J. Buhler, H. Lenstra, Jr., and C. Pomerance. Factoring integers with the number field sieve. In A. Lenstra and H. Lenstra, Jr., editors, *The development of the number field sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 50–94. Springer–Verlag, 1993.

[Bürgisser et al. 1997] P. Bürgisser, M. Clausen, and M. Shokrollahi. *Algebraic Complexity Theory.* Springer–Verlag, 1997.

[Burnikel and Ziegler 1998] C. Burnikel and J. Ziegler. Fast recursive division. Max-Planck-Institut für Informatik Research Report MPI-I-98-1-022, 1998.
`http:www.algorilla.de/Download/FastRecursiveDivision.ps.gz`.

[Burthe 1996] R. Burthe. Further investigations with the strong probable prime test. *Math. Comp.*, 65:373–381, 1996.

[Burthe 1997] R. Burthe. Upper bounds for least witnesses and generating sets. *Acta Arith.*, 80:311–326, 1997.

[Caldwell 1999] C. Caldwell. Website for prime numbers, 1999.
`http://primes.utm.edu/`.

[Canfield et al. 1983] E. Canfield, P. Erdős, and C. Pomerance. On a problem of Oppenheim concerning "factorisatio numerorum". *J. Number Theory*, 17:1–28, 1983.

[Cassels 1966] J. Cassels. Diophantine equations with special reference to elliptic curves. *J. London Math. Soc.*, 41:193–291, 1966.

[Cesari 1998] G. Cesari. Parallel implementation of Schönhage's integer GCD algorithm. In [Buhler 1998], pages 64–76.

[Chen 1966] J. Chen. On the representation of a large even integer as the sum of a prime and the product of at most two primes. *Kexue Tongbao*, 17:385–386, 1966.

[Cochrane 1987] T. Cochrane. On a trigonometric inequality of Vinogradov. *J. Number Theory*, 27:9–16, 1987.

[Cohen 2000] H. Cohen. *A Course in Computational Algebraic Number Theory*, volume 138 of *Graduate Texts in Mathematics.* Springer–Verlag, 2000.

[Cohen et al. 1998] H. Cohen, A. Miyaji, and T. Ono. Efficient elliptic curve exponentiation using mixed coordinates. In *Advances in Cryptology, Proc. Asiacrypt '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 51–65. Springer–Verlag, 1998.

[Contini 1997] S. Contini. *Factoring integers with the self-initializing quadratic sieve.* Masters thesis, U. Georgia, 1997.

[Cooley and Tukey 1965] J. Cooley and J. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Comp.*, 19:297–301, 1965.

[Copeland and Erdős 1946] A. Copeland and P. Erdős. Note on normal numbers. *Bull. Amer. Math. Soc.*, 52:857–860, 1946.

[Coppersmith 1993] D. Coppersmith. Modifications to the number field sieve. *J. Cryptology*, 6:169–180, 1993.

[Coppersmith 1997] D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptology*, 10:233–260, 1997.

[Coppersmith et al. 2004] D. Coppersmith, N. Howgrave-Graham, and S. Nagaraj. Divisors in residue classes, constructively, 2004. `eprint.iacr.org/2004/339.ps`.

[Couveignes 1993] J.-M. Couveignes. Computing a square root for the number field sieve. In A. Lenstra and H. Lenstra, Jr., editors, *The development of the number field sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 95–102. Springer–Verlag, 1993.

[Couveignes and Morain 1994] J.-M. Couveignes and F. Morain. Schoof's algorithm and isogeny cycles. In L. Adleman and M.-D. Huang, editors, *Algorithmic Number Theory: Proc. ANTS-I, Ithaca, NY*, volume 877 of *Lecture Notes in Computer Science*, pages 43–58. Springer–Verlag, 1994.

[Couveignes et al. 1996] J.-M. Couveignes, L. Dewaghe, and F. Morain. Isogeny cycles and the Schoof–Atkin–Elkies algorithm. Unpublished manuscript, 1996.

[Cox 1989] D. Cox. *Primes of the Form $x^2 + ny^2$*. John Wiley and Sons, 1989.

[Craig-Wood 1998] N. Craig-Wood, 1998. Private communication.

[Crandall 1994a] R. Crandall. Method and apparatus for public key exchange in a cryptographic system., 1994. U.S. Patents #5159632 (1992), #5271061 (1993), #5463690 (1994).

[Crandall 1994b] R. Crandall. *Projects in Scientific Computation.* TELOS/Springer–Verlag, 1994.

[Crandall 1996a] R. Crandall. *Topics in Advanced Scientific Computation.* TELOS/Springer–Verlag, 1996.

[Crandall 1996b] R. Crandall. Method and apparatus for Digital Signature Authentication, 1996. U. S. Patent #5581616.

[Crandall 1997a] R. Crandall. The challenge of large numbers. *Scientific American*, pages 58–62, February 1997.

[Crandall 1997b] R. Crandall. Integer convolution via split-radix fast Galois transform, 1997. `http://www.perfsci.com`.

[Crandall 1998] R. Crandall. Recycled (simultaneous) evaluations of the Riemann zeta function. Unpublished manuscript, 1998.

[Crandall 1999a] R. Crandall. Applications of space-filling curves. Unpublished manuscript, 1999.

[Crandall 1999b] R. Crandall. Fast algorithms for elliptic curve cryptography. Unpublished manuscript, 1999.

[Crandall 1999c] R Crandall. Alternatives to the Riemann–Siegel formula. Unpublished manuscript, 1999.

[Crandall 1999d] R. Crandall. Parallelization of Pollard-rho factorization, 1999. `http://www.perfsci.com`.

[Crandall et al. 1997] R. Crandall, K. Dilcher, and C. Pomerance. A search for Wieferich and Wilson primes. *Math. Comp.*, 66:433–449, 1997.

[Crandall et al. 1995] R. Crandall, J. Doenias, C. Norrie, and J. Young. The twenty-second Fermat number is composite. *Math. Comp.*, 64 210:863–868, 1995.

[Crandall and Fagin 1994] R. Crandall and B. Fagin. Discrete weighted transforms and large integer arithmetic. *Math. Comp.*, 62:305–324, 1994.

[Crandall et al. 2003] R. Crandall, E. Mayer, and J. Papadopoulos. The twenty-fourth Fermat number is composite. *Math. Comp.*, 72:1555-1572, 2003.

[Crandall and Garst 2001] R. Crandall and B. Garst. Method and apparatus for fast elliptic encryption with direct embedding, U. S. Patent #6307935, 2001.

[Crandall et al. 2004] R. Crandall, E. Jones, J. Klivington, and D. Kramer. Gigaelement FFTs on Apple G5 clusters. `http://www.apple.com/acg`.

[Crandall and Papadopoulos 2003] R. Crandall and J. Papadopoulos. On the Implementation of AKS-class Primality Tests. `http://www.apple.com/acg`.

[Creutzburg and Tasche 1989] R. Creutzburg and M. Tasche. Parameter determination for complex number-theoretic transforms using cyclotomic polynomials. *Math. Comp.*, 52:189–200, 1989.

[Damgård et al. 1993] I. Damgård, P. Landrock, and C. Pomerance. Average case error estimates for the strong probable prime test. *Math. Comp.*, 61:177–194, 1993.

[Darmon and Granville 1995] H. Darmon and A. Granville. On the equations $z^m = F(x, y)$ and $Ax^p + By^q = Cz^r$. *Bull. London Math. Soc.*, 27:513–543, 1995.

[Davenport 1980] H. Davenport. *Multiplicative Number Theory* (second edition). Springer–Verlag, 1980.

[Davis 1973] M. Davis. Hilbert's tenth problem is unsolvable. *Amer. Math. Monthly*, 80:233–269, 1973.

[De Win et al. 1998] E. De Win, S. Mister, B. Preneel, and M. Wiener. On the performance of signature schemes based on elliptic curves. In [Buhler 1998], pages 252–266.

[Deléglise and Rivat 1996] M. Deléglise and J. Rivat. Computing $\pi(x)$: the Meissel, Lehmer, Lagarias, Miller, Odlyzko method. *Math. Comp.*, 65:235–245, 1996.

[Deléglise and Rivat 1998] M. Deléglise and J. Rivat. Computing $\psi(x)$. *Math. Comp.*, 67:1691–1696, 1998.

[Deshouillers et al. 1998] J.-M. Deshouillers, H. te Riele, and Y. Saouter. New experimental results concerning the Goldbach conjecture. In [Buhler 1998], pages 204–215.

[Deuring 1941] M. Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkörper. *Abh. Math. Sem. Hansischen Univ.*, 14:197–272, 1941.

[Deutsch 1982] D. Deutsch. Is there a fundamental bound on the rate at which information can be processed? *Phys. Rev. Lett.*, 42:286–288, 1982.

[Deutsch 1985] D. Deutsch. Quantum theory, the Church–Turing principle, and the universal quantum computer. *Proc. Roy. Soc. London Ser. A*, 400:97–117, 1985.

[Dickson 1904] L. Dickson. A new extension of Dirichlet's theorem on prime numbers. *Messenger of Math.*, 33:155–161, 1904.

[Diffie and Hellman 1976] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, 22:644–654, 1976.

[Dilcher 1999] K. Dilcher. Nested squares and evaluation of integer products, 1999. `http://www.mscs.dal.ca/~dilcher/Preprints/nested.ps`.

[Dimitrov et al. 1995] V. Dimitrov, T. Cooklev, and B. Donevsky. Number theoretic transforms over the golden section quadratic field. *IEEE Trans. Sig. Proc.*, 43:1790–1797, 1995.

[Dimitrov et al. 1998] V. Dimitrov, G. Jullien, and W. Miller. A residue number system implementation of real orthogonal transforms. *IEEE Trans. Sig. Proc.*, 46:563–570, 1998.

[Ding et al. 1996] C. Ding, D. Pei, and A. Salomaa. *Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography.* World Scientific, 1996.

