Институт проблем информационной
безопасности МГУ

О. Н. ВАСИЛЕНКО

ТЕОРЕТИКО-ЧИСЛОВЫЕ АЛГОРИТМЫ В КРИПТОГРАФИИ

МЦНМО, 2003

УДК 511+519.719.2
ББК 32.81в6
В19

Издание осуществлено при поддержке РФФИ (издательский проект № 03-01-14110).

Василенко О. Н.

В19 Теоретико-числовые алгоритмы в криптографии. — М.: МЦНМО, 2003. — 328 с.

ISBN 5-94057-103-4

В монографии представлено современное состояние алгоритмической теории чисел, имеющей важные приложения в криптографии.

Предназначено для студентов старших курсов и аспирантов математических факультетов вузов, а также для специалистов, желающих познакомиться с последними достижениями в данной области.

ББК 32.81в6

ISBN 5-94057-103-4

Оглавление

Предисловие

Данная книга посвящена алгоритмической теории чисел — интенсивно развивающемуся последние тридцать лет направлению в теории чисел, имеющему важные приложения в криптографии. Актуальность этого направления неизмеримо увеличилась в 70-е годы XX века в связи с появлением криптосистем Диффи—Хеллмана и RSA. В настоящее время, по некоторым оценкам, практически весь мировой парк средств асимметричной криптографии в математическом плане основан на теоретико-числовых задачах.

Для целей криптографии (как для практической реализации и обоснования стойкости криптографических средств, так и для разработки методов их вскрытия) необходимо в первую очередь повышать эффективность следующих методов и алгоритмов:

- алгоритмы проверки простоты целых чисел;
- методы факторизации (т. е. методы поиска разложения целых чисел на множители);
- вычисления, использующие эллиптические кривые над конечными полями;
- алгоритмы дискретного логарифмирования;
- методы разложения многочленов на множители над конечными полями и над полем рациональных чисел;
- способы решения систем линейных уравнений над конечными полями;
- алгоритмы для выполнения арифметических операций с большими целыми числами;
- алгоритмы полиномиальной арифметики.

В нашей книге мы стремились представить современное состояние алгоритмической теории чисел, стараясь при этом дать достаточно аккуратные обоснования описываемых алгоритмов. Некоторые наиболее сложные и объемные алгоритмы и методы (такие, как проверка простоты чисел с помощью тригонометрических сумм, алгоритмы решета числового поля) мы приводим лишь на идейном уровне, в виде некоторых общих схем. Мы также старались (хотя бы обзорно) охватить современные работы по данной тематике как можно более

полно. Заинтересованный читатель может пополнить наш список литературы, обратившись к книгам [60; 101], а также к интернет-сайтам `www.cryptography.ru`; `www.math.uga.edu/~ntheory`.

Данная книга написана на основе спецкурсов по алгоритмической теории чисел, которые автор читал на механико-математическом факультете МГУ им. М. В. Ломоносова с 1993 по 2001 год. Кроме того, в течение ряда лет автор руководил спецсеминарами по теоретико-числовым алгоритмам в МГУ, последние годы — межведомственным спецсеминаром «Теоретико-числовые алгоритмы в криптографии» на кафедре информационной безопасности МГУ (совместно с академиком Академии криптографии Российской Федерации В. М. Сидельниковым). Ряд результатов, приведенных в данной книге, был получен автором в сотрудничестве с лабораторией по математическим проблемам криптографии МГУ им. М. В. Ломоносова.

Эта книга не является книгой по элементарной математике. Для ее чтения требуется достаточно серьезная подготовка, например, в объеме первых двух-трех курсов математического факультета университета.

Мы предполагаем, что читатель знаком с теорией чисел в объеме книги И. М. Виноградова «Основы теории чисел» (любое издание). Для чтения некоторых параграфов требуется знание теории непрерывных (цепных) дробей, см., например, книги [41; 29]. Для удобства читателя мы поместили основные определения и факты в Приложение. Для чтения некоторых параграфов требуется знание основ теории конечных полей, изложенной, например, в книге [31], а также знание основ алгебраической теории чисел, см., например, книгу [9]. В некоторых параграфах требуется более глубокое знание алгебраической теории чисел; в таких местах мы даем ссылки на соответствующую литературу. Из учебных пособий по криптографии мы рекомендуем книгу [3].

Во многих описываемых алгоритмах в качестве вспомогательных используются алгоритмы нахождения наибольшего общего делителя целых чисел и алгоритмы возведения в натуральную степень. Эти алгоритмы общеизвестны и изложены в различных книгах, см., например, [25; 89; 60]. Мы приводим эти алгоритмы в Приложении, некоторые без обоснования корректности.

Везде в книге, когда речь заходит о том, что алгоритм делает какое-либо количество арифметических операций, имеются в виду арифметические операции (сложение, вычитание, умножение, деление) с большими целыми числами (арифметика многократной точности).

Сложность алгоритма — это количество выполняемых им арифметических операций. Обычно сложность алгоритма представляют

в виде какой-либо функции от *длины входа*, т. е. от количества битов N, требуемых для записи входных данных. Если эта функция — многочлен от N, то говорят, что алгоритм имеет *полиномиальную сложность* (*полиномиальный* алгоритм); если эта функция имеет вид $L_N[\gamma; c] = e^{(c+o(1))(\log N)^{\gamma}(\log\log N)^{1-\gamma}}$, где $0 < \gamma < 1$, $c = \text{const}$, $c > 0$, то оценка сложности алгоритма называется *субэкспоненциальной* с показателем γ; если функция имеет вид e^{cN}, где $c = \text{const}$, то алгоритм имеет *экспоненциальную* сложность. Пусть, например, $n \in \mathbb{N}$ и мы хотим разложить n на множители. Длина входа равна $N = [\log_2 n] + 1 = O(\log n)$. Полиномиальный алгоритм факторизации имеет поэтому сложность $O((\log n)^c)$, субэкспоненциальный — сложность $e^{(c+o(1))(\log n)^{\gamma}(\log\log n)^{1-\gamma}}$, экспоненциальный — сложность $O(n^c)$.

Мы называем число *B-гладким*, если все его простые делители не превосходят B (здесь B — некоторое положительное число, называемое *границей гладкости*). Целое число называется *B-степенно-гладким*, если все его примарные делители (степени простых чисел) не превосходят B.

Через $\log x$ мы обозначаем натуральный логарифм положительного числа x.

Автор благодарит А. А. Сальникова, В. В. Ященко и Д. В. Матюхина за помощь в работе над книгой, неоднократные обсуждения и многочисленные советы по улучшению содержания рукописи.

Автор также благодарит Д. В. Матюхина за огромную работу по научному редактированию рукописи.

Обозначения

$\mathbb{N}$	множество натуральных чисел;
$\mathbb{Z}$	кольцо целых чисел;
$\mathbb{Z}_{\geqslant a}$	множество целых чисел, больших или равных a;
$\mathbb{R}$	поле действительных чисел;
$\mathbb{R}_{\geqslant a}$	множество действительных чисел, больших или равных a;
$\mathbb{C}$	поле комплексных чисел;
$\mathbb{P}$	множество простых чисел;
$\lvert S\rvert$, $\#S$	количество элементов в множестве S;
$\operatorname{Re}\alpha$	действительная часть числа α;
$\operatorname{Im}\alpha$	мнимая часть числа α;
$a \mid b$	a делит b;
$a \nmid b$	a не делит b;
$p^k \| a$	p^k делит a, но p^{k+1} не делит a;
$b \vdots a$	b делится на a (нацело);
$\nu_a(b)$	наибольшее $k \in \mathbb{Z}_{\geqslant 0}$ такое, что $a^k \mid b$;
(a, b), НОД(a, b)	наибольший общий делитель a и b, где a и b — целые числа или многочлены над некоторым полем;
$[a, b]$, НОК(a, b)	наименьшее общее кратное a и b;
$[a]$	целая часть числа a;
$\lceil a \rceil$	целая часть сверху, т. е. наименьшее $n \in \mathbb{Z}$, для которого $n \geqslant a$;
$\{a\}$	дробная часть числа a;
const	какая-либо положительная постоянная;
$n \asymp 10^k$	натуральное число n записывается k десятичными цифрами;
$a \equiv b \pmod{c}$	$a - b$ делится на c в рассматриваемом кольце (целых чисел или многочленов);
$a \not\equiv b \pmod{c}$	$a - b$ не делится на c;

$\mathbb{Z}/p\mathbb{Z}$, $GF(p)$, $\mathbb{Z}_p$	поле из p элементов, где p — простое число;
$GF(q)$	поле из q элементов, где q — степень простого числа;
$\mathbb{Z}/n\mathbb{Z}$	кольцо вычетов по модулю n;
$(\mathbb{Z}/n\mathbb{Z})^*$	группа обратимых по умножению элементов кольца $\mathbb{Z}/n\mathbb{Z}$;
R^*	группа обратимых по умножению элементов кольца R;
$\langle g\rangle_n$	циклическая группа из n элементов с образующим g;
$\operatorname{ord} a$	порядок элемента a в конечной группе;
$\operatorname{char}\mathbb{K}$	характеристика поля $\mathbb{K}$;
$\varphi(n)$	функция Эйлера от натурального числа n;
$\left(\frac{a}{p}\right)$, $\left(\frac{a}{m}\right)$	символы Лежандра и Якоби;
$\pi(x)$	функция Чебышёва, равная количеству простых чисел, не превосходящих x;
$\log x$	натуральный логарифм x;
$\binom{i}{j}$	биномиальный коэффициент, число сочетаний из i по j;
$L_x[\gamma;\, c]$	$e^{(c+o(1))(\log x)^{\gamma}(\log\log x)^{1-\gamma}}$, $o(1)\to 0$ при $x\to+\infty$, c и γ — постоянные;
M^T	транспонированная матрица (или вектор) M;
$\operatorname{rank} M$	ранг матрицы M;
$L_{\text{об}}(b_1, \ldots, b_n)$	линейная оболочка векторов $b_1, \ldots, b_n$;
$L^{\perp}$	ортогональное дополнение к линейному подпространству L в евклидовом пространстве;
$\mathbb{K}[x_1, \ldots, x_n]$	кольцо многочленов от переменных $x_1, \ldots, x_n$ над кольцом $\mathbb{K}$;
$\deg f(x)$	степень многочлена $f(x)$;
$\operatorname{Res}(f(x), g(x))$	результант многочленов $f(x)$ и $g(x)$;
$y := x$	y присвоить значение x.

Глава 1. Тестирование чисел на простоту и построение больших простых чисел

§ 1.1. Введение

Натуральное число p, большее единицы, называется простым, если оно делится нацело только на 1 и на себя. Основная теорема арифметики гласит, что любое натуральное число n, большее единицы, может быть разложено в произведение простых чисел, причем это разложение единственно с точностью до порядка простых сомножителей. Каноническое разложение натурального числа n имеет вид

$$n = p_1^{\alpha_1} \cdot \ldots \cdot p_k^{\alpha_k},$$

где $p_1 < p_2 < \ldots < p_k$ — различные простые числа, $\alpha_1, \ldots, \alpha_k \in \mathbb{N}$.

Задача проверки простоты натурального числа и задача построения больших простых чисел имеют важные приложения в криптографии. В данной главе мы опишем различные алгоритмы решения этих задач.

§ 1.2. Элементарные методы проверки простоты чисел

Пусть $n \in \mathbb{N}$. Как проверить, является ли n простым?

Метод пробных делений

Если n — составное, то $n = ab$, где $1 < a \leqslant b$, причем $a \leqslant \sqrt{n}$. Поэтому для $d = 2, 3, \ldots, [\sqrt{n}]$ мы проверяем, делится ли n на d? Если делитель числа n не будет найден, то n — простое. В противном случае будет найден минимальный простой делитель числа n, т. е. мы даже разложим n на два множителя. Сложность метода составляет $O(n^{1/2})$ арифметических операций с целыми числами.

Возможны модификации этого метода. Например, мы можем проверить, делится ли n на 2 и на 3, и если нет, то перебираем далее только числа d вида $1 + 6j$ и $5 + 6j$, $j = 1, 2, \ldots$ Сложность метода отличается от предыдущей лишь постоянной в $O(\cdot)$ (см. об этом также [40]).

Решето Эратосфена

Если мы хотим составить таблицу всех простых чисел среди чисел $2, 3, \ldots, N$, то нужно сперва вычеркнуть все числа, делящиеся на 2, кроме 2. Затем взять число 3 и вычеркнуть все последующие числа, делящиеся на 3. Затем взять следующее невычеркнутое число (т. е. 5) и вычеркнуть все последующие делящиеся на него числа, и так далее. В итоге останутся лишь простые числа. Для реализации метода нужен большой объем памяти ЭВМ, однако для составления таблиц простых чисел он является наилучшим.

В книге [101, гл. 3] описаны эффективные алгоритмы, реализующие решето Эратосфена для построения таблиц простых чисел и вычисление факторных баз.

Тест на основе малой теоремы Ферма

Для проверки простоты чисел n порядка 10^{30}—10^{40} метод пробных делений уже неприменим. Следующий тест основан на малой теореме Ферма: если n простое, то для любого $a \in \mathbb{Z}$ выполнено сравнение

$$a^n \equiv a \pmod{n};$$

если же при этом $(a, n) = 1$, то

$$a^{n-1} \equiv 1 \pmod{n}.$$

Поэтому для проверки простоты n мы выбираем какое-либо $a \in \mathbb{Z}$ и проверяем выполнение малой теоремы Ферма за $O(\log n)$ арифметических операций (с помощью бинарного возведения в степень в кольце $\mathbb{Z}/n\mathbb{Z}$). Если малая теорема Ферма не выполнена, то n — составное. Если же она выполнена, то мы еще не можем сделать вывод о простоте n, поскольку теорема дает лишь необходимое условие. Этот тест является эффективным для обнаружения составных чисел. Например, для 100-значных чисел n вида

$$n = 10^{99} + \delta, \qquad \delta = 1, 3, 5, 7, \ldots,$$

мы проверяли выполнение теста $13^{n-1} \equiv 1 \pmod{n}$, и первые десять чисел, прошедших этот тест, оказались впоследствии простыми (дальнейшая проверка простоты проводилась алгоритмом Ленстры—Коена, о котором будет рассказано в одном из следующих параграфов).

Однако существуют такие составные числа n, называемые *числами Кармайкла*, что для любого $a \in \mathbb{Z}$ выполнено сравнение $a^n \equiv a \pmod{n}$.

Наименьшее такое n равно $561 = 3 \cdot 11 \cdot 17$. Докажем, что 561 — число Кармайкла. Сравнение $a^{561} \equiv a \pmod{561}$ равносильно тому, что $a^{561} \equiv a \pmod 3$, $a^{561} \equiv a \pmod{11}$, $a^{561} \equiv a \pmod{17}$. Если $3 \mid a$, то $a^{561} \equiv a \equiv 0 \pmod 3$. Если же $3 \nmid a$, то $a^2 \equiv 1 \pmod 3$, откуда $a^{560} \equiv 1 \pmod 3$, $a^{561} \equiv a \pmod 3$. Аналогичная проверка делается для 11 и 17. В работе [51] показано, что чисел Кармайкла бесконечно много.

Таким образом, для проверки простоты чисел малой теоремы Ферма недостаточно.

Определение 1.1. Пусть $n > 1$ — нечетное натуральное число, $n - 1 = 2^s \cdot d$, где d — нечетно. Число n называется *строго псевдопростым в базе* a, $a \in \mathbb{N}$, если $(a, n) = 1$ и либо $a^d \equiv 1 \pmod n$, либо $a^{d \cdot 2^r} \equiv -1 \pmod n$ при некотором r, $0 \leqslant r < s$.

Основанием для такого определения служат следующие рассуждения. Если n простое, то $\mathbb{Z}/n\mathbb{Z}$ — поле, и $a^{n-1} \equiv 1 \pmod n$, т. е. $a^{2^s \cdot d} \equiv 1 \pmod n$. Отсюда $a^{2^{s-1} \cdot d} \equiv \pm 1 \pmod n$. Если $a^{2^{s-1} \cdot d} \equiv -1 \pmod n$, то мы останавливаемся, если $a^{2^{s-1} \cdot d} \equiv 1 \pmod n$, то снова извлекаем корень до тех пор, пока не дойдем до a^d или пока корень не будет сравним с -1.

В работе [228] эмпирическим путем получен следующий тест проверки простоты нечетных чисел n, $7 < n < 25 \cdot 10^9$.

1 шаг. Проверить строгую псевдопростоту n в базах 2, 3, 5, 7. Если n не строго псевдопростое в одной из баз, то оно составное.

2 шаг. Если $n = 3215031751$, то n — составное, иначе — простое.

Итак, мы видим, что тест на строгую псевдопростоту является эффективным для обнаружения составных чисел, однако и он является лишь необходимым условием для простоты n. В работе [12] предложен алгоритм, содержащий необходимые и достаточные условия для проверки простоты чисел $n < (67107840)^2$. Он использует кроме тестов на псевдопростоту некоторые свойства чисел Ферма.

Новые результаты о тестах на псевдопростоту были получены в работе [52]. В частности, было показано, что существует бесконечно много нечетных составных n, для которых наименьшее $\omega(n) \in \mathbb{N}$, такое, что n не является строго псевдопростым в базе $\omega(n)$, удовлетворяет неравенству

$$\omega(n) > (\log n)^{\frac{1}{3 \log\log\log n}}.$$

Об элементарных методах проверки простоты чисел см. также обзорные работы [40; 10] и книгу [77].

§ 1.3. Тесты на простоту для чисел специального вида

Рассмотрим сначала числа n вида $n = 2^m + 1$, где $m \in \mathbb{N}$. Если m делится на простое $p > 2$, т. е. $m = pm_1$, $m_1 \geqslant 1$, то $n = (2^{m_1})^p + 1$ делится на $2^{m_1} + 1$, т. е. является составным. Поэтому простым число n может быть лишь при $m = 2^k$.

Определение 1.2. Числа $F_k = 2^{2^k} + 1$, $k = 0, 1, 2, \ldots$, называются *числами Ферма*.

В настоящее время известно, что числа F_0, F_1, F_2, F_3, F_4 — простые, а все последующие числа Ферма, которые удалось проверить на простоту, оказались составными. При этом такая проверка оказалась возможной, например, для F_{23471} (это рекорд 1984 г. [161]), а про F_{31} до сих пор неизвестно (см. [101]), простое оно или составное.

Для проверки простоты чисел Ферма существует следующий тест.

Теорема 1.3. *Число $n = F_k$ при $k > 0$ является простым тогда и только тогда, когда*

$$3^{\frac{n-1}{2}} \equiv -1 \pmod{n}.$$

Доказательство. Докажем достаточность. Поскольку $n - 1 = 2^{2^k}$ — степень 2, порядок 3 (mod n) равен $n - 1 = 2^{2^k}$. Следовательно, $(\mathbb{Z}/n\mathbb{Z})^*$ содержит не менее $n - 1$ элемента, и поэтому все ненулевые элементы $\mathbb{Z}/n\mathbb{Z}$ обратимы, т. е. n — простое число.

Теперь докажем необходимость. Заметим, что $2^{2^k} = 4^{2^{k-1}} \equiv 1 \pmod 3$. Поэтому $n > 3$, $n \equiv 2 \pmod 3$, $n \equiv 1 \pmod 4$. По квадратичному закону взаимности $\left(\frac{3}{n}\right) = \left(\frac{n}{3}\right) \cdot (-1)^{\frac{n-1}{2} \cdot \frac{3-1}{2}} = \left(\frac{n}{3}\right) = \left(\frac{2}{3}\right) = -1$; по критерию Эйлера $\left(\frac{3}{n}\right) \equiv 3^{\frac{n-1}{2}} \pmod n$. □

Тест теоремы проверяется за $O(\log n)$ арифметических операций по модулю n, однако числа Ферма растут очень быстро, и поэтому данный тест так же быстро становится неэффективным.

Теперь рассмотрим числа n вида $n = 2^m - 1$. Если m — составное, $m = ab$, $1 < a \leqslant b$, то $n = 2^{ab} - 1$ делится на $2^a - 1$. Поэтому простым число n может быть лишь при простом m.

Определение 1.4. Пусть p — простое число, и $M_p = 2^p - 1$ — также простое. Тогда M_p называется *числом Мерсенна*.

С помощью чисел Мерсенна получаются все четные совершенные числа; они имеют вид $2^{p-1} \cdot M_p$. Числа Мерсенна крайне редки;

в 2001 г. было найдено тридцать девятое число Мерсенна — это $M_{13466917}$. О числах Мерсенна см. также [36, Приложения 5.1, 5.2].

Для проверки простоты чисел Мерсенна используется следующий тест.

Теорема 1.5. *Пусть q — простое число, $q > 2$, $n = 2^q - 1$. Рассмотрим последовательность $L_0, L_1, L_2, \ldots$, определяемую соотношениями*

$$L_0 = 4; \qquad L_{j+1} \equiv L_j^2 - 2 \pmod{n}.$$

Число n — простое тогда и только тогда, когда $L_{q-2} \equiv 0 \pmod{n}$.

Доказательство этой теоремы мы приведем в конце данного параграфа.

Некоторые результаты о простоте чисел Ферма получены в работе [12]. Например, справедлива следующая теорема.

Теорема 1.6. *Пусть p — простое число, $p \equiv 3 \pmod 4$, $M_p = 2^p - 1$ — число Мерсенна. Число Ферма F_p является простым тогда и только тогда, когда*

$$M_p^{(F_p-1)/2} \equiv -1 \pmod{F_p}.$$

Доказательство. Достаточность доказывается аналогично теореме 1.3. Докажем необходимость. По малой теореме Ферма $2^{2^p-1} \equiv 2 \pmod{M_p}$, откуда $F_p = 2 \cdot 2^{2^p-1} + 1 \equiv 5 \pmod{M_p}$. Следовательно, $(F_p, M_p) = 1$. Далее,

$$M_p^{\frac{F_p-1}{2}} \equiv \left(\frac{M_p}{F_p}\right) \equiv \left(\frac{F_p}{M_p}\right) \equiv \left(\frac{5}{M_p}\right) \equiv \left(\frac{M_p}{5}\right) \pmod{F_p}$$

(мы воспользовались критерием Эйлера и квадратичным законом взаимности для символа Лежандра). Далее, $M_p = 2^{4k+3} - 1 \equiv 2^3 - 1 \pmod 5 \equiv 2 \pmod 5$, $\left(\frac{M_p}{5}\right) = \left(\frac{2}{5}\right) = -1$. $\square$

Прежде чем приступить к доказательству теоремы 1.5, мы должны изучить свойства последовательностей чисел Люка́. Эти последовательности используются также в более общем $(N+1)$-методе проверки простых чисел (см. следующий параграф).

Определение 1.7. Последовательности $u_0, u_1, u_2, \ldots$ и $v_0, v_1, v_2, \ldots$, где $u_0 = 0$, $u_1 = 1$, $v_0 = 2$, $v_1 = 4$, а последующие члены получаются по рекуррентной формуле $x_{j+1} = 4x_j - x_{j-1}$, называются *последовательностями чисел Люка́*.

Очевидно, что $u_2 = 4$, $u_3 = 15$, $v_2 = 14$. Рассмотрим свойства последовательностей чисел Люка́.

Лемма 1.8. *$v_j = u_{j+1} - u_{j-1}$ при $j \geqslant 1$.*

Доказательство. Для $j = 1, 2$ утверждение очевидно. Пусть оно верно для всех номеров, не превосходящих j. Тогда при $j \geqslant 2$ получим

$$v_{j+1} = 4v_j - v_{j-1} = 4(u_{j+1} - u_{j-1}) - (u_j - u_{j-2}) = \\ = 4u_{j+1} - u_j - (4u_{j-1} - u_{j-2}) = u_{j+2} - u_j.$$

Лемма доказана. □

Лемма 1.9. $u_j = \dfrac{(2+\sqrt{3})^j - (2-\sqrt{3})^j}{2\sqrt{3}}$ *при* $j = 0, 1, 2, \ldots$

Доказательство. При $j = 0, 1$ утверждение очевидно. Далее, характеристическое уравнение рекуррентной последовательности имеет вид $\lambda^2 - 4\lambda + 1 = 0$; его корни равны $2 \pm \sqrt{3}$. Утверждение леммы теперь очевидно. □

Лемма 1.10. $v_j = (2+\sqrt{3})^j + (2-\sqrt{3})^j$ *при* $j = 0, 1, 2, \ldots$

Доказательство аналогично предыдущему.

Лемма 1.11. $u_{j+k} = u_j u_{k+1} - u_{j-1} u_k$ *при условии, что* $j - 1 \geqslant 0$, $k \geqslant 0$.

Доказательство. Проведем индукцию по $j + k = N$. Если $N = 1$, то $j = 1$, $k = 0$. Тогда равенство $u_1 = u_1 u_1 - u_0 u_0$, очевидно, выполнено.

Пусть утверждение верно при $j + k \leqslant N$. Докажем его для $N + 1$. Представим $N + 1$ в виде $N + 1 = j + (k + 1)$. Тогда $N = j + k$, $N - 1 = j + (k - 1)$,

$$u_{N+1} = 4u_N - u_{N-1} = 4(u_j u_{k+1} - u_{j-1} u_k) - (u_j u_k - u_{j-1} u_{k-1}) = \\ = u_j(4u_{k+1} - u_k) - u_{j-1}(4u_k - u_{k-1}) = u_j u_{k+2} - u_{j-1} u_{k+1},$$

что и требовалось доказать. Случай $N + 1 = (j + 1) + k$ рассматривается аналогично. □

Лемма 1.12. $u_{2n} = u_n v_n$.

Доказательство легко следует из лемм 1.9 и 1.10.

Лемма 1.13. $v_{2n} = v_n^2 - 2$.

Доказательство. В самом деле,

$$v_{2n} = (2+\sqrt{3})^{2n} + (2-\sqrt{3})^{2n},\ v_n^2 = ((2+\sqrt{3})^n + (2-\sqrt{3})^n)^2 = v_{2n} + 2.\ \square$$

Лемма 1.14. *Пусть* p *— простое число,* $e \in \mathbb{N}$. *Если* $u_j \equiv 0 \pmod{p^e}$, *то* $u_{pj} \equiv 0 \pmod{p^{e+1}}$.

Доказательство. Пусть числа a и b таковы, что $u_j = p^e \cdot b$, $u_{j+1} = a$. Тогда

$$u_j(2u_{j+1} - 4u_j) = u_j(2(4u_j - u_{j-1}) - 4u_j) = u_j(4u_j - 2u_{j-1}) = \\ = u_j(u_{j+1} + u_{j-1} - 2u_{j-1}) = u_j u_{j+1} - u_{j-1} u_j = u_{2j}$$

по лемме 1.11. Поэтому

$$u_{2j} = b \cdot p^e(2a - 4bp^e) \equiv 2au_j \pmod{p^{e+1}}.$$

Далее, по лемме 1.11

$$u_{2j+1} = u_{(j+1)+j} = u_{j+1}^2 - u_j^2 \equiv a^2 \pmod{p^{e+1}}.$$

Предположим, что выполнены следующие соотношения:

А) $u_{(k-1)j} \equiv (k-1)a^{k-2}u_j \pmod{p^{e+1}}$,

Б) $u_{(k-1)j+1} \equiv a^{k-1} \pmod{p^{e+1}}$.

Тогда по лемме 1.11 при $kj = j + (k-1)j$ получим

$$\begin{aligned} u_{kj} &= u_{(k-1)j+1}u_j - u_{(k-1)j}u_{j-1} \equiv \\ &\equiv a^{k-1}u_j - (k-1)a^{k-2}u_ju_{j-1} \pmod{p^{e+1}} \equiv \\ &\equiv a^{k-1}u_j - (k-1)a^{k-2}u_j(4u_j - u_{j+1}) \pmod{p^{e+1}} \equiv \\ &\equiv ka^{k-1}u_j \pmod{p^{e+1}}. \end{aligned}$$

Также при $kj + 1 = j + 1 + (k-1)j$ получим

$$u_{kj+1} = u_{(k-1)j+1}u_{j+1} - u_{(k-1)j}u_j \equiv a^k \pmod{p^{e+1}}.$$

Это означает, что соотношения А) и Б) выполняются и при замене $k-1$ на k.

Положим $k - 1 = p$. Тогда

$$u_{pj} \equiv pa^{p-1}u_j \pmod{p^{e+1}} \equiv 0 \pmod{p^{e+1}},$$

что и требовалось доказать. □

Лемма 1.15. *Справедливы равенства*:

$$u_j = \sum_{0 \leqslant k \leqslant \frac{j-1}{2}} \binom{j}{2k+1} 2^{j-2k-1}3^k, \qquad v_j = \sum_{0 \leqslant k \leqslant \frac{j}{2}} \binom{j}{2k} 2^{j-2k+1}3^k.$$

Доказательство. По лемме 1.9 имеем

$$\begin{aligned} u_j = \frac{1}{2\sqrt{3}} \sum_{l=0}^{j} \binom{j}{l} ((\sqrt{3})^l 2^{j-l} - (-\sqrt{3})^l 2^{j-l}) = \\ = \sum_{0 \leqslant l = 2k+1 \leqslant j} \binom{j}{2k+1} 2^{j-(2k+1)} \cdot 3^k. \end{aligned}$$

Формула для v_j получается из леммы 1.10. □

Лемма 1.16. *Если p — нечетное простое число, то*

$$u_p \equiv 3^{\frac{p-1}{2}} \pmod p, \qquad v_p \equiv 2^{p+1} \pmod p \equiv 4 \pmod p.$$

Доказательство. Поскольку биномиальный коэффициент удовлетворяет сравнению $\binom{p}{l} \equiv 0 \pmod p$ при $0 < l < p$, доказательство следует из леммы 1.15.□

Лемма 1.17. *Пусть $u_2 \equiv 0 \pmod 2$, $u_3 \equiv 0 \pmod 3$, и для любого простого p, $p > 3$, найдется значение $\varepsilon(p) \in \{\pm 1\}$ такое, что*

$$u_{p+\varepsilon(p)} \equiv 0 \pmod p.$$

Доказательство. По лемме 1.16 при $p > 3$ выполнены сравнения $u_p \equiv 3^{\frac{p-1}{2}} \equiv \pm 1 \pmod p$. Если $u_p \equiv 1 \pmod p$, то по лемме 1.8 получаем

$$u_{p-1} = 4u_p - u_{p+1} \equiv 4 - v_p - u_{p-1} \pmod p \equiv -u_{p-1} \pmod p,$$

откуда $\varepsilon(p) = -1$.

Если $u_p \equiv -1 \pmod p$, то

$$u_{p+1} = 4u_p - u_{p-1} \equiv -4 + v_p - u_{p+1} \pmod p \equiv -u_{p+1} \pmod p,$$

откуда $\varepsilon(p) = +1$. □

Определение 1.18. Пусть $N \in \mathbb{N}$. *Рангом $m(N)$* появления N в последовательности $\{u_j\}$ называется величина

$$m(N) = \min\{m \geqslant 1 \mid u_m \equiv 0 \pmod N\}.$$

Если же такого m не существует, то ранг $m(N)$ не определен.

Лемма 1.19. *Если ранг $m(N)$ определен, то $u_j \equiv 0 \pmod N$ тогда и только тогда, когда $j \equiv 0 \pmod{m(N)}$.*

Доказательство. Положим $m = m(N)$, $a \equiv u_{m+1} \pmod N$. Из рекуррентной формулы легко следует, что $(u_j, u_{j+1}) = 1$ для любого $j \geqslant$. Поэтому $(a, u_m) = 1$. По лемме 1.11 при $j \geqslant 1$ имеем

$$u_{m+j} = u_j u_{m+1} - u_{j-1} u_m \equiv u_j u_{m+1} \pmod N \equiv a u_j \pmod N.$$

Значит, члены последовательности $u_m, u_{m+1}, u_{m+2}, \ldots$ сравнимы по модулю N с членами последовательности $au_0, au_1, au_2, \ldots$

Следовательно, u_{m+j} делится на N тогда и только тогда, когда au_j делится на N, что равносильно делимости на N числа u_j. Отсюда следует утверждение леммы. □

Лемма 1.20. *Для последовательности чисел L_j из теоремы 1.5 справедливо соотношение* $L_j \equiv v_{2^j} \pmod n$.

Доказательство. Очевидно, что $L_0 = v_1$. Если утверждение леммы верно для j, то, применяя лемму 1.13, получаем

$$L_{j+1} \equiv (L_j^2 - 2) \pmod n \equiv v_{2^j}^2 - 2 \pmod n \equiv v_{2^{j+1}} \pmod n,$$

что и требовалось доказать. □

Лемма 1.21. *При $j \geqslant 0$ справедливы утверждения*:

1) $(u_j, u_{j+1}) = 1$;

2) $\text{НОД}(u_j, v_j) \leqslant 2$.

Доказательство. Первое утверждение было проверено при доказательстве леммы 1.19. Второе утверждение следует из первого и из равенства $2u_{j+1} = 4u_j + v_j$, которое справедливо по лемме 1.8. □

Приступим к доказательству теоремы 1.5.

Сначала докажем достаточность. Пусть $L_{q-2} \equiv 0 \pmod n$. Тогда $v_{2^{q-2}} \equiv 0 \pmod{2^q - 1}$ по лемме 1.20. Далее,

$$u_{2^{q-1}} = u_{2^{q-2}} v_{2^{q-2}} \equiv 0 \pmod{2^q - 1}$$

по лемме 1.12. Кроме того,

$$u_{2^{q-2}} \not\equiv 0 \pmod{2^q - 1},$$

так как по лемме 1.21 числа u_j и v_j не имеют общих нечетных простых делителей.

Пусть $m = m(2^q - 1)$ — ранг появления числа $n = 2^q - 1$ в последовательности $\{u_j\}$. Число m определено и, по лемме 1.19, m делит 2^{q-1}, но не делит 2^{q-2}. Значит, $m = 2^{q-1} = \dfrac{n+1}{2}$.

Предположим, что n — составное, и придем к противоречию. Пусть $n = p_1^{e_1} \dots p_r^{e_r}$ — разложение n на простые сомножители, и либо $r > 1$, либо $e_1 > 1$. Все $p_j > 3$, так как $n \equiv (-1)^q - 1 \equiv -2 \pmod 3$.

Положим

$$t = \text{НОК}(p_1^{e_1 - 1}(p_1 + \varepsilon_1), \dots, p_r^{e_r - 1}(p_r + \varepsilon_r)),$$

где $\varepsilon_j = \varepsilon(p_j) \in \{\pm 1\}$ — величины из леммы 1.17, $\varepsilon_j \in \{\pm 1\}$. Тогда $u_t \equiv 0 \pmod{2^q - 1}$ по леммам 1.14, 1.17, 1.19. Следовательно, по лемме 1.19 число t делится на $m = \dfrac{n+1}{2}$.

Пусть

$$n_0 = \prod_{j=1}^{r} p_j^{e_j - 1}(p_j + \varepsilon_j).$$

Тогда, так как $p_j \geqslant 5$, то

$$n_0 \leqslant \prod_{j=1}^{r} p_j^{e_j-1}\left(p_j + \frac{p_j}{5}\right) = n\left(\frac{6}{5}\right)^r.$$

Поскольку числа $p_j + \varepsilon_j$ четные, по свойству наименьшего общего кратного получаем, что $t \leqslant \frac{n_0}{2^{r-1}}$. Тогда

$$m \leqslant t \leqslant \left(\frac{6}{5}\right)^r n/2^{r-1} = 2 \cdot \left(\frac{3}{5}\right)^r \cdot n.$$

Так как $n = 2m - 1 < 2m$, то

$$m < 4 \cdot \left(\frac{3}{5}\right)^r \cdot m < 3m$$

в силу неравенств $4 \cdot \left(\frac{3}{5}\right)^r \leqslant 4 \cdot \frac{3}{5} < 3$. Кроме того, при $r \geqslant 3$ имеем $4 \cdot \left(\frac{3}{5}\right)^r \leqslant 4 \cdot \frac{27}{125} < 1$. Значит, $r = 1$ или $r = 2$.

Поскольку t делится на m и $m \leqslant t < 3m$, то $t = m = 2^{q-1}$ или $t = 2m = 2^q$. То есть t является степенью числа 2. Следовательно, по определению числа t получим, что $e_1 = 1 = e_r$. Кроме того, $p_1 + \varepsilon_1$ и $p_r + \varepsilon_r$ — степени 2, т. е. $p_1 + \varepsilon_1 = 2^k$, $p_2 + \varepsilon_2 = 2^l$. Если n составное, то $r = 2$, и тогда

$$n = 2^q - 1 = (2^k \pm 1)(2^l \mp 1)$$

($\varepsilon_1 = -\varepsilon_2$, так как $n \equiv -1 \pmod 4$). Отсюда $2^q = 2^{k+l} \pm 2^l \mp 2^k$. Пользуясь делимостью всех слагаемых на $2^{\max(k,l)}$, находим, что $k = l$. Тогда $n = 2^q - 1 = (2^k + 1)(2^k - 1)$, что противоречит нечетности q. Достаточность доказана.

Теперь докажем необходимость. Пусть $n = 2^q - 1$ — простое число. Покажем, что $v_{2^{q-2}} \equiv 0 \pmod n$, и тогда по лемме 1.20 получим $L_{q-2} \equiv 0 \pmod n$, что и требуется доказать.

По лемме 1.13 имеем $v_{2^{q-1}} = (v_{2^{q-2}})^2 - 2$, поэтому надо показать, что $v_{2^{q-1}} \equiv -2 \pmod n$. Справедливо равенство

$$2 \pm \sqrt{3} = \left(\frac{\sqrt{2} \pm \sqrt{6}}{2}\right)^2.$$

Тогда по лемме 1.10 получаем:

$$v_{2^{q-1}} = \left(\frac{\sqrt{2}+\sqrt{6}}{2}\right)^{n+1} + \left(\frac{\sqrt{2}-\sqrt{6}}{2}\right)^{n+1} =$$
$$= 2^{-n} \sum_{0\leqslant k\leqslant \frac{n+1}{2}} \binom{n+1}{2k} (\sqrt{2})^{n+1-2k} (\sqrt{6})^{2k} =$$
$$= 2^{\frac{n+1}{2}-n} \sum_{0\leqslant k\leqslant \frac{n+1}{2}} \binom{n+1}{2k} \cdot 3^k = 2^{\frac{1-n}{2}} \sum_{0\leqslant k\leqslant \frac{n+1}{2}} \binom{n+1}{2k} \cdot 3^k.$$

Так как n — нечетное простое число, то $\binom{n+1}{2k} = \dfrac{(n+1)!}{(2k)!(n+1-2k)!}$ делится на n при всех k, кроме $k=0$ и $k=\dfrac{n+1}{2}$. Следовательно,

$$2^{\frac{n-1}{2}} \cdot v_{2^{q-1}} \equiv 1 + 3^{\frac{n+1}{2}} \pmod{n}.$$

Но

$$3^{\frac{n-1}{2}} \equiv \left(\frac{3}{n}\right) \equiv (-1)^{\frac{3-1}{2}\cdot\frac{n-1}{2}} \left(\frac{n}{3}\right) \equiv -1 \pmod{n}.$$

Поэтому

$$2^{\frac{n-1}{2}} v_{2^{q-1}} \equiv 1 + 3\cdot(-1) \equiv -2 \pmod{n}.$$

Далее, $n = 2^q - 1 \equiv -1 \pmod 8$, откуда

$$2^{\frac{n-1}{2}} \equiv \left(\frac{2}{n}\right) \equiv 1 \pmod{n}.$$

В итоге $v_{2^{q-1}} \equiv -2 \pmod{n}$, что мы и хотели доказать.

Теорема 1.5 полностью доказана.

Замечание 1.22. В книгах Рибенбойма [231; 232] содержится описание различных рекордов в области проверки простоты чисел, в частности, для чисел Ферма и Мерсенна, для построения пар близнецов (простых чисел вида p и $p+2$) и для других простых чисел специального вида.

§ 1.4. $(N \pm 1)$-методы проверки простоты чисел и построения больших простых чисел

В данном параграфе мы описываем методы, с помощью которых можно проверять простоту натурального числа N, если известно полное или частичное разложение $N-1$ или $N+1$ на множители. Прекрас-

ная библиография по этому вопросу содержится в работе [40], а более поздние работы можно найти в библиографии к [60]. Мы также опишем здесь некоторые способы построения больших простых чисел, широко используемых в криптографии. Иные способы можно найти в работах [214; 99; 259; 186].

Докажем сначала теорему, относящуюся к $(N-1)$-методам проверки простоты чисел.

Теорема 1.23. *Пусть $n \in \mathbb{N}$, $n > 1$, n нечетно, $n - 1 = \prod_{i=1}^{k} p_i^{\alpha_i}$ — известное разложение $n-1$ на простые множители. Если для каждого $i = 1, \ldots, k$ существует такое $a_i \in \mathbb{N}$, что*

$$a_i^{n-1} \equiv 1 \pmod{n}, \qquad a_i^{\frac{n-1}{p_i}} \not\equiv 1 \pmod{n},$$

то n — простое число.

Доказательство. Обозначим m_i порядок $a_i \pmod{n}$ в кольце $\mathbb{Z}/n\mathbb{Z}$. Из условия следует, что $m_i \mid n-1$, $m_i \nmid (n-1)/p_i$, поэтому $p_i^{\alpha_i} \mid m_i$ для $i = 1, \ldots, k$. Следовательно, $b_i \equiv a_i^{m_i/p_i^{\alpha_i}} \pmod{n}$ имеют порядок $p_i^{\alpha_i}$ в $(\mathbb{Z}/n\mathbb{Z})^*$, а элемент $b \equiv b_1 \cdot \ldots \cdot b_k \pmod{n}$ — порядок $p_1^{\alpha_1} \cdot \ldots \cdot p_k^{\alpha_k} = n-1$ в $(\mathbb{Z}/n\mathbb{Z})^*$. Поэтому $\mathbb{Z}/n\mathbb{Z}$ — поле, и число n — простое. $\square$

Как применять теорему 1.23 на практике? Зная разложение $n-1$ на множители, надо либо перебором подряд $a = 2, 3, \ldots$, либо случайным выбором a искать числа a_i, удовлетворяющие условию теоремы. Если для некоторого a, $1 < a < n$, окажется, что $a^{n-1} \not\equiv 1 \pmod{n}$, то n составное. Если же мы найдем $a_1, \ldots, a_k$, то тем самым покажем, что n простое.

Заметим, что мы фактически уже пользовались методом теоремы 1.23 применительно к числам Ферма (см. теорему 1.3 из § 1.3). При этом числа a_i искать было не нужно, нам заранее известно, что $a = 3$.

Аналогичный результат справедлив в случае, когда известно полное разложение $n+1$ на множители.

Теорема 1.24. *Пусть P, $Q \in \mathbb{Z}$, $D = P^2 - 4Q \neq 0$. Определим последовательность чисел Люка $u_0, u_1, \ldots$ дискриминанта D следующими соотношениями: $u_0 = 0$, $u_1 = 1$, $u_{j+2} = Pu_{j+1} - Qu_j$ при $j \geqslant 0$. Пусть n — нечетное натуральное число, $n > 1$, $n + 1 = \prod_{i=1}^{k} q_i^{\beta_i}$ — разложение на простые множители, и $\left(\frac{D}{n}\right) = -1$. Если для каждого $i = 1, \ldots, k$ найдутся такие P_i, $Q_i \in \mathbb{Z}$, $D = P_i^2 - 4Q_i$, что*

связанная с ними последовательность Люка $u_0^{(i)}, u_1^{(i)}, \ldots$ *удовлетворяет условиям*

$$n \mid u_{n+1}^{(i)}, \quad n \nmid u_{(n+1)/q_i}^{(i)},$$

то число n *— простое. Если же существует такая последовательность Люка* $\{u_j\}$ *дискриминанта* D, *что* $n \nmid u_{n+1}$, *то* n *— составное число.*

Эта теорема принадлежит к так называемым $(N+1)$-методам проверки простоты чисел. Ее доказательство, равно как и описание других результатов, относящихся к $(N+1)$-методам, выходит за рамки данной книги. Заметим, что простейший результат здесь, связанный с проверкой простоты чисел Мерсенна, мы рассмотрели в теореме 1.5 из § 1.3.

Вернемся теперь к $(N-1)$-методам проверки простоты чисел и покажем, как с их помощью можно строить большие простые числа.

Теорема 1.25. *Пусть* $n \in \mathbb{N}$, $n > 1$, n *нечетно*, $n - 1 = F_1 \cdot R_1$, *где* $(F_1, R_1) = 1$. *Пусть известно полное разложение* $F_1 = \prod_{j=1}^{k} q_j^{\alpha_j}$ *на простые сомножители. Если для любого* $j = 1, \ldots, k$ *найдется такое* $a_j \in \mathbb{N}$, *что*

$$a_j^{n-1} \equiv 1 \pmod{n}, \quad (a_j^{(n-1)/q_j} - 1, n) = 1,$$

то, при условии, что

$$F_1 \geqslant \sqrt{n},$$

число n *— простое.*

Доказательство. Пусть p — произвольный простой делитель числа n. Мы покажем, что $p > \sqrt{n}$, откуда следует простота n.

Из $a_j^{n-1} \equiv 1 \pmod{n}$ следует, что $a_j^{n-1} \equiv 1 \pmod{p}$, откуда $(a_j, p) = 1$ и порядок e_j элемента $a_j \pmod{p}$ в $\mathbb{Z}/p\mathbb{Z}$ делит $n-1$. Кроме того, по малой теореме Ферма $e_j \mid p-1$. Далее, из условия теоремы следует, что $a_j^{n-1/q_j} \not\equiv 1 \pmod{p}$, откуда $q_j^{\alpha_j} \mid e_j$. Следовательно, $q_j^{\alpha_j} \mid p-1$, и поэтому $F_1 = \prod_{j=1}^{k} q_j^{\alpha_j} \mid p-1$. Значит, $p - 1 \geqslant F_1$, $p > F_1 \geqslant \sqrt{n}$. $\square$

Покажем, как с помощью теоремы 1.25 можно строить большие простые числа. Мы строим последовательность простых чисел $p_1 < p_2 < p_3 < \ldots$, пока не найдем простое число нужной нам величины. Простое нечетное число p_1 выбираем произвольно, например, $p_1 = 3$. Пусть мы уже построили простое p_{i-1}. Выберем случайное r, $1 \leqslant r \leqslant p_{i-1} - 1$. Пусть $r = 2^s \cdot t$, t — нечетно. Тогда в качестве кандидата на очередное простое p_i возьмем $n = 2rp_{i-1} + 1 = 2^{s+1} p_{i-1} \cdot t + 1$.

Положим $F_1 = 2^{s+1}p_{i-1}$, $R_1 = t$. Очевидно, что $(F_1, R_1) = 1$. Далее, $F_1 > \sqrt{n}$, поскольку $n = 2^{s+1}p_{i-1}t + 1 < 2^{s+2}p_{i-1}t < 2^{s+2}p_{i-1}^2 \leqslant F_1^2$. Следовательно, для доказательства простоты n нужно найти (перебором) числа a_1 и a_2 такие, что

$$a_1^{n-1} \equiv a_2^{n-1} \equiv 1 \pmod{n}, \qquad \left(a_1^{\frac{n-1}{2}} - 1,\ n\right) = \left(a_2^{\frac{n-1}{p_{i-1}}} - 1,\ n\right) = 1.$$

Если в ходе перебора найдется такое a, что $a^{n-1} \not\equiv 1 \pmod{n}$, или один из двух наибольших общих делителей с n дает нетривиальный делитель числа n, то n — составное; тогда нужно выбрать другое случайное r (и другое n). Если же мы докажем простоту n, то положим $p_i = n$.

Другой способ применения теоремы 1.25 заключается в следующем. Мы снова строим цепочку простых чисел, и пусть построено $p_{i-1} > 3$. Выберем случайное четное r, $1 \leqslant r \leqslant p_{i-1} - 3$, и положим $n = p_{i-1}r + 1$. Пусть $F_1 = p_{i-1}$, $R_1 = r$, $(F_1, R_1) = 1$. Нам нужно найти всего лишь одно натуральное a такое, что $a^{n-1} \equiv 1 \pmod{n}$, $(a^r - 1,\ n) = 1$ $\left(\text{поскольку } \dfrac{n-1}{p_{i-1}} = r\right)$. Действительно, неравенство $F_1 = p_{i-1} > \sqrt{n}$ выполнено, так как

$$n = p_{i-1}r + 1 \leqslant p_{i-1}(p_{i-1} - 3) + 1 = p_{i-1}^2 - 3p_{i-1} + 1 \leqslant$$
$$\leqslant p_{i-1}^2 - 3 \cdot 5 + 1 < p_{i-1}^2.$$

Перебор a осуществляется так же, как в предыдущем способе.

Следующая теорема является более эффективной для построения больших простых, поскольку в ней отсутствует вычисление наибольшего общего делителя.

Теорема 1.26. *Пусть $n = 2rq + 1$, где q — нечетное простое число, и $r \leqslant 2q + 1$. Если существует такое $a \in \mathbb{N}$, что*

$$a^{n-1} \equiv 1 \pmod{n}, \qquad a^{2r} \not\equiv 1 \pmod{n},$$

то n — простое число.