[Dixon 1981] J. Dixon. Asymptotically fast factorization of integers. *Math. Comp.*, 36:255–260, 1981.

[Dress and Olivier 1999] F. Dress and M. Olivier. Polynômes prenant des valeurs premières. *Experiment. Math.*, 8:319–338, 1999.

[Dubner et al. 1998] H. Dubner, T. Forbes, N. Lygeros, M. Mizony, and P. Zimmermann. Ten consecutive primes in arithmetic progression, 1998. http://listserv.nodak.edu/archives/nmbrthry.html.

[Dubner and Gallot 2002] H. Dubner and Y. Gallot. Distribution of generalized Fermat numbers. *Math. Comp.* 71:825–832, 2002.

[Dudon 1987] J. Dudon. The golden scale. *Pitch*, I/2:1–7, 1987.

[Dutt and Rokhlin 1993] A. Dutt and V. Rokhlin. Fast Fourier Transforms for Nonequispaced Data. SIAM J. Sci. Comput. 14:1368–1393, 1993.

[Edwards 1974] H. Edwards. *Riemann's Zeta Function.* Academic Press, 1974.

[Ekert and Jozsa 1996] A. Ekert and R Jozsa. Quantum computation and Shor's factoring algorithm. *Rev. Mod. Phys.*, 68:733–753, 1996.

[Elkenbracht-Huizing 1997] M. Elkenbracht-Huizing. *Factoring integers with the Number Field Sieve.* PhD thesis, University of Leiden, 1997.

[Elkies 1991] N. Elkies. Explicit isogenies. Unpublished manuscript, 1991.

[Elkies 1997] N. Elkies. Elliptic and modular curves over finite fields and related computational issues. In J. Teitelbaum, editor, *Computational Perspectives on Number Theory (Chicago, IL, 1995)*, volume 7 of *AMS/IP Stud. Adv. Math.*, pages 21–76. Atkin Conference, Amer. Math. Soc., 1998.

[Ellison and Ellison 1985] W. Ellison and F. Ellison. *Prime Numbers.* John Wiley and Sons, 1985.

[Engelsma 2004 1999] T. Engelsma. Website for k-tuple permissible patterns, 2004. `http://www.opertech.com/primes/k-tuples.html`.

[Erdős 1948] P. Erdős. On arithmetical properties of Lambert series. *J. Indian Math. Soc. (N.S.)*, 12:63–66, 1948.

[Erdős 1950] P. Erdős. On almost primes. *Amer. Math. Monthly*, 57:404–407, 1950.

[Erdős and Pomerance 1986] P. Erdős and C. Pomerance. On the number of false witnesses for a composite number. *Math. Comp.*, 46:259–279, 1986.

[Erdős et al. 1988] P. Erdős, P. Kiss, and A. Sárközy. A lower bound for the counting function of Lucas pseudoprimes. *Math. Comp.*, 51:315–323, 1988.

[Escott et al. 1998] A. Escott, J. Sager, A. Selkirk, and D. Tsapakidis. Attacking elliptic curve cryptosystems using the parallel Pollard rho method. *RSA Cryptobytes*, 4(2):15–19, 1998.

[Estermann 1952] T. Estermann. *Introduction to Modern Prime Number Theory.* Cambridge University Press, 1952.

[Faure 1981] H. Faure. Discrépances de suites associées à un système de numération (en dimension un). *Bull. Soc, Math. France*, 109:143–182, 1981.

[Faure 1982] H. Faure. Discrépances de suites associées à un système de numération (en dimension $s$). *Acta Arith.*, 41:337–351, 1982.

[Fessler and Sutton 2003] J. Fessler and B. Sutton. Nonuniform Fast Fourier Transforms Using Min-Max Interpolation. *IEEE Trans. Sig. Proc.*, 51:560-574, 2003.

[Feynman 1982] R. Feynman. Simulating physics with computers. *Intl. J. Theor. Phys.*, 21(6/7):467–488, 1982.

[Feynman 1985] R. Feynman. Quantum mechanical computers. *Optics News*, II:11–20, 1985.

[Flajolet and Odlyzko 1990] P. Flajolet and A. Odlyzko. Random mapping statistics. In *Advances in cryptology, Eurocrypt '89*, volume 434 of *Lecture Notes in Comput. Sci.*, pages 329–354, Springer—Verlag, 1990.

[Flajolet and Vardi 1996] P. Flajolet and I. Vardi. Zeta Function Expansions of Classical Constants, 1996.
`http://pauillac.inria.fr/algo/flajolet/Publications/Landau.ps`.

[Forbes 1999] T. Forbes. Prime $k$-tuplets, 1999.
`http://www.ltkz.demon.co.uk/ktuplets.htm`.

[Ford 2002] K. Ford. Vinogradov's integral and bounds for the Riemann zeta function. *Proc. London Math. Soc. (3)*, 85:565–633, 2002.

[Fouvry 1985] E. Fouvry. Théorème de Brun–Titchmarsh: application au théorème de Fermat. *Invent. Math.*, 79:383–407, 1985.

[Fraser 1976] D. Fraser. Array permutation by index-digit permutation. *J. ACM*, 23:298–309, 1976.

[Franke et al. 2004] J. Franke, T. Kleinjung, F. Morain, and T. Wirth. Proving the primality of very large numbers with fastECPP. In *Algorithmic number theory: Proc. ANTS VI, Burlington, VT*, volume 3076 of *Lecture Notes in Computer Science*, pages 194–207. Springer-Verlag, 2004.

[Friedlander and Iwaniec 1998] J. Friedlander and H. Iwaniec. The polynomial $X^2 + Y^4$ captures its primes. *Ann. of Math.*, 148:945–1040, 1998.

[Friedlander et al. 2001] J. Friedlander, C. Pomerance, and I. Shparlinski. Period of the power generator and small values of Carmichael's function. *Math. Comp.* 70:1591–1605, 2001.

[Frind et al. 2004] M. Frind, P. Jobling, and P. Underwood. 23 primes in arithmetic progression. `http://primes.plentyoffish.com`.

[Furry 1942] W. Furry. Number of primes and probability considerations. *Nature*, 150:120–121, 1942.

[Gabcke 1979] W. Gabcke. *Neue Herleitung und explizite Restabschätzung der Riemann–Siegel Formel.* PhD thesis, Georg-August-Universität zu Göttingen, 1979.

[Gallot 1999] Y. Gallot, 1999. Private communication.

[Galway 1998] W. Galway, 1998. Private communication.

[Galway 2000] W. Galway. *Analytic computation of the prime-counting function.* PhD thesis, U. Illinois at Urbana-Champaign, 2000.

[Gardner 1977] M. Gardner. Mathematical games: a new kind of cipher that would take millions of years to break. *Scientific American*, August 1977.

[Gentleman and Sande 1966] W. Gentleman and G. Sande. Fast Fourier transforms—for fun and profit. In *Proc. AFIPS*, volume 29, pages 563–578, 1966.

[Goldwasser and Kilian 1986] S. Goldwasser and J. Kilian. Almost all primes can be quickly certified. In *Proc. 18th Annual ACM Symposium on the Theory of Computing*, pages 316–329, 1986.

[Goldwasser and Micali 1982] S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all mental information. In *Proc. 14th Annual ACM Symposium on the Theory of Computing*, pages 365–377, 1982.

[Golomb 1956] S. Golomb. Combinatorial proof of Fermat's 'little theorem'. *Amer. Math. Monthly*, 63, 1956.

[Golomb 1982] S. Golomb. *Shift Register Sequences,* (revised version). Aegean Park Press, 1982.

[Gong et al. 1999] G. Gong, T. Berson, and D. Stinson. Elliptic curve pseudorandom sequence generators. In *Proc. Sixth Annual Workshop on Selected Areas in Cryptography*, Kingston, Canada, August 1999.

[Gordon 1993] D. Gordon. Discrete logarithms in $GF(p)$ via the number field sieve. *SIAM J. Discrete Math.*, 16:124–138, 1993.

[Gordon and Pomerance 1991] D. Gordon and C. Pomerance. The distribution of Lucas and elliptic pseudoprimes. *Math. Comp.*, 57:825–838, 1991. Corrigendum *ibid.* 60:877, 1993.

[Gordon and Rodemich 1998] D. Gordon and G. Rodemich. Dense admissible sets. In [Buhler 1998], pages 216–225.

[Gourdon and Sebah 2004] X. Gourdon and P. Sebah. Numbers, constants and computation, 2004. `http://numbers.computation.free.fr/Constants/constants.html`.

[Graham and Kolesnik 1991] S. Graham and G. Kolesnik. *Van der Corput's method of exponential sums*, volume 126 of *Lecture Note Series*. Cambridge University Press, 1991.

[Grantham 1998] J. Grantham. A probable prime test with high confidence. *J. Number Theory*, 72:32–47, 1998.

[Grantham 2001] J. Grantham. Frobenius pseudoprimes. *Math. Comp.* 70:873–891, 2001.

[Granville 2004a] A. Granville. It is easy to determine if a given number is prime. *Bull. Amer. Math. Soc.*, 42:3–38, 2005.

[Granville 2004b] A. Granville. Smooth numbers: computational number theory and beyond. In J. Buhler and P. Stevenhagen, editors *Cornerstones in algorithmic number theory* (tentative title), a Mathematical Sciences Research Institute Publication. Cambridge University Press, to appear.

[Granville and Tucker 2002] A. Granville and T. Tucker. It's as easy as *abc*. *Notices Amer. Math. Soc.* 49:1224–1231, 2002.

[Green and Tao 2004] B. Green and T. Tao. The primes contain arbitrarily long arithmetic progressions. `http://arxiv.org/abs/math.NT/0404188`.

[Guy 1976] R. Guy. How to factor a number. In *Proceedings of the Fifth Manitoba Conference on Numerical Mathematics (Univ. Manitoba, Winnipeg, Man., 1975)*, volume 16 of *Congressus Numerantium*, pages 49–89, 1976.