Доказательство. Пусть n — составное, $n = pN$, где p — простое, $N > 1$. Поскольку $n \nmid a^{2r} - 1$, то найдется простой делитель n, который входит в n в большей степени, чем в $a^{2r} - 1$. Его и обозначим через p. Тогда $\nu_p(n) > \nu_p(a^{2r} - 1)$, и при $s = \nu_p(a^{2r} - 1) + 1 \geqslant 1$ имеем $p^s \mid n$, $p^s \nmid a^{2r} - 1$.

Пусть d — порядок $a \pmod{p^s}$ в $\mathbb{Z}/p^s\mathbb{Z}$. Число d определено и $d \mid n - 1 = 2rq$, поскольку $a^{n-1} \equiv 1 \pmod{n}$. Кроме того, $d \mid \varphi(p^s) = p^{s-1}(p - 1)$. Но из соотношения $a^{2r} \not\equiv 1 \pmod{p^s}$ следует, что $d \nmid 2r$.

Значит, $q \mid d$. Поэтому $q \mid p^{s-1}(p-1)$. Числа q и p различны, так как $p \mid n$, $q \mid n-1$. Следовательно, $q \mid p-1$. Отсюда $p \equiv 1 \pmod{q}$, и в силу нечетности p и q получаем, что $p \equiv 1 \pmod{2q}$. Кроме того, $n \equiv 1 \pmod{2q}$. Поэтому, так как $n = pN$, $N \equiv 1 \pmod{2q}$. Поскольку $p > 1$ и $N > 1$, то $p \geqslant 1 + 2q$, $N \geqslant 1 + 2q$. Значит, $n = pN \geqslant (1+2q)^2$. Однако $n = 2rq + 1 \leqslant 2q(2q+1) + 1 < (1+2q)^2$. Полученное противоречие доказывает теорему. □

Замечание 1.27. Из теоремы 1.26 также следует метод построения простых чисел, аналогичный предыдущим.

Замечание 1.28. Если заменить неравенство $r \leqslant 2q+1$ на неравенство $r \leqslant 2q+2$, то утверждение теоремы 1.26 о простоте n будет неверным. Можно даже привести пример a и n, для которых условия теоремы будут выполнены, но n будет составным. Следующая теорема показывает, как можно увеличить верхнюю границу для r.

Теорема 1.29. *Пусть* $n = 2rq + 1$, *где* q — *простое число*, $r \leqslant 4q+2$. *Пусть существует такое* $a \in \mathbb{N}$, *что*

$$a^{n-1} \equiv 1 \pmod{n}, \qquad a^{2r} \not\equiv 1 \pmod{n}.$$

Тогда либо n — *простое*, *либо* $n = p^2$, *где* $p = 2q+1$ — *простое число и* $a^{p-1} \equiv 1 \pmod{p^2}$.

Доказательство. Пусть n — составное, $n = pN$, где p и N те же, что в доказательстве теоремы 1.26. Мы показали, что

$$n \equiv p \equiv N \equiv 1 \pmod{2q}.$$

Если одно из чисел p и N строго больше $1+2q$, то оно больше или равно $1+4q$, и тогда $n = pN \geqslant (1+2q)(1+4q) = 8q^2 + 6q + 1$. Но по условию $n \leqslant 2q(4q+2) + 1 = 8q^2 + 4q + 1$. Следовательно, $p = N = 1 + 2q$, $n = p^2$. Осталось показать, что $a^{p-1} \equiv 1 \pmod{p^2}$. Действительно, по условию $a^{p^2-1} \equiv 1 \pmod{p^2}$, и по теореме Эйлера $a^{p^2-p} \equiv 1 \pmod{p^2}$, откуда следует требуемое сравнение. □

Замечание 1.30. Если q известно, то проверить равенство $n = (2q+1)^2$ очень легко. То есть, узнав a, мы будем знать, простое n или составное. Тест теоремы 1.29 очень эффективен для построения больших простых чисел, так как чем больше верхняя граница для выбора случайного r, тем дальше мы можем «шагнуть» при построении очередного простого числа.

В рассмотренных выше теоремах 1.25—1.29 величина разложенной части $n-1$ имеет порядок $\sqrt{n}$. Следующая теорема ослабляет это условие до величины порядка $n^{1/3}$ (см. [99]).

Теорема 1.31. *Пусть n — нечетно, $n > 1$, $n - 1 = F_1 R_1$, где $(F_1, R_1) = 1$, F_1 — четно, и известно полное разложение F_1 на простые множители. Пусть для любого простого делителя q числа F_1 найдется такое $a_q \in \mathbb{N}$, что*

$$a_q^{n-1} \equiv 1 \pmod{n}, \quad (a_q^{(n-1)/q} - 1, n) = 1.$$

Пусть $m \in \mathbb{N}$ и для каждого $l = 1, 2, \ldots, m-1$ верно, что $lF_1 + 1 \nmid n$. Если

$$n < (mF_1 + 1)(2F_1^2 + (L - m)F_1 + 1),$$

где $R_1 = 2F_1 L_1 + L$, $1 \leqslant L < 2F_1$, то n — простое тогда и только тогда, когда либо $R_1 = L$, либо число $L^2 - 8L_1$ не является полным квадратом.

Обзор результатов, относящихся к $(N \pm 1)$-методам проверки простоты, см. в [40; 10].

Ряд результатов о построении больших простых чисел, соединяющих в себе $N-1$-метод и тесты с тригонометрическими суммами, приведен в работе [11]. Например, справедлива следующая теорема.

Теорема 1.32. *Пусть $n \in \mathbb{N}$, $n > 1$, $(30, n) = 1$ и i — мнимая единица. Пусть также выполнены следующие условия*:

1) $3^{(n-1)/2} \equiv (-1)^{(n-1)/2} \left(\frac{n}{3}\right) \pmod{n}$;

2) *если* $n \equiv 1 \pmod 8$, *то существует* $a \in \mathbb{N}$ *такое, что* $a^{(n-1)/2} \equiv -1 \pmod{n}$;

3) *если* $n \equiv 3, 5 \pmod 8$, *то* $2^{(n-1)/2} \equiv -1 \pmod{n}$;

4) *если* $n \equiv 1 \pmod 4$, *то при* $i = (-1)^{1/2} \in \mathbb{C}$

$$(2i + 1)^{(n-1)/2} \cdot 5^{(n-1)/4} \equiv \xi \pmod{n\mathbb{Z}[i]},$$

а если $n \equiv 3 \pmod 4$, *то*

$$(2i + 1)^{(n+1)/2} \cdot 5^{(n-3)/4} \equiv \xi \pmod{n\mathbb{Z}[i]},$$

где $\xi^4 = 1$;

5) *если* $n \equiv 7 \pmod 8$, *то*

$$5^{(n-1)/2} \equiv -1 \pmod{n}$$

и

$$(2i + 1)^{(n+1)/2} \cdot 5^{(n-3)/4} \equiv \xi \pmod{n)\mathbb{Z}[i]},$$

где $\xi = i$ *или* $\xi = -i$.

Тогда для любого $s \in \mathbb{N}$, делящего n, выполнено сравнение $s = n^j \pmod{240}$ при некотором j, $0 \leqslant j \leqslant 3$.

Эта теорема также применима для построения больших простых чисел. При этом, если уже построено простое число p_{j-1}, то очередное простое число ищется в виде $n = rp_{j-1} + 1$, где r — случайное четное число, $(r,\ p_{j-1}) = 1$, удовлетворяющее неравенству

$$r < 57600p_{j-1}.$$

Таким образом, мы получили дальнейшее увеличение верхней границы для r из теоремы 1.29, альтернативное теореме 1.31.

В заключение отметим, что существуют методы проверки простоты, соединяющие в себе информацию от разложений $N-1$ и $N+1$, а также использующие разложение N^2+1 и т. д. Информацию о них можно найти в обзорах [40; 10].

В конце этого параграфа дадим понятие сертификата простоты числа.

Определение 1.33. *Сертификатом Пратта* для простого нечетного натурального числа n называется набор $\{p_1,\ \ldots,\ p_k,\ a\}$, состоящий из всех различных простых делителей p_i числа $n-1$ и натурального числа a, удовлетворяющего соотношениям

$$a^{n-1} \equiv 1 \pmod{n}, \qquad a^{(n-1)/p_j} \not\equiv 1 \pmod{n}, \qquad j = 1,\ \ldots,\ k.$$

Если число n — простое, то для него существует сертификат Пратта, хотя задача нахождения его для конкретного n может оказаться непростой. Если же сертификат Пратта известен, то с его помощью можно убедиться в простоте n за $O(\log^2 n)$ арифметических операций.

Сертификат простоты может иметь и другой вид; в частности, такой сертификат может быть получен с помощью эллиптических кривых. Об этом будет рассказано далее, в главе 4.

§ 1.5. Алгоритм Конягина—Померанса

Если $n \in \mathbb{N}$ и известно полное разложение $n-1$ на простые множители, или достаточно большая часть этого разложения, то можно с полиномиальной сложностью проверить, простое n или составное. Наилучшая оценка сложности здесь получена в алгоритме Конягина—Померанса [148]:

Теорема 1.34. *Пусть* $n \in \mathbb{N}$, $n > 1$, n *— нечетно*, $n-1 = \prod_{j=1}^{k} q_j^{\alpha_j}$ *— известное разложение* $n-1$ *на простые множители. Тогда про-*

верку простоты n можно провести за $O\Big(\frac{(\log n)^{17/7}}{\log\log n}\Big)$ арифметических операций.

Теорема 1.35. *Пусть $n \in \mathbb{N}$, $n > 1$, n — нечетно, $n - 1 = F_1 R_1$, где $(F_1, R_1) = 1$, и известно разложение F_1 на простые сомножители. Если $F_1 \geqslant n^{\frac{1}{4n}+\varepsilon}$, где ε — положительная постоянная, то проверку простоты n можно провести за $O((\log n)^{c(\varepsilon)})$ арифметических операций (здесь $c(\varepsilon)$ — положительная постоянная, зависящая от ε).*

Мы приведем здесь более бесхитростный алгоритм, который в условиях теоремы 1.34 проверяет, простое n или составное, за $O((\log n)^5)$ арифметических операций. Далее мы предполагаем, что выполнены обозначения и условия этой теоремы. Мы также будем считать, что $n > 5$.

Лемма 1.36. *Пусть $a, m \in \mathbb{N}$, $a^m \equiv 1 \pmod n$, и пусть для любого простого числа q, делящего m, $(a^{m/q} - 1, n) = 1$. Тогда, если p — простое и $p \mid n$, то $p \equiv 1 \pmod m$.*

Доказательство. Очевидно, что m — порядок $a \pmod n$ в $\mathbb{Z}/n\mathbb{Z}$. Пусть p — простое число, делящее n, и пусть k — порядок $a \pmod p$. Тогда $k = m$. Действительно, $k \mid m$, так как из $a^m \equiv 1 \pmod n$ следует, что $a^m \equiv 1 \pmod p$. Если $k < m$, то найдется простое число q, $q \mid m$, что $k \mid m/q$. Отсюда $a^{m/q} \equiv 1 \pmod p$, т. е. $p \mid (a^{m/q} - 1, n)$, что противоречит условию леммы.

Далее, по малой теореме Ферма $a^{p-1} \equiv 1 \pmod p$. Следовательно, $m \mid p - 1$, что и требовалось доказать. □

Алгоритм проверки простоты числа.

1 этап. Заготавливаем таблицу всех простых чисел, не превосходящих $[\log^2 n] + 1$. Присваиваем $F(1) := 1$. Затем для каждого $a = 2, 3, \ldots$ $\ldots, [\log^2 n] + 1$ осуществляем 2-й этап, пока не докажем, что n — простое или же что n — составное число.

2 этап.

1 шаг. Если a — составное (см. таблицу 1 этапа), то $F(a) := F(a - 1)$ и идем на 6 шаг. Если a — простое, и $a^{F(a-1)} \equiv 1 \pmod n$, то $F(a) := F(a - 1)$ и идем на 6 шаг. Иначе проверяем, выполняется ли сравнение:

$$a^{n-1} \equiv 1 \pmod n.$$

Если нет, то n составное.

2 шаг. Используя разложение $n - 1$ на простые сомножители, найти порядок числа $a \pmod n$, т. е. наименьшее натуральное число $E(a)$, такое, что $a^{E(a)} \equiv 1 \pmod n$.

3 шаг. Проверяем, выполняется ли условие:

$$\text{НОД}\Bigg(\prod_{\substack{q\mid E(a)\\ q\ —\ \text{простое}}}(a^{E(a)/q}-1),\ n\Bigg)=1.$$

Если нет, то n — составное.

4 шаг. $F(a):=\text{НОК}(F(a-1),\ E(a))$.

5 шаг. Если $F(a)\geqslant\sqrt{n}$, то n — простое число.

6 шаг. Если $a\leqslant[\log^2 n]$, то возвращаемся к началу выполнения 2 этапа для следующего значения a. Если же $a=[\log^2 n]+1$, то n — составное.

Конец алгоритма.

Докажем корректность алгоритма и получим оценку его сложности.

Таблица простых чисел на 1 этапе находится с помощью решета Эратосфена за $O(\log^4 n)$ арифметических операций.

Текущее значение $F(a)$ является делителем $n-1$, поэтому 1 шаг 2 этапа выполняется за $O(\log n)$ операций (бинарное возведение в степень).

2 шаг 2 этапа выполняется за $O(\log^3 n)$ операций с помощью следующего вспомогательного алгоритма (см. [89, гл. 1]).

Алгоритм нахождения порядка элемента.

На входе алгоритма заданы a, $n\in\mathbb{N}$, $n-1=\prod\limits_{j=1}^{N}p_j^{\alpha_j}$ — известное разложение $n-1$ на простые множители; на выходе порядок $a \pmod n$ в $\mathbb{Z}/n\mathbb{Z}$.

1 шаг. $M:=n-1$, $j:=0$.

2 шаг. $j:=j+1$, $M:=M/p_j^{\alpha_j}$, $A:=a^M$.

3 шаг (цикл). Для $l=0,\ 1,\ \ldots,\ \alpha_j$ проверить, выполнено ли сравнение:

$$A\equiv 1 \pmod n.$$

Если да, то перейти на 4 шаг. Иначе

$$M:=Mp_j,\qquad A:=A^{p_j};$$

перейти к следующему значению l в цикле.

4 шаг. Если $j<N$, то вернуться на 2 шаг; иначе выдать M.

Конец алгоритма.

Корректность алгоритма нахождения порядка очевидна. Для получения оценки его сложности заметим, что

$$n-1 \geqslant \prod_{j=1}^{N} p_j \geqslant 2^N,$$

откуда $N = O(\log n)$, $p_j = O(\log n)$; также $\alpha_j = O(\log n)$. Поэтому внешний и внутренний циклы делают $O(\log n)$ шагов, и на каждом шаге делается $O(\log n)$ арифметических операций. Суммарная сложность — $O(\log^3 n)$ операций.

Вернемся к алгоритму проверки простоты. Если НОД на 3 шаге 2 этапа не равен 1, то n составное. Действительно, по определению $E(a)$ ни одно из чисел $a^{E(a)/q} - 1$ не делится на n. Значит, если НОД больше 1, то одно из чисел $a^{E(a)/q} - 1$ имеет с n нетривиальный общий делитель d, $1 < d < n$. Сложность 3 шага 2 этапа — $O(\log^2 n)$ операций.

После прохождения 4 шага выполнено следующее: для всех b, $2 \leqslant b \leqslant a$,

$$b^{F(a)} \equiv 1 \pmod{n}.$$

Докажем корректность 5 шага. У нас $a^{F(a)-1} \not\equiv 1 \pmod{n}$, $a^{E(a)} \equiv \equiv 1 \pmod{n}$, и для любого простого q, $q \mid E(a)$,

$$\text{НОД}(a^{E(a)/q} - 1,\, n) = 1.$$

Также $a^{F(a)} \equiv 1 \pmod{n}$. Если мы докажем, что для любого простого p, $p \mid n$, выполнено сравнение $p \equiv 1 \pmod{F(a)}$, то $p \geqslant 1 + F(a) \geqslant 1 + \sqrt{n}$, откуда следует простота n. Сравнение $p \equiv 1 \pmod{F(a)}$ докажем индукцией по a. Предположим, что $p \equiv 1 \pmod{F(a-1)}$. Тогда по лемме получим $p \equiv 1 \pmod{E(a)}$, откуда

$$p \equiv 1 \pmod{\text{НОК}(F(a-1),\, E(a))} \equiv 1 \pmod{F(a)},$$

что и требовалось доказать.

Теперь обоснуем 6 шаг 2 этапа. Предположим, что n — простое число, $a > [\log^2 n]$, $F = F(a) < \sqrt{n}$, и придем к противоречию. По построению $F(a) \mid n-1$, так как $E(a) \mid n-1$. Положим

$$H = \{b \in \{1, \ldots, n-1\} \mid b^{F(a)} \equiv 1 \pmod{n}\}.$$

Из простоты n следует, что $|H| = F$. Действительно, $x^{n-1} \equiv 1 \pmod{n}$ для всех $x \in \mathbb{Z}/n\mathbb{Z}$; также $n - 1 = F \cdot M$,

$$x^{n-1} - 1 = x^{F \cdot M} - 1 = (x^F - 1)(x^{F(M-1)} + \ldots + 1),$$

т. е. ровно F элементов $\mathbb{Z}/n\mathbb{Z}$ — корни $x^F - 1$. Множество H содержит все a-гладкие числа, не превосходящие $n-1$, т. е.

$$H \supseteq H_1 = \{b \mid 1 \leqslant b \leqslant n, \text{ все простые делители } b \text{ не превосходят } a\}.$$

Это следует из того, что если r — простое, $r \leqslant a$, то $r^{F(a)} \equiv 1 \pmod{n}$. Само n не входит в H_1, т. к. оно простое и $n > a$. Отсюда $|H| \geqslant |H_1|$.

Справедлив следующий результат из теории распределения простых чисел (см. для справок [148]): если обозначить $\psi(n, a) = |H_1|$, то при $n \geqslant 5$, $2 \leqslant a \leqslant n$ выполнено неравенство

$$\psi(n, a) > n^{1 - \frac{\log\log n}{\log a}}.$$

Отсюда

$$F \geqslant \psi(n, a) \geqslant n^{1 - \frac{\log\log n}{\log a}} \geqslant n^{1 - \frac{\log\log n}{2\log\log n}} = \sqrt{n},$$

так как из $a > \log^2 n$ следует $\log a > 2\log\log n$. Полученное противоречие завершает наши рассуждения.

§ 1.6. Алгоритм Миллера

В работе [187] приведен алгоритм, который детерминированно проверяет простоту n за $O(n^{1/7})$ арифметических операций. Тот же алгоритм можно модифицировать так, что он будет делать $O(\log^4 n)$ арифметических операций; однако в этом случае его корректность опирается на справедливость расширенной гипотезы Римана. Эта гипотеза гласит, что если $\chi(a)$ — числовой характер по модулю m, то нули L-функции Дирихле

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

в полосе $0 < \operatorname{Re} s < 1$ лежат на прямой $\operatorname{Re} s = 1/2$.

Пусть $f\colon \mathbb{N} \to \mathbb{R}_{>0}$ — некоторая функция на множестве натуральных чисел, причем $f(n) < n$.

Алгоритм Миллера A_f.

На входе задано нечетное число n, $n > 1$.

1 шаг. Проверить, выполняется ли равенство $n = m^s$ при некоторых s, $m \in \mathbb{N}$, $s \geqslant 2$. Если выполняется, то n — составное число, и алгоритм останавливается.

2 шаг. Выполнить шаги (i)—(iii) для всех $a \leqslant f(n)$.

(i) Проверить условие $a \mid n$.

(ii) Проверить условие $a^{n-1} \not\equiv 1 \pmod{n}$.

(iii) Выяснить, верно ли, что при некотором k, $1 \leqslant k \leqslant \nu_2(n-1)$,

$$1 < \text{НОД}\Big(a^{\frac{n-1}{2^k}} - 1 \pmod n,\ n\Big) < n.$$

Если одно из условий (i)—(iii) выполнено, то n — составное, и алгоритм останавливается.

3 шаг. Если мы дошли до этого шага, то n — простое число.

Конец алгоритма.

Теорема 1.37. *Если $f(n) = c \cdot n^{0{,}133}$ (где c — некоторая положительная постоянная), то алгоритм детерминированно проверяет простоту n за $O(n^{1/7})$ арифметических операций. Если же $f(n) = c \log^2 n$, то алгоритм детерминированно проверяет простоту n за $O(\log^4 n)$ операций в предположении справедливости расширенной гипотезы Римана.*

Замечание 1.38. В работе [64] показано, что проверку 1 шага можно выполнить за $(\log n)^{1+o(1)}$ арифметических операций.

Мы докажем вторую часть этой теоремы, следуя работе [187]. Мы будем считать, что $f(n) = c \log^2 n$ для некоторой достаточно большой абсолютной постоянной c, и что n не является степенью, т. е. $n \neq m^s$, $m,\ s \in \mathbb{N}$, $s > 1$. Значение $c = 2$ было получено в более поздних работах, см. для справок [60, гл. 9; 101].

Пусть n — нечетное составное число, $n > 1$, $n = p_1^{v_1} \dots p_u^{v_u}$ — разложение n на простые сомножители. Везде далее используем это обозначение. Тогда $u \geqslant 2$.

Определение 1.39. *Функция Кармайкла* определяется равенством

$$\lambda(n) = \operatorname*{НОК}_i (p_i^{v_i - 1}(p_i - 1));$$

функция $\lambda'(n)$ определяется равенством $\lambda'(n) = \operatorname*{НОК}_i (p_i - 1)$.

Лемма 1.40. *Нечетное $n \in \mathbb{N}$ удовлетворяет малой теореме Ферма $a^n \equiv a \pmod n$ для всех $a \in \mathbb{N}$, $(a, n) = 1$, тогда и только тогда, когда $\lambda(n) \mid n - 1$.*

Доказательство. Сравнение $a^n \equiv a \pmod n$ при $(a, n) = 1$ равносильно тому, что выполнена система сравнений

$$a^{n-1} \equiv 1 \pmod{p_j^{v_j}}, \quad j = 1, \dots, k.$$

Поскольку найдется $a_j \in \mathbb{N}$ такое, что $a_j \pmod{p_j^{v_j}}$ является первообразным корнем (т. е. имеет порядок $p_j^{v_j - 1}(p_j - 1)$), и при этом $a_j \equiv 1 \pmod{p_l^{v_l}}$ при $l \neq j$, то наша система сравнений равносильна тому, что $\varphi(p_j^{\alpha_j}) \mid n - 1$, $j = 1, \dots, u$, что равносильно $\lambda(n) \mid n - 1$. $\square$

Лемма 1.41. *Если* $\lambda'(n) \nmid n-1$, *то найдутся простые числа* p, q *такие, что*

1) $p \mid n$, $p-1 \nmid n-1$ *и при некотором* $m \geqslant 1$ $q^m \mid p-1$, $q^m \nmid n-1$;

2) *если для этих* p *и* q *число* a *является невычетом* q*-й степени по модулю* p (*т. е. уравнение* $x^q \equiv a \pmod p$ *неразрешимо*), *то* $a^{n-1} \not\equiv 1 \pmod n$.

Доказательство. 1) По условию среди чисел p_i найдется число p, такое, что $p-1 \nmid n-1$. Это, в свою очередь, означает, что найдутся простое q и натуральное m, для которых $q^m \mid p-1$, $q^m \nmid n-1$.

2) Если $a^{n-1} \equiv 1 \pmod n$, то $a^{n-1} \equiv 1 \pmod p$. Пусть b — первообразный корень по модулю p, $a \equiv b^{\operatorname{ind} a} \pmod p$; тогда $b^{(n-1)\operatorname{ind} a} \equiv 1 \pmod p$. Отсюда $p-1 \mid (\operatorname{ind} a)(n-1)$. Тогда $q^m \mid (\operatorname{ind} a)(n-1)$, откуда $q \mid \operatorname{ind} a$, что противоречит тому, что a невычет q-й степени. □

Определение 1.42. Обозначим через $N(p, q)$ наименьшее натуральное число a такое, что $(a, p) = 1$ и a есть невычет q-й степени по модулю p. Число $N(p, q)$ определено лишь при $q \mid p-1$.

Теорема 1.43 (см. [54]). *При условии выполнения расширенной гипотезы Римана*

$$N(p, q) = O(\log^2 p).$$

Следствие 1.44. *Если* $\lambda'(n) \nmid n-1$ *и выполняется расширенная гипотеза Римана, то случай* (ii) 2 *шага алгоритма обнаружит, что* n *составное. Действительно, достаточно взять в качестве* a *число* $N(p, q) \leqslant c \log^2 p \leqslant c \log^2 n$; *по лемме* 1.41 $a^{n-1} \not\equiv 1 \pmod n$.

Фактически мы обосновали алгоритм Миллера и доказали теорему в случае $\lambda'(n) \nmid n-1$. Далее считаем, что $\lambda'(n) \mid n-1$.