[Guy 1994] R. Guy. *Unsolved Problems in Number Theory*. Second edition, volume I of *Problem Books in Mathematics. Unsolved Problems in Intuitive Mathematics*. Springer—Verlag, 1994.

[Hafner and McCurley 1989] J. Hafner and K. McCurley. A rigorous subexponential algorithm for computation of class groups. *J. Amer. Math. Soc.*, 2:837–850, 1989.

[Halberstam and Richert 1974] H. Halberstam and H.-E. Richert. *Sieve Methods*, volume 4 of *London Mathematical Society Monographs*. Academic Press, 1974.

[Hardy 1966] G. Hardy. *Collected Works of G. H. Hardy*, Vol. I. Clarendon Press, Oxford, 1966.

[Hardy and Wright 1979] G. Hardy and E. Wright. *An Introduction to the Theory of Numbers*. Fifth edition. Clarendon Press, Oxford, 1979.

[Harley 2002] R. Harley. Algorithmique avancée sur les courbes elliptiques. PhD thesis, University Paris 7, 2002.

[Håstad et al. 1999] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Computing*, 28:1364–1396, 1999.

[Hensley and Richards 1973] D. Hensley and I. Richards. Primes in intervals. *Acta Arith.*, 25:375–391, 1973/74.

[Hey 1999] T. Hey. Quantum computing. *Computing and Control Engineering*, 10(3):105–112, 1999.

[Higham 1996] N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 1996.

[Hildebrand 1988a] A. Hildebrand. On the constant in the Pólya–Vinogradov inequality. *Canad. Math. Bull.*, 31:347–352, 1988.

[Hildebrand 1988b] A. Hildebrand. Large values of character sums. *J. Number Theory*, 29:271–296, 1988.

[Honaker 1998] G. Honaker, 1998. Private communication.

[Hooley 1976] C. Hooley. *Applications of Sieve Methods to the Theory of Numbers*, volume 70 of *Cambridge Tracts in Mathematics*. Cambridge University Press, 1976.

[Ivić 1985] A. Ivić. *The Riemann Zeta-Function*. John Wiley and Sons, 1985.

[Izu et al. 1998] T. Izu, J. Kogure, M. Noro, and K. Yokoyama. Efficient implementation of Schoof's algorithm. In *Advances in Cryptology, Proc. Asiacrypt '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 66–79. Springer—Verlag, 1998.

[Jaeschke 1993] G. Jaeschke. On strong pseudoprimes to several bases. *Math. Comp.*, 61:915–926, 1993.

[Joe 1999] S. Joe. An average $L_2$ discrepancy for number-theoretic rules. *SIAM J. Numer. Anal.*, 36:1949–1961, 1999.

[Johnson et al. 2001] D. Johnson, A. Menezes, and S. Vanstone. The elliptic curve digital signature algorithm (ECDSA). *International Journal of Information Security*, 1:36–63, 2001.

[Jurišić and Menezes 1997] A. Jurišić and A. Menezes. Elliptic curves and cryptography. *Dr. Dobb's Journal*, pages 26–36, April 1997.

[Kaczorowski 1984] J. Kaczorowski. On sign changes in the remainder-term of the prime-number formula. I. *Acta Arith.*, 44:365–377, 1984.

[Kaliski 1988] B. Kaliski, Jr. *Elliptic Curves and Cryptography: a Pseudorandom Bit Generator and other Tools*. PhD thesis, Massachusetts Institute of Technology, 1988.

[Kaliski 1991] B. Kaliski, Jr. One-way permutations on elliptic curves. *J. Cryptology*, 3:187–199, 1991.

[Keller 1999] W. Keller. Prime factors $k.2^n+1$ of Fermat numbers $F_m$ and complete factoring status, 1999. `http://www.prothsearch.net/fermat.html`.

[Knuth 1971] D. Knuth. The analysis of algorithms. In *Actes du Congrès International des Mathématiciens (Nice 1970)*, Volume 3, pages 269–274. Gauthier-Villars, 1971.

[**Knuth 1981**] D. Knuth. *Seminumerical Algorithms* (Second edition), volume 2 of *The Art of Computer Programming*. Addison-Wesley, 1981.

[Knuth and Trabb Pardo 1976] D. Knuth and L. Trabb Pardo. Analysis of a simple factorization algorithm. *Theoret. Comput. Sci.*, 3:321–348, 1976-77.

[Koblitz 1987] N. Koblitz. Elliptic curve cryptosystems. *Math. Comp.*, 48:203–209, 1987.

[Koblitz 1994] N. Koblitz. *A Course in Number Theory and Cryptography*. Springer—Verlag, 1994.

[Koç et al. 1996] Ç. Koç, T. Acar, and B. Kaliski, Jr. Analyzing and comparing Montgomery multiplication algorithms. *IEEE Micro*, 16:26-33, 1996.

[Koç and Hung 1997] Ç. Koç and C. Hung. Fast algorithm for modular reduction. *IEEE Proc.: Computers and Digital Techniques*, 145(4), 1998.

[Kocis and White 1997] L. Kocis and W. Whiten. Computational investigations of low-discrepancy sequences. *ACM Trans. Math. Soft.*, 23:266–294, 1997.

[Konyagin and Pomerance 1997] S. Konyagin and C. Pomerance. On primes recognizable in deterministic polynomial time. In *The Mathematics of Paul Erdős, I*, volume 13 of *Algorithms and Combinatorics*, pages 176–198. Springer—Verlag, 1997.

[Korobov 1992] N. Korobov, *Exponential Sums and their Applications*, Kluwer Academic Publishers, 1992.

[Kuipers and Niederreiter 1974] L. Kuipers and H. Niederreiter. *Uniform Distribution of Sequences*. John Wiley and Sons, 1974.

[Kurlberg and Pomerance 2004] P. Kurlberg and C. Pomerance. On the periods of the linear congruential and power generators. Preprint, 2004.

[Lagarias 1990] J. Lagarias. Pseudorandom number generators in cryptography and number theory. In C. Pomerance, editor, *Cryptology and computational number theory*, volume 42 of *Proc. Sympos. Appl. Math.*, pages 115–143. Amer. Math. Soc., 1990.

[Lagarias 1999] J. Lagarias. On a positivity property of the Riemann $\xi$-function. *Acta Arith.*, 89:217–234, 1999.

[Lagarias et al. 1985] J. Lagarias, V. Miller, and A. Odlyzko. Computing $\pi(x)$: the Meissel-Lehmer method. *Math. Comp.*, 44:537–560, 1985.

[Lagarias and Odlyzko 1987] J. Lagarias and A. Odlyzko. Computing $\pi(x)$: an analytic method. *J. Algorithms*, 8:173–191, 1987.

[Languasco 2000] A. Languasco. Some refinements of error terms estimates for certain additive problems with primes. *J. Number Theory*, 81:149–161, 2000.

[Lavenier and Saouter 1998] D. Lavenier and Y. Saouter. Computing Goldbach Partitions Using Pseudo-random Bit Generator Operators on a FPGA Systolic Array. *Lecture Notes in Computer Science*, Springer–Verlag, 1482:316, 1998.

[L'Ecuyer and Simard 1999] P. L'Ecuyer and R. Simard. Beware of linear congruential generators with multipliers of the form $a = \pm 2^q \pm 2^r$. *ACM Trans. Math. Soft.*, 25:367–374, 1999.

[Lehman 1974] R. Lehman. Factoring large integers. *Math. Comp.*, 28:637–646, 1974.

[Lehmer 1964] E. Lehmer. On the infinitude of Fibonacci pseudo-primes. *Fibonacci Quart.*, 2:229–230, 1964.

[Lenstra 1983] A. Lenstra. Factoring polynomials over algebraic number fields. In *Computer algebra (London, 1983)*, volume 162 of *Lecture Notes in Computer Science*, pages 245–254. Springer—Verlag, 1983.

[Lenstra and Lenstra 1993] A. Lenstra and H. Lenstra, Jr., editors. *The development of the number field sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer—Verlag, 1993.

[Lenstra et al. 1982] A. Lenstra, H. Lenstra, Jr., and L. Lovasz. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[Lenstra et al. 1993a] A. Lenstra, H. Lenstra, Jr., M. Manasse, and J. Pollard. The factorization of the ninth Fermat number. *Math. Comp.*, 61:319–349, 1993.

[Lenstra and Manasse 1994] A. Lenstra and M. Manasse. Factoring with two large primes. *Math. Comp.*, 63:785–798, 1994.

[Lenstra 1981] H. Lenstra, Jr. Primality testing algorithms (after Adleman, Rumely and Williams). In *Seminar Bourbaki 33 (1980/81)*, volume 901 of *Lecture Notes in Mathematics*, exp. 576. Springer—Verlag, 1981.

[Lenstra 1984] H. Lenstra, Jr. Divisors in residue classes. *Math. Comp.*, 42:331–340, 1984.

[Lenstra 1985] H. Lenstra, Jr. Galois theory and primality testing. In *Orders and their applications (Oberwolfach, 1984)*, volume 1142 of *Lecture Notes in Mathematics*, pages 169–189. Springer–Verlag, 1985.

[Lenstra 1987] H. Lenstra, Jr. Factoring integers with elliptic curves. *Ann. of Math.*, 2:649–673, 1987.

[Lenstra 1991] H. Lenstra, Jr., 1991. Private communication.

[Lenstra et al. 1993b] H. Lenstra, Jr., J. Pila, and C. Pomerance. A hyperelliptic smoothness test. I. *Philos. Trans. Roy. Soc. London Ser. A*, 345:397–408, 1993. Special issue compiled and edited by R. Vaughan: Theory and applications of numbers without large prime factors.

[Lenstra and Pomerance 1992] H. Lenstra, Jr. and C. Pomerance. A rigorous time bound for factoring integers. *J. Amer. Math. Soc.*, 5:483–516, 1992.