Определение 1.45. Скажем, что число n имеет тип **A**, если найдется такой номер j, что

$$\nu_2(\lambda'(n)) > \nu_2(p_j - 1).$$

Иначе n имеет тип **B**, т. е. в этом случае для любого j, $1 \leqslant j \leqslant u$, $\nu_2(\lambda'(n)) = \nu_2(p_j - 1)$.

Лемма 1.46. *Пусть* n — *составное типа* **A**, *простые* p, q *делят* n, *причем*

$$\nu_2(\lambda'(n)) = \nu_2(p-1) > \nu_2(q-1).$$

Пусть $1 < a < n$, $\left(\dfrac{a}{p}\right) = -1$. *Тогда либо* a, *либо* $(a^{\lambda'(n)/2} - 1) \pmod n$ *имеет нетривиальный наибольший общий делитель с* n (*т. е. этот делитель отличен от* 1 *и* n).

Доказательство. Заметим, что $\nu_2(\lambda'(n)) \geqslant 2$, поскольку $\nu_2(q-1) \geqslant$ $\geqslant 1$. Пусть $(a, n) = 1$. Так как $q - 1 \,\Big|\, \dfrac{\lambda'(n)}{2}$, то $a^{\lambda'(n)/2} \equiv 1 \pmod q$. Кроме того,

$$a^{\lambda'(n)/2} \equiv \pm 1 \pmod p.$$

Если $a^{\lambda'(n)/2} \equiv 1 \pmod p$, то $(\operatorname{ind} a) \cdot \lambda'(n)/2 \equiv 0 \pmod{p-1}$, где $\operatorname{ind} a$ — индекс $a \pmod p$ по отношению к какому-либо первообразному корню в $\mathbb{Z}/p\mathbb{Z}$. Так как $\nu_2(\lambda'(n)) = \nu_2(p-1)$, то $\operatorname{ind} a$ — четен; значит, $\left(\dfrac{a}{p}\right) = 1$, что противоречит условию. Итак,

$$a^{\lambda'(n)/2} \equiv 1 \pmod q, \qquad a^{\lambda'(n)/2} \equiv -1 \pmod p.$$

Поэтому $(a^{\lambda'(n)/2} - 1, n)$ делится на q и не делится на p, что и требовалось доказать. □

Лемма 1.47. *Пусть n — составное число типа* **B**, *простые числа p и q делят n, $p \neq q$, и число a удовлетворяет условиям*

$$1 < a < n, \qquad \left(\frac{a}{pq}\right) = -1.$$

Тогда либо a, либо $(a^{\lambda'(n)/2} - 1) \pmod n$ имеет с n нетривиальный наибольший общий делитель.

Доказательство. Пусть $(a, n) = 1$, и предположим, не ограничивая общности, что $\left(\dfrac{a}{p}\right) = -1$, $\left(\dfrac{a}{q}\right) = 1$. Так как n типа **B**, то $\nu_2(p-1) =$ $= \nu_2(q-1) = \nu_2(\lambda'(n))$. Рассуждая аналогично доказательству леммы 1.46, получим

$$a^{\lambda'(n)/2} \equiv 1 \pmod q, \qquad a^{\lambda'(n)/2} \equiv -1 \pmod p.$$

Отсюда НОД$((a^{\lambda'(n)/2} - 1) \pmod n, n)$ делится на q и не делится на p. Лемма доказана. □

Теперь нужно обеспечить проверку утверждений лемм 1.46, 1.47, не зная $\lambda'(n)$.

Лемма 1.48. *Пусть p — простое число, $p \mid n$, $\lambda'(n) \mid n - 1$,*

$$k = \nu_2\left(\frac{n-1}{\lambda'(n)}\right) + 1.$$

Пусть $a \in \mathbb{N}$, $1 < a < n$, $(a, n) = 1$. Тогда

$$a^{\lambda'(n)/2} \equiv a^{(n-1)/2^k} \pmod p.$$

Замечание 1.49. Поскольку $\lambda'(n)$ четно, то $1 \leqslant k \leqslant \nu_2(n-1)$.

Доказательство леммы 1.48. Поскольку $a^{\lambda'(n)} \equiv 1 \pmod p$, то $a^{\lambda'(n)/2} \equiv \pm 1 \pmod p$.

1) Пусть $a^{\lambda'(n)/2} \equiv 1 \pmod p$. У нас $\lambda'(n) \mid n-1$, и по определению k

$$\frac{\lambda'(n)}{2} \,\Big|\, \frac{n-1}{2^k}.$$

Значит, утверждение леммы в этом случае выполнено.

2) Пусть $a^{\lambda'(n)/2} \equiv -1 \pmod p$. Но тогда

$$a^{(n-1)/2^k} = (a^{\lambda'(n)/2})^{(n-1)/(\lambda'(n)2^{k-1})} \equiv (-1)^{(n-1)/(\lambda'(n)2^{k-1})} \pmod p.$$

При этом $\nu_2\Big(\dfrac{n-1}{\lambda'(n)2^{k-1}}\Big) = 0$ по определению k. Поэтому $a^{(n-1)/2^k} \equiv$ $\equiv -1 \pmod p$. $\square$

Следствие 1.50. *Пусть* $\lambda'(n) \mid n-1$, *и пусть* n *имеет тип* **A**. *Если* $a \in \mathbb{N}$, $1 < a < n$, *и* $\left(\dfrac{a}{p}\right) = -1$, *то либо* $(a, n) \neq 1$, n, *либо при некотором* k, $1 \leqslant k \leqslant \nu_2(n-1)$,

$$\text{НОД}\Big((a^{\frac{n-1}{2^k}} - 1) \pmod n,\, n\Big) \neq 1,\, n.$$

Поэтому, если в ходе перебора в алгоритме Миллера мы дойдем до этого значения a, то в пункте (iii) 2 шага алгоритма будет обнаружено, что n — составное. Очевидно, что наименьшее такое $a = N(p, 2) = O(\log^2 p) \leqslant c \log^2 n$ (по теореме 1.43, сформулированной выше) содержится среди перебираемых в алгоритме значений при достаточно большом c, т. е. алгоритм корректно работает для n типа **A**.

Следствие 1.51. *Пусть* $\lambda'(n) \mid n-1$, *и пусть* n *имеет тип* **B**. *Если* $a \in \mathbb{N}$, $1 < a < n$, *и* $\left(\dfrac{a}{pq}\right) = -1$, *то либо* $(a, n) \neq 1$, n, *либо при некотором* k, $1 \leqslant k \leqslant \nu_2(n-1)$,

$$\text{НОД}\Big((a^{\frac{n-1}{2^k}} - 1) \pmod n,\, n\Big) \neq 1,\, n.$$

Поэтому для доказательства корректности алгоритма Миллера для n типа **B** нужна оценка сверху для величины

$$1(pq) = \min\Big\{a \;\Big|\; a \in \mathbb{N},\ \left(\frac{a}{pq}\right) = -1\Big\}.$$

Теорема 1.52 (см. [54]). *При условии выполнения расширенной гипотезы Римана*

$$N(pq) = O(\log^2 pq).$$

Следовательно, в алгоритме Миллера для n типа **B** значение $a = N(pq) \leqslant c \log^2 n$ будет найдено, и в пункте (iii) 2 шага будет обнаружено, что n — составное.

Это рассуждение завершает обоснование алгоритма Миллера и доказательство теоремы.

§ 1.7. Вероятностные тесты на простоту

Пусть $n \in \mathbb{N}$, n нечетно, $n > 1$. Вероятностный тест на простоту проводится следующим образом. Выбирается случайное $a \in \mathbb{N}$, $1 \leqslant a < n$, и для него проверяется выполнение некоторых условий. Если какое-то из условий не выполнено, то число n — составное, поскольку для простых чисел эти условия являются необходимыми. Если же все условия выполнены, то из этого еще не следует простота n. Однако можно будет считать, что «n — простое число с некоторой вероятностью». Кроме того, обычно доказывают оценку снизу для этой вероятности. Чем больше значений a мы протестируем, тем ближе эта вероятность к единице.

Рассмотрим тест Соловея—Штрассена [262].

Теорема 1.53. *Пусть n — нечетное составное число. Тогда количество целых чисел a, $0 \leqslant a \leqslant n-1$, удовлетворяющих условиям*

1) $(a, n) = 1$,

2) $a^{(n-1)/2} \equiv \left(\dfrac{a}{n}\right) \pmod{n}$,

не превышает $n/2$.

Следствие 1.54. *Если n — простое, то условия 1 и 2 теоремы, очевидно, выполняются для всех a, $1 \leqslant a \leqslant n-1$. Если же n — составное, то для случайно выбранного a из промежутка $0 \leqslant a \leqslant n-1$ вероятность выполнения обоих условий теоремы не превосходит $1/2$. Поэтому, если для k случайных значений a мы проверим выполнение условий теоремы и не обнаружим, что n — составное, то будем считать, что n — простое с вероятностью, не меньшей чем $1 - 1/2^k$.*

Доказательство теоремы 1.53. Покажем сперва, что существует $b \in \mathbb{N}$, для которого $(b, n) = 1$ и $b^{\frac{n-1}{2}} \not\equiv \left(\dfrac{b}{n}\right) \pmod{n}$. Пусть $n = p_1^{\alpha_1} \dots p_k^{\alpha_k}$ — разложение n на простые сомножители.

Если n делится на квадрат простого числа, то найдется $b \in \mathbb{N}$, $(b, n) = 1$, такое, что $b^{n-1} \not\equiv 1 \pmod{n}$, откуда $b^{\frac{n-1}{2}} \not\equiv \pm 1 \pmod{n}$. Действительно, фиксируем номер i такой, что $\alpha_i \geqslant 2$. По китайской теореме об остатках можно найти $b \in \mathbb{N}$, для которого $b \pmod{p_i^{\alpha_i}}$ является первообразным корнем в $\mathbb{Z}/p_i^{\alpha_i}\mathbb{Z}$, а при $j \neq i$ $b \equiv 1 \pmod{p_j^{\alpha_j}}$.

Если $b^{n-1} \equiv 1 \pmod{n}$, то $b^{n-1} \equiv 1 \pmod{p_i^{\alpha_i}}$, откуда $n-1 \vdots \varphi(p_i^{\alpha_i}) = p_i^{\alpha_i - 1}(p_i - 1)$, что невозможно, так как $n-1$ не делится на p_i.

Предположим теперь, что n бесквадратно, $n = p_1 \dots p_k$. Найдем такое $b \in \mathbb{N}$, что $b \pmod{p_1}$ — первообразный корень в $\mathbb{Z}/p_1\mathbb{Z}$, и $b \equiv 1 \pmod{p_j}$ при $j > 1$. Тогда $(b, n) = 1$ и

$$\left(\frac{b}{n}\right) = \left(\frac{b}{p_1}\right) \dots \left(\frac{b}{p_k}\right) = \left(\frac{b}{p_1}\right) = -1.$$

Сравнение $b^{\frac{n-1}{2}} \equiv -1 \pmod{n}$ равносильно сравнению $b^{\frac{n-1}{2}} \equiv -1 \pmod{p_j}$ для $j = 1, \dots, k$. Так как $k \geqslant 2$, то $1 \equiv b^{\frac{n-1}{2}} \equiv -1 \pmod{p_2}$, что невозможно.

Итак, число b существует. Зафиксируем его и рассмотрим два множества:

$$W_1 = \left\{ a \mid 1 \leqslant a \leqslant n-1,\ (a, n) = 1,\ a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right) \pmod{n} \right\},$$

$$W_2 = \left\{ a \mid 1 \leqslant a \leqslant n-1,\ (a, n) = 1,\ a^{\frac{n-1}{2}} \not\equiv \left(\frac{a}{n}\right) \pmod{n} \right\},$$

Если $a_1 \in W_1$, $a_2 \in W_2$, то $a_1 a_2 \in W_2$, поскольку $\left(\frac{a_1 a_2}{n}\right) = \left(\frac{a_1}{n}\right)\left(\frac{a_2}{n}\right)$. Поэтому для каждого $a \in W_1$ наименьший неотрицательный вычет $ba \pmod{n}$ принадлежит W_2. Следовательно, $|W_2| \geqslant |W_1|$, откуда вытекает утверждение теоремы. □

Теперь рассмотрим более эффективный тест Миллера—Рабина (см. [230; 20]).

Теорема 1.55. *Пусть n — нечетное составное число. Пусть $3 \nmid n$, $n - 1 = 2^r t$, где $r \geqslant 1$, t — нечетно. Тогда количество таких чисел a, $0 < a \leqslant n-1$, что либо $a^t \equiv 1 \pmod{n}$, либо при некотором j, $1 \leqslant j \leqslant r$,*

$$a^{(n-1)/2^j} \equiv -1 \pmod{n},$$

не превосходит $n/4$.

Следствие 1.56. *Из теоремы 1.55 аналогично следствию 1.54 получаем вероятностный тест на простоту. При этом, если для k случайных значений a мы не обнаружим, что n — составное, то будем считать, что n — простое с вероятностью, не меньшей чем $1 - \frac{1}{4^k}$.*

Замечание 1.57. Если n — простое, то $\mathbb{Z}/n\mathbb{Z}$ — поле, $a^{n-1} \equiv 1 \pmod{n}$ для всех a, $1 \leqslant a \leqslant n-1$. Так как уравнение $x^2 \equiv 1 \pmod{n}$

имеет в $\mathbb{Z}/n\mathbb{Z}$ ровно два решения ± 1, то сравнения теоремы 1.55 выполнены для всех a, $1 \leqslant a \leqslant n-1$.

Замечание 1.58. Тест Миллера—Рабина всегда сильнее теста Соловея—Штрассена, как показано в работе [190]. Точнее, если при фиксированном n число a проходит тест Миллера—Рабина и не показывает, что n составное, то оно проходит тест Соловея—Штрассена с тем же результатом.

Замечание 1.59. В работе [13] показано, что некоторый аналог алгоритма Миллера—Рабина может быть применен для проверки простоты главных идеалов в круговых полях. В работе [17] показана возможность применения этого алгоритма в некоторых криптосистемах типа RSA.

Для доказательства теоремы 1.55 нам потребуется ряд лемм. Обозначим через S множество $a \pmod n$, $1 \leqslant a \leqslant n$, таких, что либо $a^t \equiv 1 \pmod n$, либо при некотором j, $1 \leqslant j \leqslant r$, $a^{(n-1)/2^j} \equiv -1 \pmod n$. Мы считаем далее, что n — нечетное, составное, не делящееся на 3 число.

Лемма 1.60. *Если существует простое число p такое, что $p^2 \mid n$, то множество*

$$G = \left\{ 1 + k\frac{n}{p} \pmod n \;\middle|\; k = 0, \ldots, p-1 \right\}$$

является подгруппой $(\mathbb{Z}/n\mathbb{Z})^*$ порядка p.

Доказательство. Из сравнений

$$\left(1 + \frac{kn}{p}\right)\left(1 + \frac{k'n}{p}\right) \equiv 1 + ((k+k') \pmod p)\frac{n}{p} \pmod n$$

и

$$\left(1 + \frac{kn}{p}\right)^l \equiv 1 + (kl \pmod p)\frac{n}{p} \pmod n$$

легко следует утверждение леммы. □

Определение 1.61. Обозначим A множество элементов $a \in (\mathbb{Z}/n\mathbb{Z})^*$, для которых выполнено одно из следующих двух условий:

1) $a^{n-1} \not\equiv 1 \pmod n$;

2) $a^k \not\equiv -1 \pmod n$ для любого $k \in \mathbb{Z}$, и для некоторого простого p, $p \mid n$, порядок $a \pmod p$ равен $p-1$.

Лемма 1.62. *Пусть $a \in A$, $s \in S$. Тогда $as \notin S$, т. е. $aS \cap S = \varnothing$.*

Доказательство. Если $a^{n-1} \not\equiv 1 \pmod n$, то поскольку $s^{n-1} \equiv 1 \pmod n$, получим $(as)^{n-1} \not\equiv 1 \pmod n$, т. е. $as \notin S$.

Пусть далее a не удовлетворяет первому, но удовлетворяет второму условию в определении множества A. Тогда $a^{n-1} \equiv 1 \pmod{n}$ и для любого целого k выполнено сравнение $a^k \not\equiv -1 \pmod{n}$. Также фиксируем простое число p, $p \mid n$, для которого порядок $a \pmod{p}$ равен $p-1$. Зафиксируем число i такое, что

$$a^{(n-1)/2^i} \equiv 1 \pmod{n}.$$

Такие i существуют, например, $i = 0$. Кроме того, поскольку $p \mid n$, то

$$a^{(n-1)/2^i} \equiv 1 \pmod{p}.$$

По условию тогда $\dfrac{n-1}{2^i} \vdots p-1$, откуда в силу четности $p-1$ получаем неравенство $0 \leqslant i < r$. В частности, отсюда следует,что

$$a^{(n-1)/2^r} = a^t \not\equiv 1 \pmod{n}.$$

Покажем, что если $s \in S$, то при всех j таких, что $0 \leqslant j \leqslant i < r$, выполнено сравнение

$$s^{(n-1)/2^j} \equiv 1 \pmod{n}. \tag{1.1}$$

Если $s \in S$ и $s^{(n-1)/2^r} \equiv 1 \pmod{n}$, то сравнение (1.1) выполнено. Предположим теперь, что $s \in S$ и для некоторого j_1, $0 \leqslant j_1 \leqslant r$,

$$s^{(n-1)/2^{j_1}} \equiv -1 \pmod{n}.$$

В случае $j_1 > i$ отсюда следует, что при всех j, $0 \leqslant j \leqslant i$, выполнено сравнение

$$s^{(n-1)/2^j} \equiv 1 \pmod{n},$$

т. е. формула (1.1) верна.

Рассмотрим случай $j_1 \leqslant i$ и придем к противоречию. Поскольку

$$s^{(n-1)/2^{j_1}} \equiv -1 \pmod{n},$$

то

$$s^{(n-1)/2^{j_1}} \equiv -1 \pmod{p}.$$

Кроме того, по предположению $j_1 \leqslant i$, откуда по доказанному выше

$$p-1 \mid \frac{n-1}{2^i} \mid \frac{n-1}{2^{j_1}}.$$

Из малой теоремы Ферма тогда следует, что

$$s^{(n-1)/2^{j_1}} \equiv 1 \pmod{p}.$$

Это и есть противоречие, так как $1 \not\equiv -1 \pmod{p}$.

Итак, формула (1.1) верна. Далее, из (1.1) следует, что

$$s^{(n-1)/2^{i+1}} \equiv \pm 1 \pmod{n},$$

поскольку $s \in S$.

Теперь выберем i максимальным. Тогда, поскольку по доказанному $i < r$, имеем

$$a^{(n-1)/2^i} \equiv 1 \pmod{n},$$
$$a^{(n-1)/2^{i+1}} \not\equiv \pm 1 \pmod{n}.$$

(Мы воспользовались тем, что $a^k \not\equiv -1 \pmod{n}$ для всех $k \in \mathbb{Z}$.) Тогда при всех j, $0 \leqslant j \leqslant i$, выполнено сравнение

$$(as)^{(n-1)/2^j} \equiv 1 \pmod{n},$$

но

$$(as)^{(n-1)/2^{i+1}} \equiv \pm a^{(n-1)/2^{i+1}} \not\equiv \pm 1 \pmod{n},$$

Отсюда следует, что $as \notin S$. □

Лемма 1.63. *Пусть a, $b \in (\mathbb{Z}/n\mathbb{Z})^*$, $a \neq b$. Множества aS и bS не пересекаются тогда и только тогда, когда не пересекаются множества $ab^{-1}S$ и S.*

Доказательство очевидно.

Следствие 1.64. *Пусть G — подгруппа $(\mathbb{Z}/n\mathbb{Z})^*$. Множества g_1S и g_2S не пересекаются при всех g_1, $g_2 \in S$, $g_1 \neq g_2$, тогда и только тогда, когда не пересекаются множества S и gS для всех $g \in G$, $g \neq 1$.*

Лемма 1.65. *Пусть n составное и n делится на p^2, где p — простое число. Тогда*

$$|S| \leqslant \frac{1}{4}|(\mathbb{Z}/n\mathbb{Z})^*|.$$

Доказательство. Пусть G — подгруппа $(\mathbb{Z}/n\mathbb{Z})^*$ из леммы 1.60. Поскольку $p \mid n$, то $p \nmid n-1$, и тогда для любого $g \in G$, $g \neq 1$, выполнено сравнение $g^{n-1} \not\equiv 1 \pmod{n}$. Поэтому $g \in A$ и по лемме 1.62

множества S и gS не пересекаются. Значит, по следствию леммы 1.63 множества gS, $g \in G$, попарно не пересекаются. Поэтому $\left|\bigcup\limits_{g\in G} Sg\right| = |G|\cdot|S| = p|S|$, и $p|S| \leqslant |(\mathbb{Z}/n\mathbb{Z})^*| = \varphi(n)$, откуда

$$|S| \leqslant \frac{\varphi(n)}{p} \leqslant \frac{1}{4}|(\mathbb{Z}/n\mathbb{Z})^*|,$$

так как $p \geqslant 5$ по условию теоремы. Лемма 1.65 доказана. □

Лемма 1.66. *Пусть $n = p_1p_2$, где p_1 и p_2 — различные простые числа. Тогда $n-1$ не делится на одно из двух чисел $p_i - 1$.*

Доказательство следует из равенства

$$n - 1 = p_1p_2 - 1 = (p_1 - 1)(p_2 - 1) + (p_1 - 1) + (p_2 - 1).$$

Лемма 1.67. *Пусть $n = p_1p_2$, где $p_1 \neq p_2$. Тогда $|S| \leqslant \varphi(n)/4$.*

Доказательство. По китайской теореме об остатках найдутся числа a_1 и a_2 такие, что $a_i \equiv 1 \pmod{p_{3-i}}$, и $a_i \pmod{p_i}$ — первообразный корень по модулю p_i для $i = 1, 2$. Тогда $a_i^k \not\equiv -1 \pmod{p_{3-i}}$ для любого $k \in \mathbb{Z}$; кроме того, $a_i^k \equiv 1 \pmod{p_i}$ тогда и только тогда, когда $p_i - 1 \mid k$. Это значит, что $a_i \in A$. Очевидно также, что $a_i^{-1} \pmod{n} \in A$. Далее, для элемента $a \equiv a_1a_2 \pmod{n}$ сравнение $a^k \equiv 1 \pmod{n}$ выполнено тогда и только тогда, когда $a_i^k \equiv 1 \pmod{p_i}$ для $i = 1, 2$, что равносильно условию $p_i - 1 \mid k$. Отсюда по лемме 1.66 получаем, что $a^{n-1} \not\equiv 1 \pmod{n}$, т. е. $a = a_1a_2 \in A$. Аналогично $a_1a_2^{-1} \notin A$.

Рассмотрим теперь множества S, Sa_1, Sa_2, Sa. По леммам 1.62 и 1.63 они попарно не пересекаются. Кроме того, S, Sa_1, Sa_2, Sa содержатся в $(\mathbb{Z}/n\mathbb{Z})^*$ и состоят из одинакового количества элементов. Поэтому $|S| \leqslant \frac{1}{4}|(\mathbb{Z}/n\mathbb{Z})^*|$. □

Лемма 1.68. *Пусть n бесквадратно и делится на три различных простых числа p_1, p_2, p_3. Тогда $|S| \leqslant \varphi(n)/4$.*

Доказательство. Как и при доказательстве леммы 1.67, найдутся $a_1, a_2 \in (\mathbb{Z}/n\mathbb{Z})^*$ такие, что $a_i \equiv 1 \pmod{\frac{n}{p_i}}$, $a_i \pmod{p_i}$ — первообразный корень по модулю p_i, $i = 1, 2$. Тогда $a_i \equiv 1 \pmod{p_3}$ для $i = 1, 2$; $a_1a_2 = a \equiv 1 \pmod{p_3}$, $b \equiv a_1a_2^{-1} \equiv 1 \pmod{p_3}$. Кроме того $a_i^k \not\equiv -1 \pmod{n}$ для любого $k \in \mathbb{Z}$ (так как $1 \not\equiv -1 \pmod{p_3}$). Также $a^k \not\equiv -1 \pmod{n}$ при $a = a_1a_2$ и $b^k \not\equiv -1 \pmod{n}$ при $b = a_1a_2^{-1}$. Значит, $a_1, a_2, a, b \in A$ (требование о порядке элемента во втором условии из определения множества A выполнено по построению a_1 и a_2). По леммам 1.62 и 1.63 множества S, Sa_1, Sa_2, Sa попарно

не пересекаются и равномощны. Аналогично доказательству леммы 1.67 получим неравенство $|S| \leqslant \frac{1}{4}|(\mathbb{Z}/n\mathbb{Z})^*| = \varphi(n)/4$. Лемма 1.68 доказана. □

Доказательство теоремы 1.55 очевидным образом получается из лемм 1.65, 1.67, 1.68.