[Lenstra and Pomerance 2005] H. Lenstra, Jr. and C. Pomerance. Primality testing with Gaussian periods. Preprint, 2005.

[Li 1997] X. Li. The positivity of a sequence of numbers and the Riemann hypothesis. *J. Number Theory*, 65:325–333, 1997.

[Lindqvist and Peetre 1997] P. Lindqvist and J. Peetre, On the remainder in a series of Mertens, *Exposition. Math.*, 15:467–478, 1997.

[Lim and Lee 1997] C. Lim and P. Lee. A key recovery attack on discrete log-based schemes using a prime order subgroup. In *Advances in Cryptology, Proc. Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 249–265. Springer–Verlag, 1997.

[Long 1981] D. Long. Random equivalence of factorization and computation of orders, 1981. Princeton U. Dept. Elec. Eng. and Comp. Sci. Technical Report 284.

[Lovorn 1992] R. Lovorn. *Rigorous, subexponential algorithms for discrete logarithms over finite fields.* PhD thesis, U. Georgia, 1992.

[Lovorn Bender and Pomerance 1998] R. Lovorn Bender and C. Pomerance. Rigorous discrete logarithm computations in finite fields via smooth polynomials. In J. Teitelbaum, editor, *Computational Perspectives on Number Theory (Chicago, IL, 1995)*, volume 7, pages 221–232. Atkin Conference, Amer. Math. Soc., 1998.

[Madisetti and Williams 1997] V. Madisetti and D. Williams, editors. *The Digital Signal Processing Handbook.* CRC Press, 1997.

[Marcus 1977] D. Marcus. *Number Fields.* Springer–Verlag, 1977.

[Marsaglia 1991] G. Marsaglia. The mathematics of random number generators. In S. Burr, editor, *The Unreasonable Effectiveness of Number Theory*, volume 46 of *Proc. Sympos. Appl. Math.*, pages 73–90. American Math. Soc., 1991.

**[Matijasevič 1971]** Y. Matijasevič. Diophantine representations of the set of prime numbers. *Dokl. Akad. Nauk SSSR*, 12:354–358, 1971.

[Mauldin 1999] R. Mauldin. The Beal conjecture and prize, 1999. `http://www.math.unt.edu/~mauldin/beal.html`.

[McClellan and Rader 1979] J. McClellan and C. Rader. *Number Theory in Digital Signal Processing.* Prentice-Hall, 1979.

[McKee 1996] J. McKee. Turning Euler's factoring method into a factoring algorithm. *Bull. London Math. Soc.*, 28:351–355, 1996.

[McKee 1999] J. McKee. Speeding Fermat's factoring method. *Math. Comp.*, 68:1729–1737, 1999.

[Menezes et al. 1993] A. Menezes, T. Okamoto, and S. Vanstone. Reducing elliptic curve logarithms to a finite field. *IEEE Trans. Inform. Theory*, 39:1639–1646, 1993.

[Menezes et al. 1997] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997.

[Mignotte 2001] M. Mignotte. Catalan's equation just before 2000. In M. Jutila and T. Metsänkylä, editors, *Number Theory (Turku, 1999).* de Gruyter, 247–254.

[Mihăilescu and Avanzi 2003] P. Mihăilescu and R. Avanzi. Efficient 'quasi-deterministic' primality test improving AKS. `http://www-math.uni-paderborn.de/~preda/`.

[Mihăilescu 2004] P. Mihăilescu. Primary cyclotomic units and a proof of Catalan's conjecture. *J. Reine Angew. Math.* 572:167–195, 2004.

[Miller 1976] G. Miller. Riemann's hypothesis and tests for primality. *J. Comput. System Sci.*, 13:300–317, 1976.

[Miller 1987] V. Miller. Use of elliptic curves in cryptography. In H. Williams, editor, *Advances in Cryptology, Proc. Crypto '85*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer–Verlag, 1987.

[Mills 1947] W. Mills. A prime-representing function. *Bull. Amer. Math. Soc.*, 53:604, 1947.

[Moenck 1973] R. Moenck. Fast computation of GCDs. In *Proc. 5th Annual ACM Symposium on the Theory of Computing*, pages 142–151, 1973.

[Monier 1980] L. Monier. Evaluation and comparison of two efficient probabilistic primality testing algorithms. *Theoret. Comput. Sci.*, 12:97–108, 1980.

[Montgomery and Vaughan 1973] H. Montgomery and R. Vaughan. The large sieve. *Mathematika* 20:119–134, 1973.

[Montgomery 1985] P. Montgomery. Modular multiplication without trial division. *Math. Comp.*, 44:519–521, 1985.

[Montgomery 1987] P. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Math. Comp.*, 48:243–264, 1987.

[Montgomery 1992a] P. Montgomery. *An FFT Extension of the Elliptic Curve Method of Factorization.* PhD thesis, University of California, Los Angeles, 1992.

[Montgomery 1992b] P. Montgomery. Evaluating recurrences of form $X_{m+n} = f(X_m, X_n, X_{m-n})$ via Lucas chains. Unpublished manuscript, 1992.

[Montgomery 1994] P. Montgomery. Square roots of products of algebraic numbers. In W. Gautschi, editor, *Mathematics of Computation 1943–1993*, volume 48 of *Proc. Sympos. Appl. Math.*, pages 567–571. Amer. Math. Soc., 1994.

[Montgomery 1995] P. Montgomery. A block Lanczos algorithm for finding dependencies over $GF(2)$. In *Advances in Cryptology, Eurocrypt '95*, volume 921 of *Lecture Notes in Computer Science*, pages 106–120, 1995.

[Montgomery and Silverman 1990] P. Montgomery and R. Silverman. An FFT extension to the $P - 1$ factoring algorithm. *Math. Comp.*, 54:839–854, 1990.

[Morain 1990] F. Morain. *Courbes elliptiques et tests de primalité.* PhD thesis, Université Claude Bernard-Lyon I, 1990.

[Morain 1992] F. Morain. Building cyclic elliptic curves modulo large primes. Unpublished manuscript, 1992.

[Morain 1995] F. Morain. Calcul du nombre de points sur une courbe elliptique dans un corps fini: aspects algorithmiques. *J. Théor. Nombres Bordeaux*, 7:255–282, 1995. Les Dix-huitèmes Journées Arithmétiques (Bordeaux, 1993).

[Morain 1998] F. Morain. Primality proving using elliptic curves: an update. In [Buhler 1998], pages 111 –127.

[Morain 2004] F. Morain. Implementing the asymptotically fast version of the elliptic curve primality proving algorithm. `http://www.lix.polytechnique.fr/Labo/Francois.Morain`.

[Morrison and Brillhart 1975] M. Morrison and J. Brillhart. A method of factoring and the factorization of $F_7$. *Math. Comp.*, 29:183–205, 1975. Collection of articles dedicated to Derrick Henry Lehmer on the occasion of his seventieth birthday.

[Müller 1998] V. Müller. Efficient algorithms for multiplication on elliptic curves. Proceedings of GI—Arbeitskonferenz Chipkarten, TU München, 1998.

[Müller 2004] V. Müller. Publications Volker Müller, 2004. `http://lecturer.ukdw.ac.id/vmueller/publications.php`, 2004.

[Murphy 1998] B. Murphy. Modelling the yield of number field sieve polynomials. In [Buhler 1998], pages 137–150.

[Murphy 1999] B. Murphy. *Polynomial selection for the number field sieve integer factorisation algorithm.* PhD thesis, Australian National University, 1999.

[Namba 1984] M. Namba. *Geometry of Projective Algebraic Curves*, volume 88 of *Monographs and Textbooks in Pure and Applied Mathematics.* Marcel Dekker, 1984.

[Narkiewicz 1986] W. Narkiewicz. *Classical Problems in Number Theory.* PWN-Polish Scientific Publishers, 1986.

[Nathanson 1996] M. Nathanson. *Additive Number Theory: The Classical Bases*, volume 164 of *Graduate Texts in Mathematics.* Springer–Verlag, 1996.

[Nguyen 1998] P. Nguyen. A Montgomery-like square root for the number field sieve. In [Buhler 1998], pages 151–168.

[Nguyen and Liu 1999] N. Nguyen and Q. Liu. The Regular Fourier Matrices and Nonuniform Fast Fourier Transforms. *SIAM J. Sci. Comput.*, 21:283–293, 1999.

[Nicely 2004] T. Nicely. Prime constellations research project, 2004. `http://www.trnicely.net/counts.html`.

[Niederreiter 1992] H. Niederreiter. *Random Number Generation and Quasi-Monte-Carlo Methods*, volume 63 of *CBMS-NSF Regional Conference Series in Applied Mathematics.* SIAM, 1992.

[Niven et al. 1991] I. Niven, H. Zuckerman, and H. Montgomery. *An Introduction to the Theory of Numbers.* Fifth edition. John Wiley and Sons, 1991.

[Nussbaumer 1981] H. Nussbaumer. *Fast Fourier Transform and Convolution Algorithms.* Springer–Verlag, 1981.

[Odlyzko 1985] A. Odlyzko. Discrete logarithms in finite fields and their cryptographic significance. In *Advances in Cryptology, Proc. Eurocrypt '84*, volume 209 of *Lecture Notes in Computer Science*, pages 224–313. Springer–Verlag, 1985.

[Odlyzko 1987] A. Odlyzko. On the distribution of spacings between zeros of the zeta function. *Math. Comp.*, 48:273–308, 1987.

[Odlyzko 1992] A. Odlyzko. The $10^{20}$-th zero of the Riemann zeta function and 175 million of its neighbors, 1992. `http://www.research.att.com/~amo`.

[Odlyzko 1994] A. Odlyzko. Analytic computations in number theory. In W. Gautschi, editor, *Mathematics of Computation 1943–1993*, volume 48 of *Proc. Sympos. Appl. Math.*, pages 441–463. Amer. Math. Soc., 1994.

[Odlyzko 2000] A. Odlyzko. Discrete logarithms: The past and the future. *Designs, Codes, and Cryptography*, 19:129–145, 2000.

[Odlyzko 2005] A. Odlyzko. The zeros of the Riemann zeta function: the $10^{22}$-nd zero and 10 billion of its neighbors. In preparation.

[Odlyzko and te Riele 1985] A. Odlyzko and H. te Riele. Disproof of the Mertens conjecture. *J. Reine Angew. Math.*, 357:138–160, 1985.

[Odlyzko and Schönhage 1988] A. Odlyzko and A. Schönhage. Fast algorithms for multiple evaluations of the Riemann zeta-function. *Trans. Amer. Math. Soc.*, 309:797–809, 1988.

[Oesterlé 1985] J. Oesterlé. Nombres de classes des corps quadratiques imaginaires. In *Seminar Bourbaki (1983/84)*, Astérisque No. 121-122, pages 309–323, 1985.

[Ohi 2003] H. Oki, 2003. Private communication.

[Okeya and Sakurai 2001] K. Okeya and K. Sakurai. Efficient Elliptic Curve Cryptosystems from a Scalar Multiplication Algorithm with Recovery of the $y$-Coordinate on a Montgomery-Form Elliptic Curve. In C. K. Koc,

D. Naccache, C. Paar (Eds.). Third International Workshop on Cryptographic Hardware and Embedded Systems—CHES 2001. LNCS 2162:126, Springer–Verlag, Paris, France, May 14-16, 2001.

[Owen 1995] A. Owen. Randomly permuted $(t, m, s)$-nets and $(t, m, s)$-sequences. In *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, volume 106 of *Lecture Notes in Statistics*, pages 299–317. Springer–Verlag, 1995.

[Owen 1997a] A. Owen. Monte Carlo variance of scrambled net quadrature. *SIAM J. Numer. Anal.*, 34:1884–1910, 1997.

[Owen 1997b] A. Owen. Scrambled net variance for integrals of smooth functions. *Ann. Statist.*, 25:1541–1562, 1997.

[Padma and Venkataraman 1996] R. Padma and S. Venkataraman. Elliptic curves with complex multiplication and a character sum. *J. Number Theory*, 61:274–282, 1996.

[Papadopoulos 1999] J. Papadopoulos, 1999. Private communication.

[Papageorgiu and Traub 1997] A. Papageorgiu and J. Traub. Faster evaluation of multidimensional integrals, 1997. `http://arxiv.org/find/physics/1/au:+Traub/0/1/0/2000/0/1`.

[Parberry 1970] E. Parberry. On primes and pseudo-primes related to the Fibonacci sequence. *Fibonacci Quart.*, 8:49–60, 1970.

[Park and Miller 1988] S. Park and K. Miller. Random number generators: good ones are hard to find. *Comm. ACM*, 31:1192–1201, 1988.

[Paskov and Traub 1995] S. Paskov and J. Traub. Faster valuation of financial derivatives. *J. Portfolio Management*, 22:113–120, 1995.

[Patel and Sundaram 1998] S. Patel and G. Sundaram. An efficient discrete log pseudo random generator. In H. Krawczyk, editor, *Advances in Cryptology, Proc. Crypto '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 304–317. Springer–Verlag, 1998.

[Paulos 1995] J. Paulos. High 5 jive. *Forbes*, 156:102, October 1995.

**[Paun et al. 1998]** G. Paun, G. Rozenberg, and A. Salomaa. *DNA Computing: New Computing Paradigms.* Springer–Verlag, 1998.

[Peralta 1993] R. Peralta. A quadratic sieve on the $n$-dimensional hypercube. In *Advances in Cryptology, Proc. Crypto '92*, volume 740 of *Lecture Notes in Computer Science.* Springer–Verlag, 1993.

[Peralta and Okamoto 1996] R. Peralta and E. Okamoto. Faster factoring of integers of a special form. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E79-A:489–493, 1996.

[Percival 2003] C. Percival. Rapid multiplication modulo the sum and difference of highly composite numbers. *Math. Comp.* 72:387–395, 2003.

[Peterson 2000] I. Peterson. Great computations. *Science News*, 157(10):152–153, March 4, 2000.

[Pinch 1993] R. Pinch. The Carmichael numbers up to $10^{15}$. *Math. Comp.*, 61:381–391, 1993.

[Pollard 1974] J. Pollard. Theorems on factorization and primality testing. *Proc. Cambridge Philos. Soc.*, 76:521–528, 1974.

[Pollard 1975] J. Pollard. A Monte Carlo method for factorization. *Nordisk Tidskr. Informationsbehandling (BIT)*, 15:331–334, 1975.

[Pollard 1978] J. Pollard. Monte Carlo methods for index computation (mod $p$). *Math. Comp.*, 32:918–924, 1978.

[Pollard 2000] J. Pollard. Kangaroos, Monopoly and discrete logarithms. *J. Cryptology*, 13:437–447, 2000.

[Pomerance 1981] C. Pomerance. On the distribution of pseudoprimes. *Math. Comp.*, 37:587–593, 1981.

[Pomerance 1982] C. Pomerance. Analysis and comparison of some integer factoring algorithms. In H. Lenstra, Jr. and R. Tijdeman, editors, *Computational methods in number theory, Part I*, volume 154 of *Math. Centre Tracts*, pages 89–139. Math. Centrum, 1982.

[Pomerance 1985] C. Pomerance. The quadratic sieve factoring algorithm. In *Advances in cryptology, Proc. Eurocrypt '84*, volume 209 of *Lecture Notes in Computer Science*, pages 169–182. Springer–Verlag, 1985.

[Pomerance 1986] C. Pomerance. On primitive divisors of Mersenne numbers. *Acta Arith.*, 46:355–367, 1986.

[Pomerance 1987a] C. Pomerance. Very short primality proofs. *Math. Comp.*, 48:315–322, 1987.

[Pomerance 1987b] C. Pomerance. Fast, rigorous factorization and discrete logarithm algorithms. In *Discrete Algorithms and Complexity*, pages 119–143. Academic Press, 1987.

[Pomerance 1996a] C. Pomerance. Multiplicative independence for random integers. In *Analytic Number Theory, Vol. 2 (Allerton Park, IL, 1995)*, volume 139 of *Progr. Math.*, pages 703–711. Birkhäuser, 1996.

[Pomerance 1996b] C. Pomerance. A tale of two sieves. *Notices Amer. Math. Soc.*, 43:1473–1485, 1996.

[Pomerance and Smith 1992] C. Pomerance and J. Smith. Reduction of huge, sparse matrices over finite fields via created catastrophes. *Experiment. Math.*, 1:89–94, 1992.

[Pomerance et al. 1988] C. Pomerance, J. Smith, and R. Tuler. A pipeline architecture for factoring large integers with the quadratic sieve algorithm. *SIAM J. Comput.*, 17:387–403, 1988. Special issue on cryptography.

**[Prachar 1978]** K. Prachar. *Primzahlverteilung*, volume 91 of *Grundlehren der Mathematischen Wissenschaften*. Springer–Verlag, 1978. Reprint of the 1957 edition.

[Pratt 1975] V. Pratt. Every prime has a succinct certificate. *SIAM J. Comput.*, 4:214–220, 1975.

[Preskill 1999] J. Preskill. Course notes, Phys 229, Calif. Inst. of Tech., 1999. `www.theory.caltech.edu/people/preskill/ph229/`.

[Press et al. 1996] W. Press, S. Teukolsky, W. Vettering, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1996.

[Pritchard 1981] P. Pritchard. A sublinear additive sieve for finding prime numbers. *Comm. ACM*, 24:18–23, 1981.

[Pritchard et al. 1995] P. Pritchard, A. Moran, and A. Thyssen. Twenty-two primes in arithmetic progression. *Math. Comp.*, 64:1337–1339, 1995.

[Purdom and Williams 1968] P. Purdom and J. Williams. Cycle length in a random function. *Trans. Amer. Math. Soc.* 133:547–551, 1968.

[Pustyl'nikov 1999] L. Pustyl'nikov. On a property of the classical zeta-function associated with the Riemann conjecture on zeros. *Russian Math. Surveys*, 54:162–163, 1999.

[Rabin 1976] M. Rabin. Probabilistic algorithms. In *Algorithms and Complexity (Proc. Sympos., Carnegie-Mellon Univ., Pittsburgh, PA, 1976)*, pages 21–39. Academic Press, 1976.

[Rabin 1980] M. Rabin. Probabilistic algorithm for testing primality. *J. Number Theory*, 12:128–138, 1980.

[Ramaré 1995] O. Ramaré. On Šnirel'man's constant. *Ann. Scuola Norm. Sup. Pisa Cl. Sci. (4)*, 22:645–706, 1995.

[Ramaré and Rumely 1996] O. Ramaré and R. Rumely. Primes in arithmetic progressions. *Math. Comp.*, 65:397–425, 1996.

[Ribenboim 1994] P. Ribenboim. *Catalan's Conjecture: Are 8 and 9 the Only Consecutive Powers?* Academic Press, 1994.

[Ribenboim 1996] P. Ribenboim. *The New Book of Prime Number Records.* Springer–Verlag, 1996.

[Richstein 2001] J. Richstein. Verifying the Goldbach conjecture up to $4 \cdot 10^{14}$. *Math. Comp.*, 70:1745–1749, 2001.

[Riesel and Göhl 1970] H. Riesel and G. Göhl. Some calculations related to Riemann's prime number formula. *Math. Comp.*, 24:969–983, 1970.

[Rishi et al. 1984] D. Rishi, J. Parnami, and A. Rajwade. Evaluation of a cubic character sum using the $\sqrt{-19}$ division points of the curve $Y^2 = X^3 - 2^3 \cdot 19X + 2 \cdot 19^2$. *J. Number Theory*, 19:184–194, 1984.