§ 1.8. Современные методы проверки простоты чисел

В начале 80-х годов Адлеман, Померанс и Румели [46] предложили детерминированный алгоритм проверки простоты чисел. Для заданного натурального числа n алгоритм делает $O((\log n)^{c \log\log\log n})$ арифметических операций (c — некоторая абсолютная постоянная) и выдает верный ответ, составное n или простое. Описание схемы алгоритма можно найти также в [10]. Этот алгоритм оказался непрактичным и довольно сложным для реализации на компьютере.

Существенные теоретические упрощения алгоритма Адлемана—Померанса—Румели были получены Х. Ленстрой [164]. Он предложил детерминированный алгоритм, также делающий $O((\log n)^{c \log\log\log n})$ арифметических операций. Реализация этого алгоритма позволила проверять на простоту числа n порядка 10^{100} за несколько минут.

Замечание 1.69. Оценка сложности в алгоритмах Адлемана—Померанса—Румели и Ленстры является неулучшаемой, т. е. для простого числа n алгоритм выполнит не менее $c_1(\log n)^{c_2 \log\log\log n}$ операций для некоторых положительных постоянных c_1 и c_2.

Приведем упрощенную и несколько модифицированную схему алгоритма Ленстры для того, чтобы читатель мог получить представление о его внутренней структуре и используемых им средствах.

Схема алгоритма Ленстры.

Алгоритм проверяет на простоту нечетное число $n \in \mathbb{N}$, $n > 1$.

1 шаг. Выбираем различные простые числа $p_1, \ldots, p_k$ (называемые начальными простыми) так, чтобы нашлись нечетные простые числа $q_1, \ldots, q_s$ (называемые евклидовыми простыми), удовлетворяющие следующим условиям:

а) $q_j - 1 \mid p_1 \ldots p_k$, $j = 1, \ldots, s$;

б) $2q_1 \ldots q_s \geqslant \sqrt{n}$.

Пример 1.70. $\{p\} = \{2, 3, 5, 7\}$, $\{q\} = \{3, 7, 11, 31, 43, 71, 211\}$, $2q_1 \ldots q_s \geqslant 143 \cdot 10^9 > 10^{11}$. Следовательно, данные наборы $\{p_i\}$, $\{q_j\}$ можно использовать для проверки простоты всех чисел n, $n \leqslant 10^{22}$.

2 шаг. Проверяем, верно ли, что $n = p_i$ или $n = q_j$ для некоторого i или j. Если да, то n — простое. Иначе проверяем равенство

$$\text{НОД}(p_1 \dots p_k q_1 \dots q_s,\, n) = 1.$$

Если оно неверно, то n — составное.

3 шаг. Для каждой пары p, q, такой, что $p \mid q-1$, находим первообразный корень c_q по модулю q и числа a, $b \in \mathbb{N}$, которые при $p > 2$ удовлетворяют следующим условиям:

$$ab(a+b) \not\equiv 0 \pmod{p},$$
$$a^p + b^p \not\equiv (a+b)^p \pmod{p^2}.$$

Известно, что такие a, b существуют, и обычно $a = b = 1$. Далее определяем числовой характер $\chi_{p,q}$ по модулю q порядка p:

$$\chi_{p,q} \colon (\mathbb{Z}/q\mathbb{Z})^* \to \mathbb{C}, \qquad \chi_{p,q}(x) = \zeta_p^{\operatorname{ind}_q x},$$

где $\zeta_p = e^{2\pi i/p}$, $\operatorname{ind}_q(x) \in \mathbb{Z}/(q-1)\mathbb{Z}$, $c_q^{\operatorname{ind}_q x} \equiv x \pmod{q}$. Эти характеры при различных $p \mid q-1$ порождают всю группу числовых характеров по модулю q.

Вычисляем сумму Якоби

$$\nu(\chi_{p,q}) = -\sum_{x=2}^{q-1} \chi_{p,q}(x)^a \chi_{p,q}(1-x)^b = -\sum_{x=2}^{q-1} \zeta_p^{a \operatorname{ind}_q(x) + b \operatorname{ind}_q(1-x)}.$$

Это вычисление проводится быстро, поскольку евклидовы простые числа q невелики, и значения $\operatorname{ind}_q x$ быстро определяются перебором.

4 шаг. Для каждого начального простого числа p находим наибольшее натуральное число $h = h(p)$, $1 \leqslant h \leqslant t = \nu_p(n^{p-1} - 1)$, такое, что для всех q таких, что $p \mid q-1$, выполнено сравнение

$$\nu(\chi_{p,q})^{\alpha_h(n)} \equiv \xi_{p,q} \pmod{n\mathbb{Z}[\zeta_p]}. \tag{1.2}$$

Это основной тест алгоритма. Здесь $\xi_{p,q}$ — некоторый корень степени p из 1; $\alpha_h(n)$ — некоторый явно определяемый по n элемент группового кольца $\mathbb{Z}[\operatorname{Gal}(\mathbb{Q}(\zeta_p))]$, имеющий вид $\alpha_h(n) = \sum\limits_j a_{h,j}\sigma_j$, где $a_{h,j} \in \mathbb{Z}_{\geqslant 0}$, $\sigma_j \in \operatorname{Gal}(\mathbb{Q}(\zeta_p))$, $\sigma_j(\zeta_p) = \zeta_p^j$, $1 \leqslant j \leqslant p-1$. При этом, если

$$\nu(\chi_{p,q}) = \sum_{l=0}^{p-2} A_l \zeta_p^l, \qquad A_l \in \mathbb{Z},$$

то

$$\nu(\chi_{p,q})^{\alpha_h(n)} = \prod_j \left(\sum_l A_l \zeta_p^{jl} \right)^{a_{h,j}} = \sum_{l=0}^{p-2} B_l \zeta_p^l,$$

где $B_l \in \mathbb{Z}$. Поскольку $1, \zeta_p, \ldots, \zeta_p^{p-2}$ есть $\mathbb{Z}$-базис $\mathbb{Z}$-модуля $\mathbb{Z}[\zeta_p]$, мы работаем с $(p-1)$-мерными векторами, имеющими целочисленные координаты. Сравнение (1.2) означает, что координаты двух таких векторов сравнимы по модулю n.

Если сравнение (1.2) не выполнено при некоторых p, q с $h = 1$ и

$$\alpha_1(n) = \sum_{j=1}^{p-1} \left[\frac{nj}{p} \right] \sigma_{j^{-1} \pmod p},$$

то n — составное число (аналогом этого теста является невыполнение малой теоремы Ферма $a^{n-1} \equiv 1 \pmod n$).

5 шаг. Для тех p, у которых $h = h(p) < t = t(p)$ и при этом $\xi_{p,q} = 1$ для всех q таких, что $p \mid q-1$, далее проверяем следующее условие:

найдется евклидово простое q, для которого $p \mid q-1$ и при всех $j = 0, 1, \ldots, p-1$ элемент

$$\nu(\chi_{p,q})^{\alpha_{h+1}(n)} - \zeta_p^j \in \mathbb{Z}[\zeta_p],$$

представленный как вектор в базисе $1, \zeta_p, \ldots, \zeta_p^{p-2}$, имеет коэффициент, взаимно простой с n.

Если это условие не выполнено, то можно показать, что на этом шаге будет найден нетривиальный делитель n.

6 шаг. Найденные на 4-м шаге числа $\xi_{p,q}$ представим в виде $\xi_{p,q} = \zeta_p^{u_{p,q}}$, $u_{p,q} \in \mathbb{Z}_{\geqslant 0}$. Затем для каждого q найдем x_q такое, что для каждого p, $p \mid q-1$, справедливо сравнение

$$-n\psi_p(\beta) x_q \equiv u_{p,q} \pmod p.$$

Здесь

$$\psi_p(\beta) = \sum_{j=1}^{p-1} \left(\left[\frac{(a+b)j}{p} \right] - \left[\frac{aj}{p} \right] - \left[\frac{bj}{p} \right] \right) j^{-1} \pmod p$$

— целое число, зависящее от p. Нахождение x_q проводится с помощью китайской теоремы об остатках.

Далее находим $v \in \mathbb{Z}$, $1 \leqslant v < 2q_1 \ldots q_s$, удовлетворяющее системе сравнений

$$v \equiv 1 \pmod 2, \qquad v \equiv c_q^{x_q} \pmod q,$$

где q пробегает евклидовы простые числа.

7 шаг. Для каждого j, $j = 1, \ldots, p_1 \ldots p_k - 1$, находим $r_j \in \mathbb{N}$,

$$r_j \equiv v^j \pmod{2q_1 \cdots q_s}, \quad 0 < r_j < 2q_1 \ldots q_s,$$

и проверяем, делится ли наше число n на r_j? Если для всех j $r_j \nmid n$, то n — простое число.

Конец алгоритма.

Вывод. Для n проверяются некоторые тесты, обобщающие малую теорему Ферма. Если все они выполнены, то делители числа n лежат в небольшом явно описываемом множестве: это степени $v^j \pmod{2q_1 \ldots q_s}$ для некоторого явно построенного натурального числа v.

Сложность алгоритма составляет $O((p_1 \ldots p_k)^{\text{const}})$ арифметических операций, и можно показать, что для заданного n найдутся $p_1, \ldots, p_k$ такие, что

$$\prod q_i > \sqrt{n}, \quad p_1 \ldots p_k \leqslant (\log n)^{\text{const} \cdot \log\log\log n}.$$

Отсюда и получается приведенная выше оценка сложности алгоритма.

Впоследствии удалось снять условие бесквадратности чисел $q_j - 1$, что позволило использовать меньшие наборы чисел p_i. Например, имеется 27 простых чисел q_j таких, что

$$q_j - 1 \mid 2^4 \cdot 3^2 \cdot 5 \cdot 7,$$

причем $\prod\limits_j q_j > 10^{50}$. С этими наборами $\{p_i\}$ и $\{q_j\}$ можно проверять на простоту числа n, не превосходящие 10^{100}.

Указанные дальнейшие усовершенствования алгоритма Адлемана—Померанса—Румели и алгоритма Ленстры были предложены Ленстрой и Коеном [90]. Для алгоритма Ленстры—Коена нельзя получить оценку сложности $O((\log n)^{c \cdot \log\log\log n})$ арифметических операций без использования некоторых недоказанных гипотез. Однако на практике он оказался наиболее эффективным. При правильной организации его работы алгоритм всегда достаточно быстро выдает правильный ответ, простое n или составное. Описание и теоретическое обоснование алгоритма Ленстра—Коена довольно-таки объемно и выходит за рамки данной книги. Этот алгоритм проверяет на простоту числа порядка 10^{100}—10^{200} за несколько минут. Заметим также, что алгоритм Ленстры—Коена легко распараллелить на несколько компьютеров по количеству пар p и q.

В 1986 г. Голдвассер и Килиан [126] предложили алгоритм, позволяющий проверить простоту чисел с помощью эллиптических кривых. Этот алгоритм был существенно улучшен Аткином и Морейном [56].

В работе [56] приведены результаты тестирования алгоритма Аткина—Морейна на числах порядка 10^{800}—10^{1000}. Для проверки на простоту одного числа такой величины потребовалось несколько недель. Описание алгоритма Голдвассер—Килиана будет приведено в главе 4.

Естественным образом встал вопрос о том, является ли алгоритм Аткина—Морейна более быстрым, чем алгоритм Ленстры—Коена, а также вопрос о том, какого размера числа могут проверены на простоту каждым из этих алгоритмов за реальное время. Такие исследования проводились П. Михалеску (алгоритм Ленстры—Коена) и Ф. Морейном (алгоритм Аткина—Морейна), см. работы [185; 184; 199; 198; 195; 196].

Рекордные значения проверенных на простоту чисел для некоторого усовершенствования алгоритма Ленстры—Коена можно найти в работе [184]. С его помощью было проверено на простоту число $n = (2^{11279} + 1)/3$. С помощью метода эллиптических кривых было проверено на простоту число $n = (2^{12391} + 1)/3$, см. [199]. Сравнение этих двух алгоритмов проверки простоты, проведенное в [184] и [199], показывает, что алгоритм Ленстры—Коена, по-видимому, значительно быстрее. Преимущество метода эллиптических кривых заключается в том, что он предоставляет легко проверяемый сертификат простоты числа.

Несколько теоретических усовершенствований алгоритма Ленстры—Коена было предложено в работе [16]. В ней было показано, как можно применять тригонометрические суммы Гаусса и Якоби для аддитивных и мультипликативных характеров в конечных полях для проверки ряда условий в алгоритме Ленстры—Коена.

В работе [68] также были предложены некоторые усовершенствования алгоритма Ленстры—Коена.

В заключение скажем еще несколько слов об одном методе проверки простоты чисел. В 1992 г. Адлеман и Хуанг [47] предложили вероятностный алгоритм, имеющий полиномиальную сложность и позволяющий проверять простоту чисел с помощью гиперэллиптических кривых. В ходе его работы приходится проводить вычисления на якобианах алгебраических кривых. Алгоритм не реализован на компьютере и вряд ли когда-либо будет использоваться на практике. Теоретическая оценка его сложности составляет величину порядка $\log^{75} n$, где n — проверяемое на простоту число. Заметим, что лишь немногие специалисты в области алгебраической геометрии и алгебраической теории чисел понимают описание и обоснование алгоритма Адлемана—Хуанга.

На основе изложенного в данной главе материала можно сделать вывод о том, что в нашем распоряжении имеются быстрые и эффективные алгоритмы для проверки простоты чисел и построения больших простых чисел.

§ 1.9. Заключение. Детерминированный полиномиальный алгоритм проверки простоты чисел

К тому моменту, когда бóльшая часть этой книги уже была написана, появилась замечательная работа индийских математиков Агравала, Кайала и Саксены [50], в которой получен детерминированный алгоритм проверки простоты натуральных чисел, имеющий сложность $O(\log^{12} n(\log\log n)^c)$ арифметических операций (n — проверяемое на простоту число, c — некоторая абсолютная константа). Далее мы приводим описание и обоснование этого алгоритма. Символ $\widetilde{O}(t(n))$ мы будем использовать для обозначения $O(t(n)\log^{\alpha} n)$, где α — какая-либо положительная постоянная.

Алгоритм работы [50] основан на следующей теореме.

Теорема 1.71. *Пусть p — нечетное натуральное число, $a \in \mathbb{Z}$, $(a, p) = 1$. Число p является простым тогда и только тогда, когда*

$$(x-a)^p \equiv x^p - a \pmod{p} \tag{1.3}$$

(*соотношение (1.3) означает, что коэффициенты многочленов сравнимы по модулю p*).

Доказательство. Очевидно, что

$$(x-a)^p - (x^p - a) = \sum_{i=1}^{p-1} \binom{p}{i} x^i (-a)^{p-i} + a - a^p. \tag{1.4}$$

Если p — простое число, то соотношение (1.3) следует из (1.4), так как при $1 \leqslant i \leqslant p-1$ число $\binom{p}{i}$ делится на p.

Пусть соотношение (1.3) выполнено, и предположим, что p — составное. Тогда найдется простое число q и натуральное k такие, что $q^k \parallel p$, причем $q < p$. Очевидно, что q^k не делит

$$\binom{p}{q} = \frac{p(p-1)\dots(p-q+1)}{q!},$$

и поэтому коэффициент при x^q в (1.4) не делится на p, что противоречит выполнению (1.3). Теорема доказана. □

Символом $P(m)$ мы обозначаем наибольший простой делитель натурального числа m. Через $o_r(m)$ мы обозначаем порядок $m \pmod r$ в группе $(\mathbb{Z}/r\mathbb{Z})^*$.

Лемма 1.72. *Пусть p и r — различные простые числа. Тогда*

1) *для каждого $t \in \mathbb{N}$ группа $GF(p^t)^*$ является циклической*;

2) *для каждого многочлена $f(x) \in \mathbb{Z}[x]$ выполнено соотношение*

$$f(x)^p \equiv f(x^p) \pmod p;$$

3) *если $h(x) \in \mathbb{Z}[x]$, $h(x) \mid x^r - 1$, $m_1, m_2 \in \mathbb{Z}_{\geqslant 0}$, $m \equiv m_r \pmod r$, то*

$$x^m \equiv x^{m_r} \pmod{h(x)};$$

4) *если $o_r(p)$ — порядок $p \pmod r \in (\mathbb{Z}/r\mathbb{Z})^*$, то в $\mathbb{Z}/p\mathbb{Z}[x]$ многочлен $\dfrac{x^r-1}{x-1}$ раскладывается на различные неприводимые многочлены, каждый из которых имеет степень $o_r(p)$.*

Доказательство. Первое и второе утверждения леммы общеизвестны.

Пусть $m \geqslant m_r$, $m = m_r + kr$, где $k \in \mathbb{Z}_{\geqslant 0}$. Поскольку $x^{kr} \equiv 1 \pmod{x^r - 1}$, то $x^{kr+m_r} \equiv x^{m_r} \pmod{h(x)}$, что доказывает третье утверждение леммы.

Положим $d = o_r(p)$. Пусть $h(x)$ — неприводимый делитель $\dfrac{x^r-1}{x-1}$ в $\mathbb{Z}/p\mathbb{Z}[x]$, $\deg h(x) = k$. Тогда

$$\mathbb{Z}/p\mathbb{Z}[x]/(h(x)) = GF(p^k),$$
$$(\mathbb{Z}/p\mathbb{Z}[x]/(h(x)))^* = \langle g(x) \pmod{h(x)} \rangle_{p^k-1},$$

где $g(x)$ — некоторый многочлен из $\mathbb{Z}/p\mathbb{Z}[x]$. Очевидно, что $g(x)^{p^d} \equiv g(x^{p^d}) \pmod{h(x)}$. Поскольку $p^d \equiv 1 \pmod r$ и $h(x) \mid x^r - 1$, то $x^{p^d} \equiv x \pmod{h(x)}$. Следовательно, $g(x)^{p^d} \equiv g(x) \pmod{h(x)}$, откуда $g(x)^{p^d-1} \equiv 1 \pmod{h(x)}$. Это означает, что $p^k - 1 \mid p^d - 1$, и поэтому $k \mid d$.

Далее, $x^r \equiv 1 \pmod{h(x)}$ в $\mathbb{Z}/p\mathbb{Z}[x]$. Поскольку $x^r - 1$ не имеет кратных неприводимых делителей в $\mathbb{Z}/p\mathbb{Z}[x]$, то $h(x) \neq x - 1$. Следовательно, порядок $x \pmod{h(x)}$ равен простому числу r. Это, в свою очередь, означает, что $r \mid p^k - 1 = |GF(p^k)^*|$, т. е. $p^k \equiv 1 \pmod r$. По определению d тогда $d \mid k$.

Из доказанного выше следует, что $k = d$. Четвертое утверждение леммы доказано. □

Следующие две леммы содержат некоторые результаты о распределении простых чисел.

Лемма 1.73. *Существуют положительная постоянная* c_0 *и натуральное число* n_0 *такие, что для всех* $x \geqslant n_0$ *выполнено неравенство*

$$\#\{p \mid p \text{ — простое}, p \leqslant x, P(p-1) > x^{2/3}\} \geqslant \frac{c_0 x}{\log x}.$$

Доказательство можно найти в работах [123; 61].

Лемма 1.74. *При всех* $m \geqslant 2$ *выполнено неравенство*

$$\frac{m}{6\log_2 m} \leqslant \pi(m) \leqslant \frac{8m}{\log_2 m},$$

где $\pi(m)$ *— функция Чебышёва.*

Доказательство см. в [55].

Теперь опишем алгоритм проверки простоты натуральных чисел.

Алгоритм.

На входе задано нечетное число $n \in \mathbb{N}$, $n > 1$.

1 шаг. Если n имеет вид a^b, где $a \in \mathbb{N}$, $b \in \mathbb{N}$, $b \geqslant 2$, то выдать сообщение о том, что n составное, и закончить работу. (В работе [64] показано, что этот шаг может быть выполнен за $O(\log n^{1+o(1)})$ арифметических операций.)

2 шаг. $r := 2$.

3 шаг. Для текущего значения r выполнить шаги 4—8.

4 шаг. Если $r < n$ и НОД$(r, n) > 1$, то n — составное; в этом случае закончить работу с выдачей сообщения о том, что n — составное.

5 шаг. Если r — простое число, то выполнить шаги 6—7, иначе перейти на шаг 8.

6 шаг. Найти q — наибольший простой делитель числа $r-1$.

7 шаг. Если $q \geqslant 4\sqrt{r}\log_2 n$ и $n^{\frac{r-1}{q}} \not\equiv 1 \pmod r$, то перейти к 9 шагу с данным значением r.

8 шаг. $r := r+1$. Если $r \geqslant n$, то выдать сообщение, что n — простое, и закончить работу. Иначе вернуться на 3 шаг.

9 шаг. 1 случай. Если $n-1 \leqslant [2\sqrt{r}\log_2 n]$, то для всех a из промежутка $r < a \leqslant n-1$ проверить выполнение условия $(a, n) = 1$.

2 случай. Если $n-1 > [2\sqrt{r}\log_2 n]$, то для всех a из промежутка $1 \leqslant a \leqslant [2\sqrt{r}\log_2 n]$ проверить выполнение соотношения

$$(x-a)^n \equiv x^n - a \pmod{x^r - 1}$$

в кольце $\mathbb{Z}/n\mathbb{Z}[x]$. Если для некоторого a в 1-м случае выполнено неравенство $(a, n) > 1$, либо во 2-м случае соотношение по модулю $x^r - 1$ не выполняется, то n — составное, и алгоритм заканчивает работу.

10 шаг. Если мы дошли до этого шага, то число n — простое.
Конец алгоритма.

Теорема 1.75. *Алгоритм верно определяет, является ли число n простым или составным. При этом рассматриваемые в нем значения r не превосходят $A\log^6 n$ для некоторой абсолютной константы A.*

Для доказательства теоремы 1.75 нам потребуется еще несколько лемм.

Лемма 1.76. *Существуют абсолютные положительные постоянные c_1, c_2 такие, что если число n достаточно велико, то на отрезке $[c_1\log_2^6 n; c_2\log_2^6 n]$ найдется простое число r, удовлетворяющее следующим условиям: либо $r \mid n$, либо у $r-1$ есть простой делитель q, удовлетворяющий неравенству $q \geqslant 4\sqrt{r}\log_2 n$, такой, что $q \mid o_r(n)$ и $n^{(r-1)/q} \not\equiv 1 \pmod r$.*

Доказательство. Рассмотрим простые числа r, лежащие на отрезке $[c_1\log_2^6 n; c_2\log_2^6 n]$ и удовлетворяющие неравенству

$$P(r-1) > (c_2\log_2^6 n)^{2/3}, \tag{1.5}$$

где c_1 и c_2 — некоторые положительные постоянные, которые мы выберем позже; такие простые числа мы назовем специальными. Их количество не меньше, чем количество простых чисел на отрезке $[1; c_2\log_2^6 n]$, удовлетворяющих (1.5), минус количество всех простых чисел на отрезке $[1; c_1\log_2^6 n]$. С помощью лемм 1.73 и 1.74 получим, что количество специальных простых чисел будет не меньше, чем

$$\frac{c_0c_2\log_2^6 n}{\log_2 c_2 + 6\log_2\log_2 n} - \frac{8c_1\log_2^6 n}{\log_2 c_1 + 6\log_2\log_2 n} \geqslant \frac{c_0c_2\log_2^6 n}{7\log_2\log_2 n} - \frac{8c_1\log_2^6 n}{6\log_2\log_2 n} =$$
$$= \frac{\log_2^6 n}{\log_2\log_2 n}\left(\frac{c_0c_2}{7} - \frac{8c_1}{6}\right) = c_3\frac{\log_2^6 n}{\log_2\log_2 n},$$

если число n достаточно велико. Мы будем считать, что $c_1 > 4^6$, $c_2 > c_1$, и что $c_3 > 0$. Пусть $x = c_2\log_2^6 n$. Обозначим через Π произведение

$$\Pi = (n-1)(n^2-1)(n^3-1)\dots(n^{[x^{1/3}]}-1). \tag{1.6}$$

В этом произведении $[x^{1/3}]$ сомножителей, и каждый состоит из произведения не более чем $\log_2(n^{x^{1/3}}-1) \leqslant x^{1/3}\log_2 n$ простых чисел. Поэтому Π состоит из произведения не более чем $x^{2/3}\log_2 n$ простых. Далее,

$$x^{2/3}\log_2 n = c_2^{2/3}\log_2^5 n < c_3\frac{\log_2^6 n}{\log_2\log_2 n}$$

при всех достаточно больших n. Значит, хотя бы одно специальное простое r не делит Π. Покажем, что оно будет удовлетворять утверждению леммы. Пусть $r \nmid n$. Тогда $n^{r-1} \equiv 1 \pmod r$. Из (1.5) следует, что $P(r-1) > r^{2/3}$, т. е. $r-1$ имеет простой делитель q такой, что $q > r^{2/3}$. Заметим, что $r^{2/3} \geqslant 4\sqrt{r}\log_2 n$, поскольку $r \geqslant c_1 \log_2^6 n$ и $c_1 > 4^6$. Значит, $q \geqslant 4\sqrt{r}\log_2 n$ и $q^2 \nmid r-1$. Если $q \nmid o_r(n)$, то $n^{(r-1)/q} \equiv 1 \pmod r$, $r \mid n^{(r-1)/q} - 1$. Покажем, что $\dfrac{r-1}{q} \leqslant [x^{1/3}]$. Действительно, $r-1 \leqslant x^{1/3}q$, так как выполнено даже более сильное неравенство

$$r \leqslant x^{1/3} r^{2/3},$$

поскольку $r \leqslant x = c_2 \log_2^6 n$. Из доказанного следует, что $r \mid \Pi$, но это противоречит выбору r. Итак, $q \mid o_r(n)$ и $n^{(r-1)/q} \not\equiv 1 \pmod r$. Лемма доказана. $\square$

Замечание 1.77. Из леммы 1.76 следует, что если n достаточно велико, то на 3 шаге алгоритма либо обнаружится, что n составное, либо найдется простое число r, $r \leqslant c_2 \log_2^6 n$, такое, что у $r-1$ есть простой делитель q, удовлетворяющий условиям 7 шага.