[Rivest et al. 1978] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Comm. ACM*, 21:120–126, 1978.

[Rose 1988] H. Rose. *A Course in Number Theory.* Clarendon Press, Oxford, 1988.

[Rosser 1939] J. Rosser. The $n$-th prime is greater than $n \log n$. *Proc. London Math. Soc.*, 45:21–44, 1939.

[Rosser and Schoenfeld 1962] J. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois J. Math.*, 6:64–94, 1962.

[Rotkiewicz 1973] A. Rotkiewicz. On the pseudoprimes with respect to the Lucas sequences. *Bull. Acad. Polon. Sci. Sér. Sci. Math. Astronom. Phys.*, 21:793–797, 1973.

[Rumely 1993] R. Rumely. Numerical computations concerning the ERH. *Math. Comp.*, 61:415–440, S17–S23, 1993.

[Ruzsa 1999] I. Ruzsa. Erdős and the integers. *J. Number Theory*, 79:115–163, 1999.

[Saouter 1998] Y. Saouter. Checking the odd Goldbach conjecture up to $10^{20}$. *Math. Comp.*, 67:863–866, 1998.

[Satoh and Araki 1998] T. Satoh and K. Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Comment. Math. Univ. St. Paul.*, 47:81–92, 1998. Errata, *ibid.* 48:1999, 211-213.

[Schinzel and Sierpiński 1958] A. Schinzel and W. Sierpiński. Sur certaines hypothèses concernant les nombres premiers. *Acta Arith.*, 4:185–208, 1958. Erratum, *ibid.* 5:259, 1958.

[Schirokauer et al. 1996] O. Schirokauer, D. Weber, and T. Denny. Discrete logarithms: the effectiveness of the index calculus method. In *Algorithmic Number Theory: Proc. ANTS II, Talence, France*, volume 1122 of *Lecture Notes in Computer Science*, pages 337–361. Springer–Verlag, 1996.

[Schmidt 1972] W. Schmidt. Irregularities of distribution. VII. *Acta Arith.*, 21:45–50, 1972.

[Schneier 1996] B. Schneier. *Applied Cryptography.* John Wiley and Sons, 1996.

[Schoenfeld 1976] L. Schoenfeld. Sharper bounds for the Chebyshev functions $\theta(x)$ and $\psi(x)$. II. *Math. Comp.*, 30:337–360, 1976. Corrigendum, *ibid.* 30:900, 1976.

[Schönhage 1971] A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Informatica*, 1:139–144, 1971.

[Schönhage 1982] A. Schönhage. Asymptotically fast algorithms for the numerical multiplication and division of polynomials with complex coefficients. In *Computer Algebra, EUROCAM '82, Marseille*, volume 144 of *Lecture Notes in Computer Science*, pages 3–15. Springer–Verlag, 1982.

[Schönhage and Strassen 1971] A. Schönhage and V. Strassen. Schnelle Multiplikation grosser Zahlen. *Computing (Arch. Elektron. Rechnen)*, 7:281–292, 1971.

[Schoof 1982] R. Schoof. Quadratic fields and factorization. In H. Lenstra, Jr. and R. Tijdeman, editors, *Computational methods in number theory, Part I*, volume 154 of *Math. Centre Tracts*, pages 235–286. Math. Centrum, 1982.

[Schoof 1985] R. Schoof. Elliptic curves over finite fields and the computation of square roots mod $p$. *Math. Comp.*, 44:483–494, 1985.

[Schoof 1995] R. Schoof. Counting points on elliptic curves over finite fields. *J. Théor. Nombres Bordeaux*, 7:219–254, 1995. Les Dix-huitèmes Journées Arithmétiques (Bordeaux, 1993).

[Schoof 2004] R. Schoof. Four primality proving algorithms. In J. Buhler and P. Stevenhagen, editors *Cornerstones in algorithmic number theory* (tentative title), a Mathematical Sciences Research Institute Publication. Cambridge University Press, to appear.

[Schroeder 1999] M. Schroeder. *Number Theory in Science and Communication*, volume 7 of *Springer Series in Information Sciences.* Springer–Verlag, 1999. Corrected printing of the third (1997) edition.

[Scott 1999] M. Scott, 1999. Private communication.

[Selfridge and Hurwitz 1964] J. Selfridge and A. Hurwitz. Fermat numbers and Mersenne numbers. *Math. Comp.*, 18:146–148, 1964.

[Semaev 1998] I. Semaev. Evaluation of discrete logarithms in a group of $p$-torsion points of an elliptic curve in characteristic $p$. *Math. Comp.*, 67:353–356, 1998.

[Seroussi et al. 1999] G. Seroussi, N. Smart, and I. Blake. *Elliptic Curves in Cryptography*, volume 265 of *London Math. Soc. Lecture Note Series*. Cambridge University Press, 1999.

[Shamir 1999] A. Shamir. Factoring large numbers with the TWINKLE device (extended abstract). In Ç. Koç and C. Paar, editors, *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES '99, Worcester, MA*, volume 1717 of *Lecture Notes in Computer Science*, pages 2–12. Springer–Verlag, 1999.

[Shanks 1971] D. Shanks. Class number, a theory of factorization, and genera. In *1969 Number Theory Institute, Stony Brook, N.Y.*, volume 20 of *Proc. Sympos. Pure Math.*, pages 415–440. Amer. Math. Soc., 1971.

[Shanks and Schmid 1966] D. Shanks and L. Schmid. Variations on a theorem of Landau. Part I. *Math. Comp.*, 20:551–569, 1966.

[Shlesinger 1986] M. Shlesinger. On the Riemann hypothesis: a fractal random walk approach. *Physica*, 138A:310–319, 1986.

[Shor 1994] P. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proc. 35th Annual Symp. Found. Comp. Sci.*, pages 124–134, 1994.

[Shor 1999] P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41:303–332, 1999.

[Shoup 1992] V. Shoup. Searching for primitive roots in finite fields. *Math. Comp.*, 58:369–380, 1992.

[Shoup 1995] V. Shoup. A new polynomial factorization algorithm and its implementation. *J. Symbolic Comput.*, 20:363–397, 1995.

[Silva 2005] T. Silva. Goldbach conjecture verification. `http://www.ieeta.pt/ tos/goldbach.html`, 2005.

[Silverman 1986] J. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer–Verlag, 1986.

[Silverman and Wagstaff 1993] R. Silverman and S. Wagstaff, Jr. A practical analysis of the elliptic curve factoring algorithm. *Math. Comp.*, 61:445–462, 1993.

[Sloan and Wozniakowski 1998] I. Sloan and H. Wozniakowski. When are quasi-Monte Carlo algorithms efficient for high dimensional integrals? *Complexity*, 14:1–33, 1998.

[Smart 1998] N. Smart. *The algorithmic resolution of Diophantine equations*, volume 41 of *London Mathematical Society Student Texts*. Cambridge University Press, 1998.

[Smart 1999] N. Smart. The discrete logarithm problem on elliptic curves of trace one. *J. Cryptology*, 12:193–196, 1999.

[Solinas 1998] J. Solinas. Standard specifications for public key cryptography. Annex A: Number-theoretic background. *IEEE P1363 Draft(s)*, 1998–2004. `http://grouper.ieee.org/groups/1363/`.

[Solinas 1999] J. Solinas. Generalized Mersenne numbers, 1999. `http://www.cacr.math.uwaterloo.ca/techreports/1999/corr99-39.ps`.

[Sorenson 1994] J. Sorenson. Two fast GCD algorithms. *J. Algorithms*, 16:110–144, 1994.

[Srinivasan 1995] A. Srinivasan. *Compuations of Class Numbers of Quadratic Fields.* PhD thesis, U. Georgia, 1995.

[Stehlé and Zimmermann 2004] D. Stehlé and P. Zimmermann. A Binary Recursive gcd Algorithm. `http://www.loria.fr/~stehle/BINARY.html` and `http://www.loria.fr/ stehle/downloads/antsgcd.pdf`.

[Stein 1967] J. Stein. Computational problems associated with Racah algebra. *J. Comp. Phys.*, 1:397–405, 1967.

[Strassen 1977] V. Strassen. Einige Resultate über Berechnungskomplexität. *Jber. Deutsch. Math.-Verein.*, 78:1–8, 1976/77.

[Stuart 1996] I. Stuart. The magic of seven: signifies creation, the sum of the spiritual three and the material four. *British Medical Journal*, 313(7072), December 21 1996.

[Sun and Sun 1992] Z.-H. Sun and Z.-W. Sun. Fibonacci numbers and Fermat's last theorem. *Acta Arith.*, 60:371–388, 1992.

[Swarztrauber 1987] P. Swarztrauber. Multiprocessor FFTs. *Parallel Computing*, 5:197–210, 1987.

[Tanner and Wagstaff 1987] J. Tanner and S. Wagstaff, Jr. New congruences for the Bernoulli numbers. *Math. Comp.*, 48:341–350, 1987.

[Tatuzawa 1952] T. Tatuzawa. On a theorem of Siegel. *Jap. J. Math.*, 21:163–178, 1951-1952.

[Teitelbaum 1998] J. Teitelbaum. Euclid's algorithm and the Lanczos method over finite fields. *Math. Comp.*, 67:1665–1678, 1998.

[Terr 2000] D. Terr. A modification of Shanks' baby-step giant-step algorithm. *Math. Comp.*, 69:767–773, 2000.

[Teske 1998] E. Teske. Speeding up Pollard's rho method for computing discrete logarithms. In [Buhler 1998], pages 541–554.

[Teske 2001] E. Teske. On random walks for Pollard's rho method. *Math. Comp.*, 70:809–825, 2001.

[Tezuka 1995] S. Tezuka. *Uniform Random Numbers: Theory and Practice.* Kluwer Academic Publishers, 1995.