Лемма 1.78. *Если число n — простое, то алгоритм закончит работу с выдачей сообщения о том, что n — простое.*

Доказательство. Если n не очень велико, то алгоритм может просто перебрать все значения $r < n$ и на 8 шаге при $r = n$ выдать сообщение о том, что n — простое; либо мы перейдем к 9 шагу с некоторым значением r и (по теореме 1.71) тесты 9 шага будут выполнены для всех рассматриваемых a, вследствие чего алгоритм выдаст сообщение о том, что n — простое.

Предположим теперь, что n достаточно велико. Тогда по лемме 1.76 на 3 шаге алгоритма обнаружится простое число r такое, что $r \leqslant c_2 \log_2^6 n < n$, причем $r-1$ имеет простой делитель q, удовлетворяющий условиям 7 шага. На 9 шаге соотношение по модулю $x^r - 1$ будет выполнено (по теореме 1.71) для всех рассматриваемых значений a, так как

$$a \leqslant 2\sqrt{r}\log_2 n \leqslant 2\sqrt{c_2}\log_2^4 n < n-1,$$

если n достаточно велико. Значит, мы дойдем до 10 шага, и алгоритм выдаст сообщение о простоте n. $\square$

Теперь предположим, что n — составное число. Пусть $n = p_1 \dots p_k$ — разложение n на простые множители (не обязательно различные). Допустим, что в ходе перебора значений r на 3 шаге мы не обнаружили, что n составное, нашли простые числа r и q, удовлетворяющие условиям 6 и 7 шагов и перешли к 9 шагу. Поскольку $q \mid o_r(n)$

и $o_r(n) \mid \operatorname{НОК}_i(o_r(p_i))$, то существует простое число p, делящее n, такое, что $q \mid o_r(p)$. Зафиксируем это p.

Положим $l = [2\sqrt{r}\log_2 n]$, и пусть $l < n-1$. На 9 шаге мы перебираем значения a, $1 \leqslant a \leqslant l$. По п. 4 леммы 1.72 в $\mathbb{Z}/p\mathbb{Z}[x]$ существует неприводимый многочлен $h(x)$, делящий $x^r - 1$, причем $\deg h(x) = d = o_r(p) \geqslant 2$ (так как q делит d). Если соотношение по модулю $x^r - 1$ на 9 шаге для какого-то конкретного a будет выполнено, то будет выполнено и соотношение

$$(x-a)^n \equiv x^n - a \pmod{h(x)} \tag{1.7}$$

в кольце $\mathbb{Z}/p\mathbb{Z}[x]$, т. е. $(x-a)^n = x^n - a$ в поле $(\mathbb{Z}/p\mathbb{Z}[x])/(h(x)) = GF(p^d)$. При этих предположениях и обозначениях докажем леммы 1.79—1.81.

Лемма 1.79. *Рассмотрим мультипликативную группу* $G \subseteq ((\mathbb{Z}/p\mathbb{Z}[x])/(h(x)))^*$,

$$G = \left\{ \prod_{a=1}^{l} (x-a)^{\alpha_a} \pmod{h(x)} \mid \alpha_a \in \mathbb{Z}_{\geqslant 0},\ a = 1, \ldots, l \right\},$$

порожденную биномами $x-a$, $a = 1, \ldots, l$. *Эта группа является циклической. Кроме того,* $|G| > (d/l)^l$, *если алгоритм доходит до* 9*-го шага и* $l < n-1$.

Доказательство. Поскольку $\deg h(x) \geqslant 2$, то элементы $x - a \pmod{h(x)}$ имеют конечный порядок в $((\mathbb{Z}/p\mathbb{Z}[x])/(h(x)))^*$, и G является группой. Так как G — подгруппа циклической группы $GF(p^d)^*$, то она тоже является циклической.

Покажем, что элементы подмножества

$$S = \left\{ \prod_{a=1}^{l} (x-a)^{\alpha_a} \pmod{h(x)} \;\middle|\; \sum_{a=1}^{l} \alpha_a \leqslant d-1 \right\}$$

группы G различны в $(\mathbb{Z}/p\mathbb{Z}[x])/(h(x))$, если алгоритм дошел до выполнения 9 шага и $l < n-1$. На 9 шаге будет проверяться условие 2 случая и $r > q \geqslant 4\sqrt{r}\log_2 n \geqslant 2l$. При этом рассматриваемые значения a различны по модулю p, так как если $a_1 \equiv a_2 \pmod{p}$, $a_1 < a_2$, то $p \leqslant a_2 - a_1 < l < r$, и тогда мы бы уже на 4-м шаге для значения $r_1 = p < r$ обнаружили, что n — составное. Итак, величины $a \pmod{p}$ различны. Поэтому многочлены $\prod_{a=1}^{l} (x-a)^{\alpha_a}$ в кольце $\mathbb{Z}/p\mathbb{Z}[x]$ различны, а так как $\sum_{a=1}^{l} \alpha_a \leqslant d-1 < \deg h(x)$, то и элементы множества S

различны в $(\mathbb{Z}/p\mathbb{Z}[x])/(h(x))$. Поскольку в множестве S содержится $\binom{l+d-1}{l}$ элементов и $\binom{l+d-1}{l} > \left(\frac{d}{l}\right)^l$, то $|G| \geqslant |S| > (d/l)^l$. $\square$

Пусть алгоритм дошел до выполнения 9 шага, причем $l < n-1$. Так как $d = o_r(n) : q$, $q \geqslant 4\sqrt{r}\log_2 n \geqslant 2l$, то $d \geqslant 2l$ и

$$|G| > 2^l = 2^{[2\sqrt{r}\log_2 n]} > 2^{2\sqrt{r}\log_2 n - 1} = n^{2\sqrt{r}}/2. \tag{1.8}$$

Пусть $g(x)$ — образующий циклической группы G. Его порядок будет больше $n^{2\sqrt{r}}/2$. Положим

$$I_{g(x)} = \{m \mid m \in \mathbb{Z}_{\geqslant 0},\ g(x)^m \equiv g(x^m) \pmod{x^r - 1} \text{ в кольце } \mathbb{Z}/p\mathbb{Z}[x]\}.$$

Лемма 1.80. *Множество $I_{g(x)}$ замкнуто относительно умножения.*

Доказательство. Пусть $m_1, m_2 \in I_{g(x)}$. Тогда в кольце $\mathbb{Z}/p\mathbb{Z}[x]$ выполнены сравнения

$$g(x)^{m_1} \equiv g(x^{m_1}) \pmod{x^r - 1}, \qquad g(x)^{m_2} \equiv g(x^{m_2}) \pmod{x^r - 1}.$$

Подставим x^{m_1} вместо x во второе сравнение. Тогда

$$g(x^{m_1})^{m_2} \equiv g(x^{m_1 m_2}) \pmod{x^{rm_1} - 1},$$

откуда

$$g(x^{m_1})^{m_2} \equiv g(x^{m_1 m_2}) \pmod{x^r - 1}.$$

Поэтому

$$g(x)^{m_1 m_2} \equiv (g(x)^{m_1})^{m_2} \equiv (g(x^{m_1}))^{m_2} \equiv g(x^{m_1 m_2}) \pmod{x^r - 1},$$

т. е. $m_1 m_2 \in I_{g(x)}$. $\square$

Лемма 1.81. *Пусть o_g — порядок $g(x)$ в $\mathbb{Z}/p\mathbb{Z}[x]/(h(x))$. Пусть $m_1,\ m_2 \in I_{g(x)}$. Тогда из сравнения $m_1 \equiv m_2 \pmod r$ следует, что $m_1 \equiv m_2 \pmod{o_g}$.*

Доказательство. Пусть $m_2 > m_1$. Тогда $m_2 = m_1 + kr$, где $k \in \mathbb{Z}_{\geqslant 0}$. Так как в кольце $\mathbb{Z}/p\mathbb{Z}[x]$ выполнено соотношение $g(x)^{m_2} \equiv g(x^{m_2}) \pmod{x^r - 1}$ и $h(x) \mid x^r - 1$, то $g(x)^{m_2} \equiv g(x^{m_2}) \pmod{h(x)}$. Отсюда

$$g(x)^{m_1} g(x)^{kr} \equiv g(x^{m_1 + kr}) \equiv g(x^{m_1}) \pmod{h(x)}.$$

Так как $m_1 \in I_{g(x)}$, то получаем, что $g(x)^{kr} \equiv 1 \pmod{h(x)}$, т. е. $kr \equiv 0 \pmod{o_g}$. Из этого следует утверждение леммы. $\square$

Замечание 1.82. Из леммы 1.81 вытекает, что в $I_{g(x)}$ найдется не более чем r чисел, меньших o_g.

Лемма 1.83. *Если число n — составное, то алгоритм закончит работу с выдачей сообщения о том, что n — составное.*

Доказательство. Предположим, алгоритм выдал сообщение, что n — простое. Это не может произойти на шаге 8, так как тогда мы проверили бы условия шага 4 для всех $r < n$ и нашли бы делитель n. Поэтому алгоритм дошел до шага 10. Значит, был выполнен шаг 9. Если $n - 1 \leqslant [2\sqrt{r}\log_2 n]$, то на шаге 9 мы бы обнаружили a, $a \leqslant n-1$, такое, что $(a, n) > 1$, и алгоритм сообщил бы, что n составное. Значит, $[2\sqrt{r}\log_2 n] < n - 1$, и для всех a, $1 \leqslant a \leqslant [2\sqrt{r}\log_2 n] = l$, выполнено соотношение:

$$(x-a)^n \equiv x^n - a \pmod{x^r - 1} \quad \text{в } \mathbb{Z}/p\mathbb{Z}[x]. \tag{1.9}$$

Тогда в обозначениях лемм 1.80 и 1.81 (где $g(x)$ — по-прежнему образующий элемент группы G) получим, что

$$g(x)^n \equiv g(x^n) \pmod{x^r - 1} \quad \text{в } \mathbb{Z}/p\mathbb{Z}[x],$$

поскольку $g(x)$ есть произведение биномов $x - a$, для которых выполнено (1.9). Значит, $n \in I_{g(x)}$. Также по п. 2 леммы 1.72 $p \in I_{g(x)}$; кроме того, $1 \in I_{g(x)}$.

Рассмотрим множество $E = \{n^i p^j \mid 0 \leqslant i, j \leqslant [\sqrt{r}]\}$. По лемме 1.80 $E \subseteq I_{g(x)}$. Так как $|E| = ([\sqrt{r}]+1)^2 > r$, то найдутся две различные пары (i_1, j_1), (i_2, j_2), такие, что $n^{i_1}p^{j_1} \equiv n^{i_2}p^{j_2} \pmod r$. Тогда по лемме 1.81 $n^{i_1}p^{j_1} \equiv n^{i_2}p^{j_2} \pmod{o_g}$. Так как o_g делит $p^d - 1 = |(\mathbb{Z}/p\mathbb{Z}[x]/(h(x)))^*|$, то $p \nmid o_g$, и элемент $p \pmod{o_g}$ обратим в $\mathbb{Z}/o_g\mathbb{Z}$. Не ограничивая общности, будем считать, что $j_2 \geqslant j_1$. Тогда

$$n^{i_1} \equiv n^{i_2}p^{j_2 - j_1} \pmod{o_g}. \tag{1.10}$$

Поскольку $3 \leqslant p \leqslant n/3$, справедливы неравенства

$$n^{i_1} \leqslant n^{[\sqrt{r}]} \leqslant n^{\sqrt{r}} \leqslant n^{2\sqrt{r}}/2,$$

$$n^{i_2}p^{j_2-j_1} \leqslant n^{[\sqrt{r}]}\left(\frac{n}{3}\right)^{[\sqrt{r}]} = \frac{n^{2[\sqrt{r}]}}{3^{[\sqrt{r}]}} \leqslant n^{2\sqrt{r}}/2.$$

Из (1.10) и (1.8) следует теперь, что

$$n^{i_1} = n^{i_2}p^{j_2-j_1}. \tag{1.11}$$

Поскольку p — простой делитель натурального числа n, то из (1.11) вытекает, что составное число n является степенью p. (Действительно, если у числа n есть простой делитель s, $s \neq p$, то из (1.11) следует, что $i_1 = i_2$. Но тогда и $j_1 = j_2$, что противоречит условию $(i_1, j_1) \neq (i_2, j_2)$.) Однако такие составные числа мы обнаруживаем уже на шаге 1 алгоритма. Полученное противоречие доказывает лемму. □

Теперь докажем теорему 1.75. Корректность работы алгоритма следует из лемм 1.78 и 1.83. Неравенство $r \leqslant A\log^6 n$ для небольших

значений n обеспечивается за счет выбора постоянной A, а для всех достаточно больших n в силу леммы 1.76 можно рассматривать $r \leqslant \leqslant c_2 \log_2^6 n$. Теорема 1.75 полностью доказана.

Оценим количество арифметических операций, требуемых для выполнения алгоритма.

Теорема 1.84. *Количество арифметических операций, необходимых для выполнения алгоритма, равно* $\widetilde{O}(\log^{12} n)$.

Доказательство. Можно считать, что n достаточно велико. Шаг 1 алгоритма выполняется за $O((\log n)^{1+o(1)})$ арифметических операций, согласно [64]. Количество значений r, рассматриваемых на шаге 3, по лемме 1.76 не превосходит $c_2 \log_2^6 n$. Для каждого r шаг 4 выполняется за $O(\log n)$ арифметических операций, а шаги 5 и 6 с помощью решета Эратосфена выполняются за $O(r^{1/2}(\log r)^{\mathrm{const}}) = \widetilde{O}(\log^3 n)$. Шаг 7 выполняется за $O(\log r) = O(\log \log n)$ операций, шаг 8 тривиален. На шаге 9 (в силу того, что n достаточно велико) будет проверяться условие 2 случая. При этом проверка соотношения

$$(x-a)^n \equiv x^n - a \pmod{x^r - 1}$$

в кольце $\mathbb{Z}/p\mathbb{Z}[x]$ составляет с помощью бинарного возведения в степень (см. Приложение) и быстрого преобразования Фурье (см. гл. 9) $\widetilde{O}(\log n \cdot r \log n)$ операций. Поэтому 9 шаг будет выполнен за $\widetilde{O}(2\sqrt{r} \log n \cdot r \log^2 n) = \widetilde{O}(\log^{12} n)$ арифметических операций. Таким образом, теорема доказана. □

Замечание 1.85. Проверка соотношения 2 случая 9 шага алгоритма может производиться и без быстрого преобразования Фурье. Оценка сложности останется полиномиальной, но несколько худшей.

Замечание 1.86. В работе [50] показано, что если выполнена некоторая гипотеза о распределении простых чисел Софи Жермен, т. е. пар простых чисел q и $p = 2q + 1$, то можно предложить алгоритм проверки простоты чисел со сложностью $\widetilde{O}(\log^6 n)$. В предположении справедливости некоторой другой гипотезы можно описать алгоритм проверки простоты чисел со сложностью $\widetilde{O}(\log^3 n)$.

Замечание 1.87. Пока не совсем ясно, будет ли описанный выше алгоритм эффективен на практике. Двенадцатая степень логарифма n в оценке сложности — это все же довольно много. Кроме того, значение параметра r в алгоритме теоретически может иметь величину порядка $\log^6 n$, и поэтому нам придется работать с многочленами высоких степеней. Результаты практической реализации данного алгоритма пока неизвестны.

Глава 2. Факторизация целых чисел с экспоненциальной сложностью

§ 2.1. Введение. Метод Ферма

В данной главе мы рассматриваем алгоритмы разложения натурального числа n на множители, делающие $O(n^c)$ арифметических операций, где c — некоторая постоянная, $0 < c < 1$; либо делающие $O(n^{c_1} \log^{c_2} n)$ арифметических операций при некоторых постоянных c_1, c_2. Мы будем ограничиваться поиском разложения на два множителя: $n = ab$, $1 < a \leqslant b < n$. Если алгоритм находит такое разложение за $O(f(n))$ арифметических операций, то полное разложение n на простые множители будет найдено за $O(f(n) \log n)$ арифметических операций, поскольку n состоит из произведения не более чем $\log_2 n$ простых чисел.

Прежде чем приступать к факторизации целого числа, следует убедиться, что оно действительно составное. Для этого лучше всего использовать один из вероятностных тестов на простоту, например, алгоритм Миллера—Рабина из главы 1.

Простейший метод пробных делений для разложения n на множители был описан в § 1.2 главы 1. Он требует $O(n^{1/2})$ арифметических операций. Другие алгоритмы факторизации, имеющие сложность $O(n^{1/2})$, можно найти в книге Д. Кнута [25, § 4.5.4] (см. также первое издание этой книги). Мы опишем здесь алгоритм П. Ферма, полученный им в 1643 г. Этот алгоритм вычисляет наибольший множитель a числа n, не превосходящий $n^{1/2}$. При этом в алгоритме не используется операция деления, а только сложение, вычитание и умножение. Заметим, что если $n = pq$, где p и q — простые числа, примерно одинаковые по величине, то алгоритм Ферма быстро разложит n. Это следует учитывать при выборе модулей в криптосистеме RSA.

Алгоритм Ферма

Пусть n — составное число, $n = ab$, где $1 < a \leqslant b$, причем a — наибольшее возможное. Положим $a = u - v$, $b = u + v$, где u и v —

натуральные числа, $u = \dfrac{a+b}{2}$, $v = \dfrac{b-a}{2}$, $n = ab = u^2 - v^2$. Алгоритм Ферма ищет представление n в виде $n = u^2 - v^2$, откуда получается разложение $n = (u-v)(u+v) = ab$.

Мы работаем с величинами

$$r_k = x_k^2 - y_k^2 - n, \quad k = 0, 1, 2, \ldots$$

Начальное значение $(x_0, y_0) = ([\sqrt{n}], 0)$. Увеличение номера k происходит по следующим правилам. Если $r_k = 0$, то наша цель достигнута, $n = x_k^2 - y_k^2 = (x_k - y_k)(x_k + y_k)$, и алгоритм останавливается. Если $r_k > 0$, то

$$(x_{k+1}, y_{k+1}) := (x_k, y_k + 1),$$

если же $r_k < 0$, то

$$(x_{k+1}, y_{k+1}) := (x_k + 1, y_k);$$

затем

$$r_{k+1} := x_{k+1}^2 - y_{k+1}^2 - n.$$

Наша цель — доказать, что за конечное число шагов алгоритм дойдет до значения $r_k = 0$, и что для первого такого значения справедливо равенство $x_k - y_k = a$, где a — наибольший натуральный делитель n, не превосходящий $n^{1/2}$. Если n является полным квадратом, то это очевидно по определению x_0 и y_0. Далее мы считаем, что n — не полный квадрат.

Рассмотрим функцию $r(x, y) = x^2 - y^2 - n$. Очевидно, что при неотрицательных x и y выполнены неравенства

$$r(x, y+1) < r(x, y) < r(x+1, y).$$

Кроме того, в алгоритме Ферма x_k и y_k всегда неотрицательны и не убывают (по построению).

Рассмотрим декартову систему координат на плоскости и решетку целых точек $\mathbb{Z}^2$. Она разбивает плоскость на единичные квадраты; каждый квадрат будем нумеровать точкой (x, y), стоящей в его левом нижнем углу. В квадрат, пронумерованный точкой $(x, y) \in \mathbb{Z}^2$ мы впишем знак величины $r(x, y)$: $+$ (плюс), $-$ (минус) или 0, если $r(x, y) = 0$. Очевидно, что если в некотором квадрате стоит знак $-$ (минус), то и во всех квадратах над ним тоже стоит знак $-$ (минус); если в некотором квадрате стоит знак $+$ (плюс), то и во всех квадратах правее тоже стоит знак $+$ (плюс).

В алгоритме Ферма мы двигаемся по квадратам. Начало движения — квадрат, занумерованный (x_0, y_0); в нем стоит знак $-$ (минус).

Очередной шаг мы делаем вверх, если в квадрате стоит знак + (плюс), и вправо — если в квадрате стоит знак − (минус). Самый первый шаг мы делаем вправо.

При движении по квадратам в алгоритме Ферма мы не можем двигаться все время вверх, а обязательно будем делать шаги вправо. Пусть мы достигли точки (x_k, y_k), где $y_k \geqslant 1$. Покажем индукцией по k, что тогда $r(x_k, y_k - 1) > 0$, т. е. под квадратом (x_k, y_k) стоит знак + (плюс).

Основание индукции мы проверяем для значения $k = l$, для которого квадрат, занумерованный $(x_l, y_l) = (x_l, 1)$, есть первый квадрат, в котором $y_l = 1$, а $(x_{l-1}, y_{l-1}) = (x_{l-1}, 0)$. В этот квадрат мы пришли снизу, а это означает, что $r(x_{l-1}, y_{l-1}) = r(x_l, y_l - 1) > 0$. Это и есть выполнение основания индукции.

В квадрат (x_k, y_k) мы попали либо слева, либо снизу. Если снизу, то мы наращивали y, и тогда очевидно, что $r(x_k, y_k - 1) > 0$. Если слева, то $(x_{k-1}, y_{k-1}) = (x_k - 1, y_k)$. Тогда по предположению индукции

$$r(x_{k-1}, y_{k-1} - 1) > 0.$$

Из предположения индукции теперь следует, что $r(x_{k-1} + 1, y_{k-1} - 1) > 0$, т. е. $r(x_k, y_k - 1) > 0$, что и требовалось доказать.

Заметим, что число u — наименьшее натуральное число, для которого возможно представление $n = u^2 - v^2$. В самом деле, $n = ab$, $b = \dfrac{n}{a}$, $u = \dfrac{a+b}{2} = \dfrac{1}{2}\left(a + \dfrac{n}{a}\right)$; $u'(a) = \dfrac{1}{2}\left(1 - \dfrac{n}{a^2}\right)$, и поскольку $a^2 < n$, то $u'(a) < 0$; с ростом a величина $u = u(a)$ убывает и принимает наименьшее значение при наибольшем a, $a^2 < n$.

Пусть мы первый раз достигли точки (x_k, y_k), в которой $x_k = u$ (этот момент обязательно наступит согласно доказанному выше). Если $y_k = v$, то мы достигли желаемого результата, $r(x_k, y_k) = 0$, алгоритм заканчивает работу и выдает пару $(a, b) = (u - v, u + v)$.

Если $y_k < v$, то $r(x_k, y_k) = u^2 - y_k^2 - n = u^2 - y_k^2 - (u^2 - v^2) = v^2 - y_k^2 > 0$. В этом случае мы двигаемся вверх, наращивая y, до тех пор, пока $y_{k+j} = v$, т. е. мы найдем $x_{k+j} = u$, $y_{k+j} = v$, $r(x_{k+j}, y_{k+j}) = 0$ и алгоритм закончит работу и выдаст пару $(a, b) = (u - v, u + v)$.

Осталось рассмотреть последний случай $y_k > v$. Этот случай невозможен, так как по доказанному выше под квадратом (x_k, y_k) стоит знак + (плюс), т. е. $r(u, y_k - 1) > 0$. Значит, и всюду ниже стоит знак + (плюс). А в нашем квадрате $(x_k, y_k) = (u, y_k)$,

$$r(x_k, y_k) = u^2 - y_k^2 - n = v^2 - y_k^2 < 0,$$

т. е. стоит знак − (минус). Значит, 0 здесь нигде не может стоять, а он должен быть в этом столбце, поскольку $n = u^2 - v^2$.

Итак, мы достигнем в алгоритме точки $(x_k, y_k) = (u, v)$, получим значение $r_k = 0$ и найдем $a = u - v$, что и требовалось доказать.