[Thomas et al. 1986] J. Thomas, J. Keller, and G. Larsen. The calculation of multiplicative inverses over $GF(P)$ efficiently where $P$ is a Mersenne prime. *IEEE Trans. Comp.*, C-35:478–482, 1986.

[Titchmarsh and Heath-Brown 1986] E. Titchmarsh and D. Heath-Brown. *The Theory of the Riemann Zeta-function.* Oxford University Press, 1986.

[Trevisan and Carvalho 1993] V. Trevisan and J. Carvalho. The composite character of the twenty-second Fermat number. *J. Supercomputing*, 9:179–182, 1995.

[van de Lune et al. 1986] J. van de Lune, H. te Riele, and D. Winter. On the zeros of the Riemann zeta function in the critical strip. IV. *Math. Comp.*, 46:667–681, 1986.

[van der Corput 1922] J. van der Corput. Verscharfung der Abschätzungen beim Teilerproblem. *Math. Ann.*, 87:39–65, 1922.

[van der Pol 1947] B. van der Pol. An electro-mechanical investigation of the Riemann zeta function in the critical strip. *Bull. Amer. Math. Soc.*, 53, 1947.

[Van Loan 1992] C. Van Loan. *Computational Frameworks for the Fast Fourier Transform*, volume 10 of *Frontiers in Applied Mathematics*. SIAM, 1992.

[van Oorschot and Wiener 1999] P. van Oorschot and M. Wiener. Parallel collision search with cryptanalytic applications. *J. Cryptology*, 12:1–28, 1999.

[van Zyl and Hutchinson] B. van Zyl and D. Hutchinson. Riemann zeros, prime numbers, and fractal potentials. *Nonlinear Sciences Abstracts*, 2003. `http://arxiv.org/abs/nlin.CD/0304038`.

[Vaughan 1977] R. Vaughan. Sommes trigonométriques sur les nombres premiers. *C. R. Acad. Sci. Paris Sér. A-B*, 285:A981–A983, 1977.

[Vaughan 1989] R. Vaughan, A new iterative method in Waring's problem, *Acta Arith.*, 162:1–71, 1989.

[**Vaughan 1997**] R. Vaughan. *The Hardy–Littlewood Method*. Second edition, volume 125 of *Cambridge Tracts in Mathematics*. Cambridge University Press, 1997.

[Veach 1997] E. Veach. *Robust Monte Carlo methods for light transport simulation*. PhD thesis, Stanford University, 1997.

[Vehka 1979] T. Vehka. Explicit construction of an admissible set for the conjecture that sometimes $\pi(x+y) > \pi(x) + \pi(y)$. *Notices Amer. Math. Soc.*, 26, A-453, 1979.

[**Vinogradov 1985**] I. Vinogradov. *Ivan Matveevič Vinogradov: Selected Works*. Springer–Verlag, 1985. L. Faddeev, R. Gamkrelidze, A. Karacuba, K. Mardzhanishvili, and E. Miščenko, editors.

[**Vladimirov et al. 1994**] V. Vladimirov, I. Volovich, and E. Zelenov. *p-adic Analysis and Mathematical Physics*, volume 1 of *Series on Soviet and East European Mathematics*. World Scientific, 1994.

[von zur Gathen and Gerhard 1999] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, 1999.

[Wagstaff 1978] S. Wagstaff, Jr. The irregular primes to 125000. *Math. Comp.*, 32:583–591, 1978.

[Wagstaff 1993] S. Wagstaff, Jr. Computing Euclid's primes. *Bull. Inst. Combin. Appl.*, 8:23–32, 1993.

[Wagstaff 2004] S. Wagstaff, Jr. The Cunningham project. `http://www.cerias.purdue.edu/homes/ssw/cun/index.html`.

[Ware 1998] A. Ware. Fast Approximate Fourier Transforms for Irregularly Spaced Data. *SIAM Rev.*, 40:838–856, 1998.

[Warren 1995] B. Warren. An interesting group of combination-product sets produces some very nice dissonances. *The Journal of the Just Intonation Network*, 9(1):1, 4–9, 1995.

[Watkins 2004] M. Watkins. Class numbers of imaginary quadratic fields. *Math. Comp.* 73:907–938, 2004.

[Watt 1989] N. Watt. Exponential sums and the Riemann zeta-function. II. *J. London Math. Soc.*, 39, 1989.

[Weber 1995] K. Weber. The accelerated GCD algorithm. *ACM Trans. Math. Soft.*, 21:111–122, 1995.

[Weber et al. 2005] K. Weber, V. Trevisan, and L. Martins. A modular integer GCD algorithm. *Journal of Algorithms*, 54:152–167, 2005.

[Wedeniwski 2004] S. Wedeniwski. Zetagrid, 2004. `http://www.zetagrid.net`.

[Weiss 1963] E. Weiss. *Algebraic Number Theory.* McGraw-Hill, 1963.

[Weisstein 2005] E. Weisstein. Mathworld, 2005. `http://www.mathworld.wolfram.com`.

[Wellin 1998] P. Wellin, 1998. Private communication.

[Weyl 1916] H. Weyl. Über die Gleichverteilung von Zahlen mod. Eins. *Math. Ann.*, 77, 1916.

[Wiedemann 1986] D. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inform. Theory*, 32:54–62, 1986.

[Wieferich 1909] A. Wieferich. Zum letzten Fermat'schen Theorem. *J. Reine Angew. Math.*, 136:293–302, 1909.

[Wiener 1990] M. Wiener. Cryptanalysis of short RSA secret exponents. *IEEE Trans. Inform. Theory*, 36:553–558, 1990.

[Williams 1998] H. Williams. *Édouard Lucas and Primality Testing*, volume 22 of *Canadian Mathematical Society Series of Monographs and Advanced Texts.* John Wiley and Sons, 1998.

[Williams and Clearwater 1998] C. Williams and S. Clearwater. *Explorations in Quantum Computing.* TELOS/Springer–Verlag, 1998.

[Williams and Shallit 1993] H. Williams and J. Shallit. Factoring integers before computers. In W. Gautschi, editor, *Mathematics of Computation 1943–1993*, volume 48 of *Proc. Sympos. Appl. Math.*, pages 481–531. Amer. Math. Soc., 1994.

[Winterhof 1998] A. Winterhof, On Waring's problem in finite fields, *Acta Arith.*, 87:171–177, 1998.

[Wolf 1997] M. Wolf. $1/f$ noise in the distribution of prime numbers. *Physica A*, 241:493–499, 1997.

[Woltman 2000] G. Woltman. Great Internet Mersenne prime search (GIMPS), 2000. `http://www.mersenne.org`.

[Wozniakowski 1991] H. Wozniakowski. Average case complexity of multivariate integration. *Bull. Amer. Math. Soc. (N.S.)*, 24:185–194, 1991.

[Wu 1997] P. Wu. Multiplicative, congruential random-number generators. *ACM Trans. Math. Soft.*, 23:255–265, 1997.

[Yacobi 1999] Y. Yacobi. Fast exponentiation using data compression. *SIAM J. Comput.*, 28:700–703, 1999.

[Yagle 1995] A. Yagle. Fast algorithms for matrix multiplication using pseudo-number-theoretic transforms. *IEEE Trans. Sig. Proc*, 43:71–76, 1995.

[Yan et al. 1991] J. Yan, A. Yan, and B. Yan. Prime numbers and the amino acid code: analogy in coding properties. *J. Theor. Biol.*, 151(3):333–341, 1991.

[Yoshimura 1997] J. Yoshimura. The evolutionary origins of periodical cicadas during ice ages. *American Naturalist*, 149(1):112–124, 1997.

[Yu 1996] G. Yu. The differences between consecutive primes. *Bull. London Math. Soc.*, 28:242–248, 1996.

[Zhang 1998] M. Zhang. Factorization of the numbers of the form $m^3 + c_2m^2 + c_1m + c_0$. In [Buhler 1998], pages 131–136.

[Zhang and Tang 2003] Z. Zhang and M. Tang. Finding strong pseudoprimes to several bases. II. *Math. Comp.*, 72: 2085–2097, 2003.

[Zhang 2002] Z. Zhang. A one-parameter quadratic-base version of the Baillie–PSW probable prime test. *Math. Comp.*, 71: 1699–1734, 2002.

[Zimmermann 2000] P. Zimmermann. The ECMNET project, 2000. `http://www.loria.fr/~zimmerma/records/ecmnet.html`.

[Zimmermann 2004] P. Zimmermann. Private communication.

[Zinoviev 1997] D. Zinoviev. On Vinogradov's constant in Goldbach's ternary problem. *J. Number Theory*, 65:334–358, 1997.

## Работы на русском языке

Здесь приведены работы из списка литературы, имеющиеся на русском языке.

[Aho et al. 1974] *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов: Пер. с англ. — М.: Мир, 1979.

[Knuth 1981] *Кнут Д.* Искусство программирования для ЭВМ. Т. 2. Получисленные алгоритмы: Пер. с 1-го англ. изд. — М.: Мир, 1976.

[Matijasevič 1971] *Матиясевич Ю. В.* Диофантовы представления множества простых чисел. ДАН СССР. 1971. **12.** 354–358.

[Paun et al. 1998] *Паун Г., Розенберг Г., Саломаа А.* ДНК-компьютер. Новая парадигма вычислений: Пер. с англ. — М.: Мир, 2004.

[Prachar 1978] *Прахар К.* Распределение простых чисел: Пер. с нем. А. А. Карацубы под ред. А. И. Виноградова. — М.: Мир, 1967.

[Vaughan 1997] *Вон Р.* Метод Харди—Литтлвуда: Пер. с 1-го англ. изд. — М.: Мир, 1985.

[Vinogradov 1985] *Виноградов И. М.* Избранные труды. — М.: Изд-во АН СССР, 1952.

[Vladimirov et al. 1994] *Владимиров В. С., Волович И. В., Зеленов Е. И.* p-адический анализ и математическая физика. — М.: Наука, 1994.

# Список литературы, добавленной при переводе

Сюда включены работы, полезные читателям для дальнейшего чтения.


[Bach and Charles 2005] E. Bach and D. Charles. The hardness of computing an eigenform. Unpublished manuscript, 2005.

[Charles 2005] D. Charles. *Computational aspects of modular forms and elliptic curves.* PhD thesis, University of Wisconsin-Madison, 2005.

[Chen 1973] J.-R. Chen. On the representation of a large even integer as the sum of a prime and the product of at most two primes. *Sci. Sinica*, 16:157–176, 1973.

[Dorofeev et al. 2007] A. Ya. Dorofeev, D. M. Dygin, and D. V. Matyukhin. Discrete logarithms in $GF(p)$ — 135 digits. In *Diophantine and analytic problems in number theory (Abstracts) (Moscow, Russia, January 29–February 2, 2007)*, pages 14–15.

[Edixhoven et al. 2006] B. Edixhoven, J.-M. Couveignes, R. de Jong, F. Merkl, and J. Bosman. On the computation of coefficients of a modular form, 2006. `http://www.arxiv.org/abs/math.NT/0605244`.

[Erdös 1935] P. Erdös. On the difference of consecutive primes. *Quart. J. Oxford*, 6:124–128, 1935.

[Erdös 1950] P. Erdös. On integers of the form $2^k + p$. *Summa Brasil. Math.*, 2:113–123, 1950.

[Gelfond 1968] A. O. Gelfond. Sur les nombres qui out des propriétés additives et multiplicatives données. *Acta Arith.*, 13:259–265, 1968.

[Goldston et al. 2005] D. A. Goldston, J. Pintz, and C. Y. Yildirim. Primes in Tuples I, 2005. `http://arxiv.org/abs/math/0508185v1`.

[Hagis 1973] P. Hagis, Jr. A lower bound for the set of odd perfect numbers. *Math. Comp.*, 27:951–953, 1973.

[Hagis and McDaniel 1975] P. Hagis, Jr. and W. L. McDaniel. On the largest prime divisor of an odd perfect number II. *Math. Comp.*, 29:922–924, 1975.

[Kilford 2008] L. J. P. Kilford. Modular forms: a classical and computational introduction. Imperial College Press, London, 2008.

[McCurley 1988] K. S. McCurley. A key distribution system equivalent to factoring. *J. Cryptology*, 1(2):95–105, 1988.

[Nagell 1951] T. Nagell. Introduction to number theory. John Wiley and Sons, 1951.

[Rankin 1938] R. A. Rankin. On the difference between consecutive prime numbers. *J. London Math. Soc.*, 13:242–247, 1938.

[Rankin 1940] R. A. Rankin. On the difference between consecutive prime numbers II. *Proc. Cambridge Phil. Soc.*, 36:255–256, 1940.

[Romanoff 1934] N. P. Romanoff. Über einige Satze der additiven Zahlentheorie. *Math. Ann.*, 109:668–678, 1934.

[Shmuely 1985] Z. Shmuely. Composite Diffie—Hellman public-key generating systems are hard to break, Febr. 1985. IIT Comp. Sci. Dept. Tech. Technical Report 356.

[Slowinski 1979] D. Slowinski. Searching for the 27th Mersenne prime. *J. Recreational Mathematics*, 11:258–261, 1979.

[van der Corput 1950] J. G. van der Corput. On de Polignac's conjecture. *Simon Stevin*, 27:99–105, 1950.

[FIPS] FIPS 186–2. Digital signature standard. `http://csrc.nist.gov/publications/`.

[Айерлэнд, Роузен 1967] *Айерлэнд К., Роузен М.* Классическое введение в современную теорию чисел. — М.: Мир, 1967.

[Анохин и др. 1997] *Анохин М. И., Варновский Н. П., Сидельников В. М., Ященко В. В.* Криптография в банковском деле. — М.: МИФИ, 1997. — 274 с.

[Архипов и др. 2004] *Архипов Г. И., Садовничий В. А., Чубариков В. Н.* Лекции по математическому анализу. — М.: Дрофа, 2004.

[Боревич, Шафаревич 1985] *Боревич З. И., Шафаревич И. Р.* Теория чисел. — М.: Наука, 1985.

[Бюлер 1989] *Бюлер В.* Гаусс. — М.: Наука, 1989.

[Василенко 2003] *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии. — М.: МЦНМО, 2003.

[Венков 1937] *Венков Б. А.* Элементарная теория чисел. — М.—Л.: ОНТИ, 1937.

[Виноградов 1937] *Виноградов И. М.* Представление нечетного числа суммою трех простых слагаемых // ДАН СССР. 1937. **15.** 291–294.

[Виноградов 1982] *Виноградов И. М.* Основы теории чисел. — М.: Наука, 1982.

[Виноградов 1999] *Виноградов И. М.* Элементы высшей математики (Аналитическая геометрия. Дифференциальное исчисление. Основы теории чисел). Учеб. для вузов. — М.: Высшая школа, 1999.

[Гаусс 1959] *Гаусс К. Ф.* Труды по теории чисел. — М.: Изд-во АН СССР, 1959.

[Гашков, Чубариков 1999] *Гашков С. Б., Чубариков В. Н.* Арифметика. Алгоритмы. Сложность вычислений. — М.: Высшая школа, 1999.

[Гекке 1940] *Гекке Э.* Лекции по теории алгебраических чисел. — М.—Л.: ГИТТЛ, 1940.

[Давенпорт 1971] *Давенпорт Г.* Мультипликативная теория чисел. — М.: Наука, 1971.

[Делоне 1947] *Делоне Б. Н.* Петербургская школа теории чисел. — М.—Л.: Изд-во АН СССР, 1947.

[Дирихле 1936] *Дирихле Лежен П. Г.* Лекции по теории чисел. — М.—Л.: ОНТИ, 1936; 3-е изд. М.: Книжный дом «Либроком»/URSS, 2009.

[Егоров 1923] *Егоров Д. Ф.* Элементы теории чисел. — М.—П.: ГИ, 1923.

[Ингам 1936] *Ингам А. Е.* Распределение простых чисел. — М.—Л.: ОНТИ, 1936; 4-е изд. М.: Книжный дом «Либроком»/URSS, 2009.

[Карацуба 1983] *Карацуба А. А.* Основы аналитической теории чисел. — М.: Наука, 1983; 2-е изд. М.: URSS, 2004.

[Касселс 1961] *Касселс Дж. В. С.* Введение в теорию диофантовых приближений. — М.: ИЛ, 1961.

[Касселс 1965] *Касселс Дж. В. С.* Введение в геометрию чисел. — М.: Мир, 1965.

[Коблиц 2001] *Коблиц Н.* Курс теории чисел и криптографии. — М.: Научное изд-во ТВП, 2001. — 254 с.

[Нестеренко 1998] *Нестеренко Ю. В.* Алгоритмические проблемы теории чисел // Математическое просвещение. Сер. 3, №2, 1998. 87–114.

[Нечаев 1999] *Нечаев В. И.* Элементы криптографии (основы теории защиты информации): учеб. пособие для ун-тов и пед. вузов. — М.: Высшая школа, 1999. — 109 с.

[Нильсон, Чанг 2006] *Нильсон М., Чанг И. И.* Квантовые вычисления и квантовая информация: Пер. с англ. — М.: Мир, 2006.

[Постников 1971] *Постников А. Г.* Введение в аналитическую теорию чисел. — М.: Наука, 1971.

[Постникова 1973] *Постникова Л. П.* Тригонометрические суммы и теория сравнений по простому модулю. — М.: МГПИ, 1973.

[Рябко, Фионов 2004] *Рябко Б. Я., Фионов А. Н.* Основы современной криптографии для специалистов в информационных технологиях. — М.: Научный мир, 2004. — 173 с.

[Трост 1959] *Трост Э.* Простые числа. — М.: Физматлит, 1959.

[Хассе 1953] *Хассе Г.* Лекции по теории чисел. — М.: ИЛ, 1953.

[Хооли 1983] *Хооли К.* Применения методов решета в теории чисел. — М.: Наука, 1983.

[Чандрасекхаран 1975] *Чандрасекхаран К.* Арифметические функции. — М.: Наука, 1975.

[Чебышёв 1944] *Чебышёв П. Л.* Полное собрание сочинений, т. 1. — М.—Л.: Изд-во АН СССР, 1944.

[Чудаков 1947] *Чудаков Н. Г.* Введение в теорию $L$-функций Дирихле. — М.—Л.: ГИТТЛ, 1947.

[Шнирельман 1940] *Шнирельман Л. Г.* Простые числа. — М.—Л.: ГИТТЛ, 1940.

[ГОСТ] ГОСТ Р34.11–94. Информационная технология. Криптографическая защита информации. Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма.

# Именной указатель

# Предметный указатель

# Оглавление

## Ричард Крэндалл

В настоящее время работает в компании Apple в звании выдающегося ученого. Ранее занимал должности главного криптографа в компании Apple, главного специалиста в компании NeXT, Inc., возглавлял кафедру имени Воллюма в Рид-колледже. Основная область интересов — междисциплинарные научные вычисления. Является автором многочисленных теоретических работ по квантовой физике, биологии, математике и химии, обладателем различных патентов в инженерных областях.

## Карл Померанс

Профессор математики в Дартмут-колледже. Защитил диссертацию по математике в Гарвардском университете в 1972 г. Долгое время работал в университете штата Джорджия; заслуженный профессор этого университета. Работал в техническом департаменте компании Bell Labs—Lucent Technologies. Карл Померанс — популярный лектор и обладатель многочисленных премий, в том числе премии Шовене и Конанта за обзорные математические работы. Он широко известен своими исследованиями по вычислительной теории чисел, а также по созданию важнейших алгоритмов, активно используемых в настоящее время.