Замечание 2.1. В работе [178] описано некоторое усовершенствование метода Ферма.

§ 2.2. $(P-1)$-метод Полларда

Этот метод был впервые описан в работе [218], см. также, [89, гл. 8]. Он основан на следующей идее. Допустим, что у числа n, которое мы хотим разложить на множители, есть простой делитель p такой, что число $p-1$ является B-степенно-гладким для некоторой границы гладкости $B > 0$. Это означает, что для любого простого числа q, $q \mid p-1$, выполнено неравенство

$$q^{\nu_q(p-1)} \leqslant B.$$

Отсюда следует, что $p - 1 \mid \mathrm{НОК}(1, 2, \ldots, B)$. Если мы выберем $a \in \mathbb{N}$ такое, что $(a, n) = 1$, то по малой теореме Ферма

$$a^{\mathrm{НОК}(1,2,\ldots,B)} \equiv 1 \pmod{p}.$$

Следовательно, $\mathrm{НОД}(a^{\mathrm{НОК}(1,2,\ldots,B)} - 1, n)$ делится на p и поэтому содержит нетривиальный делитель n (НОД может быть и равен n).

1 стадия $(P-1)$-метода Полларда

В $(P-1)$-методе Полларда мы выбираем априорную границу гладкости B, исходя из возможностей нашего компьютера и времени, которое мы рассчитываем потратить. Обычно $B \asymp 10^5$—10^6. Далее составляем таблицу $q_1 < q_2 < \ldots < q_k \leqslant B$ всех простых чисел, не превосходящих B, и для каждого q_i полагаем

$$\beta(q_i) = \left[\frac{\log B}{\log q_i}\right], \quad \text{т. е.} \quad q_i^{\beta(q_i)} \leqslant B, \quad q_i^{\beta(q_i)+1} > B.$$

Далее мы выбираем значение a (например, $a = 2$). Затем последовательными возведениями в степень и приведениями по модулю n вычисляем

$$P_{20} = \left(a^{q_1^{\beta(q_1)}} - 1\right)\left(a^{q_1^{\beta(q_1)} \cdot q_2^{\beta(q_2)}} - 1\right) \ldots \left(a^{q_1^{\beta(q_1)} \ldots q_{20}^{\beta(q_{20})}} - 1\right) \pmod{n}$$

(параметр 20 также априорный). Далее вычисляем $\mathrm{НОД}(P_{20}, n)$. Если этот НОД тривиальный, то добавляем к P_{20} следующее произведение длины 20, т. е. находим

$$P_{40} = P_{20} \cdot \left(a^{q_1^{\beta(q_1)} \ldots q_{21}^{\beta(q_{21})}} - 1\right) \ldots \left(a^{q_1^{\beta(q_1)} \ldots q_{40}^{\beta(q_{40})}} - 1\right) \pmod{n},$$

снова считаем НОД(P_{40}, n) и так далее. Предположим, что при некотором $k \geqslant 1$ оказалось, что НОД$(P_{20k}, n) > 1$. Тогда мы возвращаемся к значению $k-1$ и, полагая $b = a^{q_1^{\beta(q_1)} \cdots q_{20(k-1)}^{\beta(q_{20(k-1)})}}$, начинаем последовательно вычислять наибольшие общие делители

$$\text{НОД}(b^{q_{20(k-1)+1}} - 1 \pmod n,\ n),$$

$$\text{НОД}\left(b^{q_{20(k-1)+1}^2} - 1 \pmod n,\ n\right),$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\text{НОД}\left(b^{q_{20(k-1)+1}^{\beta(q_{20(k-1)+1})}} - 1 \pmod n,\ n\right),$$

$$\text{НОД}\left(b^{q_{20(k-1)+1}^{\beta(q_{20(k-1)+1})} \cdot q_{20(k-1)+2}} - 1 \pmod n,\ n\right),$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\text{НОД}\left(b^{\prod\limits_{j=1}^{20} q_{20(k-1)+j}^{\beta(q_{20(k-1)+j})}} - 1 \pmod n,\ n\right),$$

до первого нетривиального общего делителя.

Значение нахождения P_{20}, P_{40}, P_{60}, …, состоящих из порций по 20 степеней простых чисел, состоит именно в экономии на вычислениях наибольшего общего делителя. Заметим, что поскольку количество простых чисел q_i не обязано делится на 20, то последняя порция может быть неполной.

2 стадия $(P-1)$-метода Полларда

Предположим, что $p \mid n$, $p-1$ не является B-степенно-гладким числом, но $p-1 = f \cdot r$, где f — B-степенно-гладкое число и r — простое число, $B < r < B_1$. Допустим, что на 1-й стадии $(P-1)$-метода Полларда мы вычислили

$$b = a^{\text{НОК}(1,2,\ldots,B)} \pmod n.$$

Тогда $b^r \equiv 1 \pmod p$, и НОД$(b^r - 1 \pmod n, n)$ будет делиться на p по малой теореме Ферма.

Поэтому на 2 стадии $(P-1)$-метода Полларда мы находим все простые числа $r_1, \ldots, r_N$, $B < r_1 < r_2 < \ldots < r_N < B_1$, и составляем разности $d_i = r_i - r_{i-1}$, $i = 2, \ldots, N$. Эти разности обычно невелики и количество различных таких разностей также невелико (при подходящем выборе B_1). Затем мы табулируем элементы $b^{d_i} \pmod n$ для всех различных значений d_i.

После этого в алгоритме мы находим

$$x_1 \equiv b^{r_1} \pmod{n},$$

после чего вычисляем

$$x_i \equiv b^{r_i} \pmod{n} \equiv x_{i-1} \cdot b^{d_i} \pmod{n}, \quad i = 2, \ldots, N,$$

и находим

$$\text{НОД}(x_i - 1 \pmod{n}, n), \quad i = 1, \ldots, N.$$

Здесь также возможна организация вычислений порциями по 20 для экономии количества нахождений наибольших общих делителей.

Замечание 2.2. Оценка сложности $(P-1)$-метода Полларда в худшем случае составляет $O(n^{1/2}\log^c n)$ арифметических операций. Однако в некоторых случаях алгоритм может быстро выдать делитель числа n. Во всех случаях алгоритм хорошо находит небольшие простые делители n, потому что они являются степенно-гладкими для небольшой границы гладкости B.

Замечание 2.3. Если какой-либо из вычисляемых в алгоритме наибольших общих делителей оказался равным n, то имеет смысл попробовать другое основание a, например, $a = 3$.

Замечание 2.4. В работе [194] предложено усовершенствование $(P-1)$-метода Полларда с помощью дискретного преобразования Фурье.

Вывод. На практике $(P-1)$-метод Полларда обычно используют до применения более сильных алгоритмов факторизации для того, чтобы отделить небольшие простые делители числа n.

§ 2.3. ρ-метод Полларда

ρ-метод Полларда был впервые описан в работе [219]. С его помощью было разложено на множители число Ферма $F_8 = 2^{256} + 1$, см. [75]. Усовершенствования этого метода можно найти в работе [71], см. также книги [89; 144; 25, гл. 4; 37]. Поскольку ρ-метод изложен во многих книгах и статьях, мы приводим здесь достаточно конспективное его описание.

Схема ρ-метода.

На входе задано число $n \in \mathbb{N}$, которое мы хотим разложить на множители.

1 шаг. Выбрать отображение

$$f\colon \mathbb{Z}/n\mathbb{Z} \longrightarrow \mathbb{Z}/n\mathbb{Z}.$$

Обычно $f(x)$ — многочлен степени большей или равной 2, например, $f(x) = x^2 + 1$.

2 шаг. Случайно выбрать $x_0 \in \mathbb{Z}/n\mathbb{Z}$ и вычислять члены рекуррентной последовательности $x_0, x_1, x_2, \ldots$ по правилу

$$x_i \equiv f(x_{i-1}) \pmod{n}.$$

3 шаг. Для некоторых номеров j, k проверять условие

$$1 < \text{НОД}(x_j - x_k, n) < n$$

до тех пор, пока не будет найден делитель числа n, или пока не закончится время, отведенное для работы алгоритма.

Конец алгоритма.

Замечание 2.5. Выбор номеров j, k на третьем шаге алгоритма обычно делают одним из следующих способов.

1. Для каждого j перебирают все k, $k < j$; это долго, и требуется большая память компьютера.

2. Рассматривают пары k и $2k$, т. е. проверяют условие

$$1 < \text{НОД}(x_{2k} - x_k, n) < n.$$

3. Если j заключено в пределах $2^h \leqslant j < 2^{h+1}$, где $h \in \mathbb{N}$, то полагают $k = 2^h - 1$.

Замечание 2.6. Основная идея ρ-метода очень проста. Если период последовательности $x_i \pmod{n}$ может быть порядка n, то период последовательности $x_i \pmod{p}$ для простого делителя p числа n не превосходит p. Это значит, что x_j и x_k могут быть различными по модулю n, но совпадать по модулю p, т. е. $p \mid \text{НОД}(x_j - x_k, n)$.

Замечание 2.7. Для нахождения периодов рекуррентных последовательностей мы рекомендуем методы, описанные в работе [246]. Именно на этих методах основан оптимальный алгоритм выбора номеров j и k на третьем шаге алгоритма. Более детальное описание реализаций различных выборов j и k см. в [37; 89; 144].

ρ-метод Полларда имеет эвристическую оценку сложности $O(n^{1/4})$ арифметических операций. Он очень популярен и обычно используется для отделения небольших простых делителей факторизуемого числа n. Покажем, как получается оценка его сложности.

Утверждение 2.8. *Пусть S — фиксированное множество из r элементов, f — какое-либо отображение $f\colon S \to S$, $x_0 \in S$, последовательность $x_0, x_1, x_2, \ldots$ определяется соотношением $x_j = f(x_{j-1})$. Пусть $\lambda > 0$, $l = 1 + [\sqrt{2\lambda r}] < r$. Тогда доля тех пар (f, x_0) (где f пробегает все отображения из S в S и x_0 пробегает*

все множество S), у которых $x_0, x_1, x_2, \ldots x_l$ попарно различны, среди всех пар (f, x_0) не превосходит $e^{-\lambda}$.

Доказательство. Всего имеется $r^r \cdot r = r^{r+1}$ различных пар (f, x_0). Пар (f, x_0), у которых $x_0, x_1, x_2, \ldots x_l$ попарно различны будет

$$r(r-1)\ldots(r-l)\cdot r^{r-l}.$$

Доля таких пар составляет величину

$$t = r^{-r-1}\cdot r^{r-l+1}\prod_{j=1}^{l}(r-j) = \prod_{j=1}^{l}\Big(1-\frac{j}{r}\Big).$$

Поскольку при $0<x<1$ выполнено неравенство $\log(1-x)<-x$, то

$$\log t = \sum_{j=1}^{l}\log\Big(1-\frac{j}{r}\Big) < -\sum_{j=1}^{l}\frac{j}{r} = -\frac{l(l+1)}{2r} < -\frac{l^2}{2r} < -\frac{2\lambda r}{2r} = -\lambda,$$

что и требовалось доказать. □

Для чего нужно доказанное утверждение? Если у n есть небольшой простой делитель p, то величина $l = l(n) = 1 + [\sqrt{2\lambda n}]$ имеет порядок $n^{1/2}$, в то время как величина $l = l(p) = 1 + [\sqrt{2\lambda p}]$ существенно меньше. А доля наборов $(f, x_0 \pmod n)$, где $f : \mathbb{Z}/n\mathbb{Z} \longrightarrow \mathbb{Z}/n\mathbb{Z}$, у которых элементы длинного набора $x_0 \pmod n, \ldots, x_{l(n)} \pmod n$ различны, не превосходит той же величины $e^{-\lambda}$, что и доля наборов $(f, x_0 \pmod p)$, где $f : \mathbb{Z}/p\mathbb{Z} \longrightarrow \mathbb{Z}/p\mathbb{Z}$, у которых различны элементы короткого набора $x_0 \pmod p, \ldots, x_{l(p)} \pmod p$. Из этих рассуждений вытекает следующий результат (см. [144]).

Теорема 2.9. *Пусть n — нечетное составное число, p — простой делитель n, $p<\sqrt{n}$, $f(x)\in\mathbb{Z}[x]$, $x_0\in\mathbb{Z}$, причем f хорошо редуцируется к модулю n, т. е. f корректно определяет отображение*

$$f\colon \mathbb{Z}/n\mathbb{Z} \longrightarrow \mathbb{Z}/n\mathbb{Z}.$$

Предположим также, что $f(x)$ хорошо редуцируется к модулю p. Если пара $(f, x_0 \pmod p)$ является не слишком редкой по своим статистическим свойствам, то ρ-метод Полларда найдет p за $O(n^{1/4}\log^3 n)$ битовых операций.

Точнее, существует константа c, такая, что для любого $\lambda>0$ вероятность не найти нетривиальный делитель n за $c\cdot\sqrt{\lambda}n^{1/4}\cdot\log^3 n$ битовых операций будет меньше, чем $e^{-\lambda}$.

Другой метод получения оценки сложности ρ-метода Полларда можно найти в [89, гл. 8]. Рассуждения являются эвристическими

и апеллируют к понятию вероятности, но иного способа получить оценку сложности для ρ-метода Полларда нет.

§ 2.4. Метод Шермана—Лемана

В работе [158] описан алгоритм Шермана—Лемана, детерминированно раскладывающий n на множители за $O(n^{1/3})$ арифметических операций.

Алгоритм.

Пусть n нечетно, $n > 8$.

1 шаг. Для $a = 2, 3, \ldots, [n^{1/3}]$ проверить условие $a \mid n$. Если на этом шаге мы не разложили n на множители, то переходим к шагу 2.

2 шаг. Если на 1 шаге делитель не найден и n — составное, то $n = pq$, где p, q — простые числа, и

$$n^{1/3} < p \leqslant q < n^{2/3}.$$

Тогда для всех $k = 1, 2, \ldots, [n^{1/3}]$ и всех $d = 0, 1, \ldots, [n^{1/6}/(4\sqrt{k})] + 1$ проверить, является ли число

$$([\sqrt{4kn}] + d)^2 - 4kn$$

квадратом натурального числа. Если является, то для $A = [\sqrt{4kn}] + d$ и $B = \sqrt{A^2 - 4kn}$ выполнено сравнение

$$A^2 \equiv B^2 \pmod{n}.$$

В этом случае проверить условие

$$1 < (A \pm B, n) < n.$$

Если это условие выполнено, то мы разложили n на два множителя и алгоритм останавливается.

Конец алгоритма.

Если алгоритм не нашел разложение n на два множителя, то n — простое число. Докажем это. Нам нужно рассмотреть лишь случай $n = pq$, где p, q — простые числа,

$$n^{1/3} < p \leqslant q < n^{2/3}.$$

Лемма 2.10. *При этих условиях существуют натуральные числа r, s такие, что*

$$rs < n^{1/3}, \quad |pr - qs| < n^{1/3}.$$

Воспользуемся этой леммой и положим в алгоритме $k = rs \leqslant [n^{1/3}]$. Тогда

$$4kn = 4rspq = (pr + qs)^2 - (pr - qs)^2.$$

Следовательно,

$$(pr + qs)^2 - 4kn = (pr - qs)^2 = B^2,$$

где $B = |pr - qs| < n^{1/3}$. Пусть

$$d = pr + qs - [\sqrt{4kn}].$$

Тогда

$$n^{2/3} > (pr + qs)^2 - 4kn =$$
$$= (pr + qs + \sqrt{4kn})(pr + qs - \sqrt{4kn}) \geqslant 2\sqrt{4kn}(d - 1).$$

Отсюда получаем

$$d < \frac{n^{2/3}}{4\sqrt{kn}} + 1 = \frac{n^{1/6}}{4\sqrt{k}} + 1.$$

Значит, k и d лежат в установленных в алгоритме пределах. Поэтому в алгоритме мы найдем число $A = pr + qs = [\sqrt{4kn}] + d$ такое, что $B = \sqrt{A^2 - 4kn} = |pr - qs|$ является натуральным числом. При этом $A^2 - B^2 = 4kn \equiv 0 \pmod{n}$. Далее, одно из двух чисел $A \pm B$ равно $2pr$ и имеет с n общий делитель, равный p, так как n нечетно и не делится на все числа, не превосходящие $n^{1/3}$, а $r < n^{1/3}$. То есть мы с помощью НОД$(A \pm B, n)$ разложим n на множители.

Доказательство леммы 2.10. Если $p = q$, т. е. $n = p^2$, то утверждение леммы выполнено для $r = s = 1$. Далее будем считать $p < q$.

Разложим q/p в непрерывную дробь. Мы обозначаем p_j/q_j j-ю подходящую дробь к q/p. Тогда

$$p_0 = [q/p], \quad q_0 = 1, \quad 0 < p_0 q_0 < n^{1/3},$$

поскольку $\dfrac{q}{p} < \dfrac{n^{2/3}}{n^{1/3}} = n^{1/3}$. Выберем первый номер m такой, что

$$p_m q_m < n^{1/3}, \quad p_{m+1} q_{m+1} > n^{1/3}.$$

Такой номер обязательно найдется, поскольку у последней подходящей дроби знаменатель $q_N = p > n^{1/3}$. Докажем, что $r = p_m$ и $s = q_m$ удовлетворяют утверждению леммы. Очевидно, что $rs < n^{1/3}$. Далее,

$$\left|\frac{r}{s} - \frac{q}{p}\right| \leqslant \left|\frac{r}{s} - \frac{p_{m+1}}{q_{m+1}}\right| = \frac{1}{sq_{m+1}}$$

по свойствам подходящих дробей.

Рассмотрим сначала случай $\frac{p_{m+1}}{q_{m+1}} \leqslant \frac{q}{p}$. В этом случае

$$|pr-qs| = ps\left|\frac{r}{s}-\frac{q}{p}\right| \leqslant \frac{ps}{sq_{m+1}} = \frac{p}{q_{m+1}} = \sqrt{\frac{p}{q_{m+1}}\cdot\frac{p}{q_{m+1}}} \leqslant$$
$$\leqslant \sqrt{\frac{p}{q_{m+1}}}\cdot\sqrt{\frac{q}{p_{m+1}}} = \sqrt{\frac{n}{p_{m+1}q_{m+1}}} < \frac{n^{1/2}}{n^{1/6}} = n^{1/3},$$

что и требовалось доказать.

Теперь рассмотрим случай $\frac{p_{m+1}}{q_{m+1}} > \frac{q}{p}$. Тогда перевернем неравенства $\frac{p_{m+1}}{q_{m+1}} > \frac{q}{p} > \frac{p_m}{q_m}$, откуда $\frac{q_m}{p_m} > \frac{p}{q} > \frac{q_{m+1}}{p_{m+1}}$. Следовательно, по свойствам непрерывных дробей, выполнены неравенства

$$\frac{1}{rq} \leqslant \left|\frac{s}{r}-\frac{p}{q}\right| \leqslant \left|\frac{s}{r}-\frac{q_{m+1}}{p_{m+1}}\right| = \frac{1}{rp_{m+1}}.$$

Отсюда

$$1 \leqslant |sq-pr| = rq\left|\frac{s}{r}-\frac{p}{q}\right| \leqslant \frac{rq}{rp_{m+1}} = \frac{q}{p_{m+1}} = \sqrt{\frac{q}{p_{m+1}}\cdot\frac{q}{p_{m+1}}} <$$
$$< \sqrt{\frac{q}{p_{m+1}}}\cdot\sqrt{\frac{p}{q_{m+1}}} = \sqrt{\frac{n}{p_{m+1}q_{m+1}}} < \frac{n^{1/2}}{n^{1/6}} = n^{1/3}.$$

Лемма доказана. □

Замечание 2.11. При реализации алгоритма Шермана—Лемана возможно эффективное использование параллельных вычислений.

§ 2.5. Алгоритм Ленстры

В работе [165] получен следующий результат.

Теорема 2.12. *Пусть r, s, n — натуральные числа, удовлетворяющие условиям*

$$1 \leqslant r < s < n, \quad n^{1/3} < s, \quad (r, s) = 1.$$

Тогда существует не более 11 делителей r_i числа n таких, что $r_i \equiv r \pmod{s}$. Имеется алгоритм, который находит все эти r_i за $O(\log n)$ арифметических операций.

Следствие 2.13. *Используя алгоритм из теоремы 2.12, можно получить метод факторизации числа n за $O(n^{1/3}\log^2 n)$ арифметических операций. Положим $s = [n^{1/3}]+1$. С помощью алгоритма Евклида представим n в виде $n = n_1 n_2$, где $(n_1, s) = 1$, а число n_2 равно произведению степеней тех простых чисел, которые делят s.*

Мы факторизуем n_1, а для факторизации n_2 поступим аналогично, заменив s на $s+1$. Теперь перебираем $r = 1, 2, \ldots, s-1$ и для тех r, которые взаимно просты с s, находим с помощью алгоритма теоремы делители r_i числа n_1, $r_i \equiv r \pmod{s}$. В силу взаимной простоты n_1 и s мы полностью разложим n_1 на множители.

Следствие 2.14. *В § 1.8 главы 1 приведено описание схемы алгоритма Ленстры для проверки простоты натуральных чисел. Оценка сложности этого алгоритма зависит от начальных простых чисел $p_1, \ldots, p_l$, таких, что*

$$s = \prod_{\substack{q \text{ — простое} \\ q-1 \mid p_1 \ldots p_l}} q > n^{1/2}.$$

Теорема 2.12 позволяет уменьшить количество простых чисел $p_1, \ldots, p_l$ и ослабить это неравенство до $s > n^{1/3}$. Затем на последней стадии алгоритма проверки простоты нам придется восстанавливать возможные делители числа n по остаткам $r^i \pmod{s}$.

Это усовершенствование на практике позволяет ускорить работу алгоритма проверки простоты чисел.

Мы докажем здесь теорему 2.12 лишь частично. А именно, мы полностью опишем алгоритм нахождения всех $r_i \equiv r \pmod{s}$ и докажем оценку сложности. Доказательство того, что таких делителей может быть не более 11, является комбинаторным и его можно найти в [165].

Замечание 2.15. Можно доказать, что для любой постоянной α, $\frac{1}{3} > \alpha > \frac{1}{4}$, существует постоянная $c(\alpha) > 0$ такая, что при $1 \leqslant r < s < n$, $(r, s) = 1$, $s > n^{\alpha}$, существует не более $c(\alpha)$ положительных делителей числа n, сравнимых с r по модулю s. Однако полиномиальный алгоритм для их нахождения пока неизвестен.

Алгоритм.

На входе заданы числа r, s, $n \in \mathbb{N}$, удовлетворяющие условиям теоремы.

1 шаг. С помощью обобщенного алгоритма Евклида найти $r^* \in \mathbb{N}$, $r^* r \equiv 1 \pmod{s}$. Найти r' такое, что $r' \equiv r^* n \pmod{s}$, $0 \leqslant r' < s$.

2 шаг. Для очередного значения $i = 0, 1, 2, \ldots$ найти числа a_i, b_i, c_i по следующим правилам:

$$a_0 = s, \quad b_0 = 0, \quad c_0 = 0,$$

$$a_1 \equiv r' r^* \pmod{s}, \quad 0 < a_1 \leqslant s, \quad b_1 = 1, \quad c_1 \equiv \frac{n - rr'}{s} \cdot r^* \pmod{s}$$

и при $i \geqslant 2$

$$a_i = a_{i-2} - q_i a_{i-1}, \quad b_i = b_{i-2} - q_i b_{i-1}, \quad c_i \equiv c_{i-2} - q_i c_{i-1} \pmod{s}.$$

Здесь целые числа q_i однозначно определяются условиями

$$0 \leqslant a_i < a_{i-1} \quad \text{при } i \text{ четном},$$
$$0 < a_i \leqslant a_{i-1} \quad \text{при } i \text{ нечетном}.$$

Фактически, q_i — частное от деления a_{i-2} на a_{i-1}, за исключением случая, когда i нечетно и остаток от деления равен нулю. Отметим, что a_i монотонно не возрастают и на четных номерах — убывают.

3 шаг. Для очередного набора a_i, b_i, c_i найти все целые числа c, удовлетворяющие условиям

$$c = c_i \pmod{s},$$
$$|c| < s, \qquad \text{если } i \text{ четное},$$
$$2a_i b_i \leqslant c \leqslant \frac{n}{s^2} + a_i b_i, \qquad \text{если } i \text{ нечетное}.$$

Таких c будет не более двух; для четного i это очевидно, а для нечетных i мы докажем это ниже.

4 шаг. Для каждого c из шага 3 решить в целых числах систему уравнений

$$\begin{cases} xa_i + yb_i = c, \\ (xs + r)(ys + r') = n. \end{cases}$$

Если x и y окажутся неотрицательными целыми числами, то добавить $xs + r$ к списку искомых делителей.

5 шаг. Если $a_i = 0$, то алгоритм заканчивает работу. Иначе возвращаемся на шаг 2 к следующему значению i.

Конец алгоритма.

Замечание 2.16. Систему линейных уравнений, возникающую на 4 шаге алгоритма, можно свести к одному квадратному уравнению. Положим

$$u = a_i(xs + r), \quad v = b_i(ys + r').$$

Тогда

$$uv = na_i b_i, \quad u + v = s(a_i x + b_i y) + a_i r + b_i r' = cs + a_i r + b_i r',$$

т. е. числа u и v являются корнями многочлена

$$T^2 - (cs + a_i r + b_i r')T + a_i b_i n.$$

Эти корни должны быть целыми числами, одно из которых делится на a_i, а другое — на b_i. Поскольку мы работаем с большими целыми числами, на практике многократное извлечение корня из дискриминанта для нахождения корней многочлена является достаточно трудоемким.

Замечание 2.17. Обозначим через t номер, для которого $a_t = 0$. Поскольку a_i получается с помощью алгоритма Евклида, примененного к $a_0 = s$ и a_1, $a_1 \leqslant s$, то очевидно, что $t = O(\log s)$. Кроме того, по определению чисел a_i номер t четен.

Лемма 2.18. *Числа a_i, b_i обладают следующими свойствами*:

1. $a_i > 0$, $b_i > 0$ *для нечетных* i, $0 < i < t$;
2. $a_i \geqslant 0$, $b_i \leqslant 0$, $(a_i, b_i) \neq (0, 0)$ *для четных* i, $0 \leqslant i \leqslant t$;
3. $b_{i+1}a_i - a_{i+1}b_i = (-1)^i \cdot s$ *для* $0 \leqslant i < t$.

Доказательство. Свойство 1 для $i = 1$, свойства 2 и 3 для $i = 0$ выполняются по определению a_0, b_0, a_1 и b_1. Дальнейшее доказательство проведем по индукции.

Свойство 3 следует из соотношения

$$b_{i+2}a_{i+1} - a_{i+2}b_{i+1} = \\ = (b_i - q_{i+2}b_{i+1})a_{i+1} - (a_i - q_{i+2}a_{i+1})b_{i+1} = -(b_{i+1}a_i - a_{i+1}b_i).$$

Неравенства $a_i > 0$ для нечетных i и $a_i \geqslant 0$ для четных i следуют из определения a_i, а неравенство $(a_i, b_i) \neq (0, 0)$ следует из свойства 3. Докажем неравенства для b_i.

Если i нечетно, то $i < t$. Тогда

$$b_i = b_{i-2} - q_i b_{i-1} > 0,$$

поскольку $b_{i-2} > 0$ и $b_{i-1} \leqslant 0$ по предположению индукции и числа q_i неотрицательны по определению. Точнее, в этом случае справедливо более сильное неравенство: $b_i \geqslant b_{i-2}$.

Пусть i четно, $i \leqslant t$. Тогда по предположению индукции $b_{i-2} \leqslant 0$, $b_{i-1} > 0$, и здесь q_i — настоящее частное, т. е. $q_i \geqslant 1$. Тогда $b_i = b_{i-2} - q_i b_{i-1} < 0$, и даже $b_i < b_{i-2}$. Лемма □

Лемма 2.19. *Пусть a_i, b_i, t — числа из алгоритма. Пусть $x, y \in \mathbb{R}_{\geqslant 0}$, $\gamma \in \mathbb{R}_{>0}$. Тогда найдется номер i, $0 \leqslant i \leqslant t$, такой, что либо*

$$-\gamma s < xa_i + yb_i < \gamma s, \quad \text{если } i \text{ четно},$$

либо

$$2\gamma a_i b_i \leqslant xa_i + yb_i \leqslant \frac{xy}{\gamma} + \gamma a_i b_i, \quad \text{если } i \text{ нечетно}.$$

Доказательство. Если $x = y = 0$, то утверждение леммы очевидно. Пусть далее x или y отлично от нуля. Поскольку по лемме 2.18

$$xa_0 + yb_0 = xs \geqslant 0, \quad xa_t + yb_t = yb_t \leqslant 0,$$

то найдется четный номер i такой, что

$$xa_i + yb_i \geqslant 0, \quad xa_{i+2} + yb_{i+2} \leqslant 0.$$

Если $xa_i + yb_i < \gamma s$ или $xa_{i+2} + yb_{i+2} > -\gamma s$, то все доказано. Если же $xa_i + yb_i \geqslant \gamma s$ и $xa_{i+2} + yb_{i+2} \leqslant -\gamma s$, то по лемме 2.18 имеем

$$\frac{xa_i + yb_i}{\gamma} \geqslant s = b_{i+1}a_i - a_{i+1}b_i \geqslant b_{i+1}a_i.$$

Тогда $xa_i + yb_i \geqslant \gamma b_{i+1}a_i$, откуда в силу неположительности yb_i находим, что $x \geqslant \gamma b_{i+1}$ (заметим, что $a_i \neq 0$, так как $i < t$). Кроме того,

$$\frac{xa_{i+2} + yb_{i+2}}{\gamma} \leqslant -s = b_{i+2}a_{i+1} - a_{i+2}b_{i+1} \leqslant b_{i+2}a_{i+1},$$

откуда $xa_{i+2} + yb_{i+2} \leqslant \gamma b_{i+2}a_{i+1}$, и в силу неотрицательности xa_{i+2} получаем $yb_{i+2} \leqslant \gamma b_{i+2}a_{i+1}$. Заметим, что $b_{i+2} < 0$, так как $xa_{i+2} + yb_{i+2} \leqslant \leqslant -\gamma s < 0$. Тогда $y \geqslant \gamma a_{i+1}$.

Из доказанных оценок снизу для x и y получаем

$$xa_{i+1} + yb_{i+1} \geqslant 2\gamma a_{i+1}b_{i+1},$$

т. е. выполнено нижнее неравенство утверждения леммы для нечетного номера $i + 1$. Также

$$(x - \gamma b_{i+1})(y - \gamma a_{i+1}) \geqslant 0.$$

Поэтому

$$xy - \gamma a_{i+1}x - \gamma b_{i+1}y + \gamma^2 b_{i+1}a_{i+1} \geqslant 0.$$

Следовательно

$$\gamma(a_{i+1}x + b_{i+1}y) \leqslant xy + \gamma^2 b_{i+1}a_{i+1},$$

отсюда следует и верхнее неравенство для номера $i + 1$. □

Теперь докажем, что алгоритм находит все делители числа n, сравнимые с r по модулю s. Пусть $xs + r$ — такой делитель (число $x \in \mathbb{Z}_{\geqslant 0}$ нам неизвестно). Тогда при некотором $d \in \mathbb{N}$ выполнено равенство $(xs + r)d = n$, откуда $dr \equiv n \pmod{s}$ и $d \equiv nr^* \equiv r' \pmod{s}$. Значит, $d = r' + ys$, где $y \in \mathbb{Z}_{\geqslant 0}$, поскольку $r' < s$. Отсюда

$$(xs + r)(ys + r') = n.$$

Тогда

$$rr' + s(xr' + yr) \equiv n \pmod{s^2},$$

и

$$xr' + yr \equiv \frac{n - rr'}{s} \pmod{s}.$$

Отсюда

$$xr'r^* + y \equiv \frac{n - rr'}{s} r^* \pmod{s},$$

т. е.

$$xa_1 + yb_1 \equiv c_1 \pmod{s}.$$

Сравнение $xa_0 + yb_0 \equiv c_0 \pmod{s}$ также выполняется очевидным образом. Тогда с помощью рекуррентных формул находим, что

$$xa_i + yb_i \equiv c_i \pmod{s}, \quad i = 0, \ldots, t.$$

Применим лемму 2.19 при $\gamma = 1$. Найдется i такое, что либо

$$|xa_i + yb_i| < s, \quad \text{если } i \text{ четно},$$

либо

$$2a_ib_i \leqslant xa_i + yb_i \leqslant xy + a_ib_i, \quad \text{если } i \text{ нечетно}.$$

Фиксируем это i и положим $c = xa_i + yb_i$. Тогда $c \equiv c_i \pmod{s}$. Из неравенства

$$xy \leqslant \frac{(xs + r)(ys + r')}{s^2} = \frac{n}{s^2}$$

мы получим, что

$$|c| < s, \qquad \text{если } i \text{ четно},$$
$$2a_ib_i \leqslant c \leqslant \frac{n}{s^2} + a_ib_i, \quad \text{если } i \text{ нечетно}.$$

Значит, c попадает в множество значений, проверяемых на 3 шаге алгоритма. Мы уже заметили ранее, что для четного i таких значений будет не более двух. Для нечетного i число c лежит на отрезке $\left[2a_ib_i, \frac{n}{s^2} + a_ib_i\right]$ длины $\frac{n}{s^2} - a_ib_i < \frac{n}{s^2}$. Поскольку $n/s^3 < 1$, то только один элемент из арифметической прогрессии $c_i \pmod{s}$ может попасть в этот отрезок. Итак, в алгоритме на 3 шаге мы дойдем до этого значения $c = xa_i + yb_i$. Решив систему на 4 шаге, мы найдем x и y, и, следовательно, найдем делитель $xs + r$.

Теперь оценим сложность алгоритма. Так как a_i получаются с помощью алгоритма Евклида, то $t = O(\log n)$. Шаги 2, 3, 4 можно выполнить

за $O(1)$ арифметических операций, поскольку согласно [53] извлечение квадратного корня из целого числа имеет такую же сложность, как умножение. В итоге мы доказали теоретическую оценку сложности $O(\log n)$ арифметических операций, хотя реально неоднократное извлечение квадратного корня из дискриминанта на 4 шаге является трудоемким, как уже отмечалось выше.

§ 2.6. Алгоритм Полларда—Штрассена

Алгоритм Полларда—Штрассена впервые описан в работе [218], (см. также [264]). Он детерминированно находит разложение n на два множителя за $O(n^{1/4}\log^4 n)$ арифметических операций, т. е. имеет наилучшую оценку сложности среди детерминированных алгоритмов факторизации. Алгоритм основан на следующей теореме.

Теорема 2.20. *Пусть $z \in \mathbb{N}$, $y = z^2$. Тогда для любого натурального числа t наименьший простой делитель числа* НОД$(t, y!)$ *может быть найден за $O(z\log^2 z \log^2 t)$ арифметических операций.*

Алгоритм Полларда—Штрассена

Положим $z = [n^{1/4}] + 1$, $y = z^2 > n^{1/2}$, $t = n$. Далее с помощью алгоритма теоремы 2.20 найдем наименьший простой делитель числа НОД$(n, y!)$. Поскольку $y!$ делится на наименьший простой делитель p числа n (так как $p \leqslant n^{1/2} < y$), то алгоритм выдаст именно это число p. Сложность алгоритма Полларда—Штрассена $O(z\log^2 z\log^2 t) = = O(n^{1/4}\log^4 n)$.

Доказательство теоремы 2.20. Справедливо равенство

$$y! = \prod_{j=1}^{z} \frac{(jz)!}{((j-1)z)!}.$$

Если мы будем вычислять

$$\text{НОД}\Big(t,\ \frac{(jz)!}{((j-1)z)!}\Big), \quad j = 1, \ldots, z,$$

то первый нетривиальный НОД покажет, что минимальный простой делитель числа НОД$(t, y!)$ лежит среди чисел

$$(j-1)z+1, \quad (j-1)z+2, \quad \ldots, \quad jz.$$

Тогда первое число в этом наборе, которое делит t будет искомым минимальным простым делителем числа НОД$(t, y!)$; для его нахождения потребуется не более чем z операций деления t на числа данного набора.

Обозначим $f(x) = \prod_{i=0}^{z-1} (x - i)$. Тогда

$$f(jz) = \frac{(jz)!}{((j-1)z)!}.$$

Ниже в главе 9, посвященной дискретному преобразованию Фурье, будет показано, что набор чисел

$$f(jz) \pmod t, \quad j = 1, \ldots, z,$$

можно найти за $O(z \log^2 z \log^2 t)$ арифметических операций. Кроме того, для нахождения первого нетривиального

$$\text{НОД}(t, f(jz) \pmod t), \quad j = 1, \ldots, z,$$

надо затратить $zO(\log t) = O(z \log t)$ арифметических операций. Итоговая оценка сложности составляет

$$O(z \log^2 z \log^2 t) + O(z \log t) + z = O(z \log^2 z \log^2 t),$$

что и завершает доказательство теоремы. □

Алгоритм Полларда—Штрассена может использоваться как непосредственно для факторизации не очень больших целых чисел, так и в качестве вспомогательного алгоритма в дополнительной стратегии PS в субэкспоненциальных алгоритмах факторизации целых чисел. Об этом будет рассказано в следующей главе.

§ 2.7. $(P + 1)$-метод Уильямса и его обобщения

В работе [283] описан метод факторизации чисел $n \in \mathbb{N}$ с помощью последовательностей чисел Люка. Этот метод аналогичен $(P - 1)$-методу Полларда, но использует разложение на множители числа $P + 1$. В работе [59] метод был обобщен на основе использования произвольных круговых многочленов. Суть $(P + 1)$-метода заключается в следующем.

Рассматривается последовательность чисел Люка, определяемая соотношениями

$$u_0 = 0, \quad u_1 = u, \quad u_{n+1} = Pu_n - Qu_{n-1},$$

где P, Q — фиксированные целые числа. Пусть p — простой делитель факторизуемого натурального числа n такой, что $p + 1$ является B-степенно-гладким, т. е.

$$p = \prod_{i=1}^{k} q_i^{\alpha_i} - 1,$$

где q_i — простые числа, $q_i^{\alpha_i} \leqslant B$. Пусть числа $\beta_i \in \mathbb{N}$ таковы, что

$$q_i^{\beta_i} \leqslant B, \quad q_i^{\beta_i+1} > B, \quad i = 1, \ldots, k.$$

Положим $R = \prod_{i=1}^{k} q_i^{\beta_i}$. Тогда $p+1 \mid R$. Если для последовательности чисел Люка параметр Q взаимно прост с n и выполнено условие

$$\left(\frac{P^2 - 4Q}{p}\right) = -1$$

(оба эти условия эвристически обеспечиваются некоторым случайным перебором P и Q), то по свойствам последовательностей чисел Люка

$$p \mid \text{НОД}(u_R, n).$$

Дальнейшая работа алгоритма заключается в достаточно быстром вычислении элемента последовательности u_R и нахождении НОД(u_R, n).

В работе [283] указано, что данный метод является довольно-таки медленным на практике. В работе [59] доказано, что для обобщения $(P+1)$-метода Уильямса с применением круговых многочленов в предположении расширенной гипотезы Римана может быть доказана вероятностная полиномиальная оценка сложности.

§ 2.8. Методы Шэнкса

Два метода факторизации целых чисел, использующих бинарные квадратичные формы, принадлежат Д. Шэнксу, см. [233; 248; 284]. Первый из них работает с положительно определенными бинарными квадратичными формами заданного отрицательного дискриминанта, и в группе классов форм он находит амбигову форму, которая дает разложение дискриминанта на множители. Сложность этого метода составляет $O(n^{1/5+\varepsilon})$ арифметических операций при условии выполнения расширенной гипотезы Римана. Второй метод носит название SQUFOF и использует группу классов бинарных квадратичных форм с положительным дискриминантом. Здесь также происходит нахождение амбиговой формы, дающей разложение дискриминанта. Сложность SQUFOF составляет $O(n^{1/4+\varepsilon})$ арифметических операций; при этом алгоритм работает с целыми числами, не превосходящими $2\sqrt{n}$. Среди алгоритмов факторизации с экспоненциальной сложностью SQUFOF считается одним из самых эффективных.

Детальное описание алгоритмов Шэнкса выходит за рамки данной книги, поскольку требует привлечения ряда фактов из теории бинарных квадратичных форм. Для справок см. [84; 89, гл. 8].

§ 2.9. Прочие методы. Заключение

Для чисел n специального вида возможны особые подходы к факторизации, поскольку делители таких чисел могут также иметь специальный вид.

Теорема 2.21. *Пусть* b, $k \in \mathbb{N}$, $b > 1$, $n = b^k - 1$. *Если* p — *простое число, делящее* n, *то выполнено одно из двух утверждений*:

1. $p \mid b^d - 1$ *при некотором* $d < k$, $d \mid k$;
2. $p \equiv 1 \pmod{k}$.

При этом, если $p > 2$ *и* k — *нечетно, то во втором случае* $p \equiv 1 \pmod{2k}$.

Доказательство. По малой теореме Ферма $b^{p-1} \equiv 1 \pmod{p}$, а также $b^k \equiv 1 \pmod{p}$. Пусть $d = \text{НОД}(k, p-1)$, тогда $b^d \equiv 1 \pmod{p}$. Если $d < k$, то это означает выполнение первого утверждения теоремы. Если же $d = k$, то $k \mid p - 1$, т. е. $p \equiv 1 \pmod{k}$. □

Пример 2.22. Пусть $n = 2^{11} - 1$. Тогда первое утверждение теоремы не выполняется, так как 11 — простое число и $d = 1$ не подходит ($2^1 - 1 = 1$). Следовательно, $p \equiv 1 \pmod{22}$. После этого сразу находим $n = 23 \cdot 89$.

Обзор методов факторизации с экспоненциальной сложностью можно найти в работе [273]. Более поздние работы см. в списках литературы книг [60; 89]. Заметим, что в печати достаточно часто появляются новые работы, содержащие алгоритмы факторизации с экспоненциальной сложностью. Однако практическая значимость таких алгоритмов, как правило, невелика. Наиболее популярными в практических вычислениях являются $(P-1)$-метод Полларда, ρ-метод Полларда и алгоритм Полларда—Штрассена. Они используются в сочетании с субэкспоненциальными методами факторизации, которые будут описаны в главе 3, и применяются, как правило, для предварительного отделения небольших простых делителей у факторизуемого числа.

Глава 3. Факторизация целых чисел с субэкспоненциальной сложностью

§ 3.1. Введение

В данной главе мы рассматриваем алгоритмы факторизации натуральных чисел n, делающие $L_n[\gamma; c]$ арифметических операций при $\gamma = \frac{1}{2}$ или $\gamma = \frac{1}{3}$ и при некоторой положительной, зависящей от алгоритма, постоянной c. Алгоритмов факторизации, имеющих оценки сложности лучше указанных, в настоящее время не существует.

Мы предполагаем далее, что число n — составное (это быстро проверяется с помощью вероятностных алгоритмов из главы 1) и что n не делится на небольшие простые числа (все небольшие простые делители мы находим перебором, а также с помощью алгоритмов из главы 2).

Описываемые далее алгоритмы находят натуральные числа x, y такие, что $x^2 \equiv y^2 \pmod{n}$, и затем проверяют условие

$$1 < \text{НОД}(x \pm y,\ n) < n.$$

Если делитель n найден, то алгоритм останавливается, иначе строит следующую пару x, y.

Утверждение 3.1. *Пусть n — нечетное составное число, не являющееся степенью простого числа. Тогда для случайной пары x, y, $1 \leqslant x$, $y \leqslant n-1$, удовлетворяющей соотношениям*

$$\text{НОД}(x,\ n) = \text{НОД}(y,\ n) = 1,$$
$$x^2 = y^2 \pmod{n},$$

вероятность того, что

$$1 < \text{НОД}(x \pm y,\ n) < n,$$

будет не меньше $1/2$.

Доказательство. Пусть $n = p_1^{\alpha_1} \dots p_k^{\alpha_k}$, $k \geqslant 2$, есть разложение n на простые сомножители. Паре x, y, удовлетворяющей условиям утверждения, соответствует число z, $1 \leqslant z \leqslant n-1$, $z^2 \equiv 1 \pmod{n}$ (очевидно,

$z = xy^{-1} \pmod n$). Нам достаточно доказать, что количество таких z, для которых дополнительно выполнено неравенство

$$1 < \text{НОД}(z \pm 1,\ n) < n,$$

будет не меньше половины. Очевидно, что условие $z^2 \equiv 1 \pmod n$ равносильно совокупности условий

$$\begin{gathered} z \equiv \pm 1 \pmod{p_1^{\alpha_1}}, \\ \dots\dots\dots\dots\dots \\ z \equiv \pm 1 \pmod{p_k^{\alpha_k}}, \end{gathered}$$

где знаки $\pm$ произвольны. Отсюда количество возможных значений для z равно 2^k, и только для двух значений $z \equiv \pm 1 \pmod n$ наибольший общий делитель $\text{НОД}(z \pm 1,\ n)$ будет равен 1 или n. Поскольку $k \geqslant 2$, то наше утверждение теперь очевидно. $\square$

Построение пар x, y таких, что $x^2 = y^2 \pmod n$, будет неэффективным для факторизации n, равного степени простого числа p, т. е. $n = p^\alpha$. Такие числа n довольно редки. Для их обнаружения можно предложить следующий способ. Если $a \in \mathbb{N}$, $p \nmid a$, то $a^p \equiv a \pmod p$. Отсюда $a^n \equiv a \pmod p$, и поэтому $\text{НОД}(a^n - a,\ n) \vdots p$, т. е. $\text{НОД}(a^n - a,\ n) > 1$. Поэтому, если для некоторого случайно выбранного a выполнено равенство $\text{НОД}(a^n - a,\ n) = 1$, то $n \neq p^\alpha$. Если же для нескольких значений a окажется $\text{НОД}(a^n - a,\ n) > 1$, то мы либо разложим n на множители (если $\text{НОД}(a^n - a,\ n) < n$), либо будем считать n степенью простого числа и попробуем факторизовать его, извлекая корни степени $2, 3, 5, 7, \dots$ из n.

В описываемых далее алгоритмах мы будем считать дополнительно, что n не является степенью простого числа.

Везде в этой главе, за исключением двух последних параграфов, мы обозначаем $L = L(n) = \exp((\log n \log\log n)^{1/2})$. Мы также будем предполагать, что n не делится на простые числа p такие, что $p \leqslant L(n)$; это легко проверяется с помощью перебора за $O(L(n)) = L_n\left[\frac{1}{2},\ 1\right]$ арифметических операций.

§ 3.2. Метод Диксона. Дополнительные стратегии

Пусть $n \in \mathbb{N}$ — число, которое мы хотим разложить на множители, $L = L(n) = \exp((\log n \log\log n)^{1/2})$. Пусть a — некоторая постоянная, $0 < a < 1$, значение которой будет определено ниже. *Факторной базой*

мы будем называть множество простых чисел p, лежащих в промежутке

$$2 \leqslant p \leqslant L^a.$$

Пусть k — количество простых чисел в факторной базе, $2 = p_1 < p_2 < \ldots < p_k \leqslant L^a$. Символом $Q(m)$ мы будем обозначать наименьший неотрицательный вычет в классе $m^2 \pmod n$.

Опишем алгоритм Диксона, следуя работе [221].

Алгоритм Диксона.

1 шаг. Случайным перебором ищем числа $m_1, \ldots, m_{k+1}$ такие, что

$$1 < m_i < n, \qquad Q(m_i) = p_1^{\alpha_{i,1}} \ldots p_k^{\alpha_{i,k}}$$

для $i = 1, \ldots, k+1$. Это означает, что $m_i^2 \equiv Q(m_i) \pmod n$ являются гладкими числами, то есть раскладываются на множители в нашей факторной базе. Обозначим $\bar v_i = (\alpha_{i,1}, \ldots, \alpha_{i,k}) \in \mathbb{Z}^k$ — векторы показателей в разложении $Q(m_i)$ на множители.

2 шаг. Решая систему линейных уравнений

$$x_1 \bar v_1 + \ldots + x_{k+1} \bar v_{k+1} \equiv \bar 0 \pmod 2$$

в векторном пространстве $(\mathbb{Z}/2\mathbb{Z})^k$, находим значения $x_1, \ldots, x_{k+1} \in \{0, 1\}$, не все равные нулю (такое решение существует, поскольку число уравнений k меньше числа неизвестных).

3 шаг. Для найденных $x_1, \ldots, x_k$ справедливо соотношение

$$(m_1^{x_1} \ldots m_{k+1}^{x_{k+1}})^2 \equiv p_1^{\sum\limits_{i=1}^{k+1} x_i \alpha_{i,1}} \ldots p_k^{\sum\limits_{i=1}^{k+1} x_i \alpha_{i,k}} \pmod n.$$

Обозначая

$$X = m_1^{x_1} \ldots m_{k+1}^{x_{k+1}}, \qquad Y = \prod_{j=1}^{k} p_j^{\left(\sum\limits_{i=1}^{k+1} x_i \alpha_{i,j}\right)\Big/2},$$

(числа $\left(\sum\limits_{i=1}^{k+1} x_i \alpha_{i,j}\right)\Big/2$ — целые по определению x_i), мы получим соотношение

$$X^2 \equiv Y^2 \pmod n.$$

Далее проверяем условие

$$1 < \text{НОД}(X \pm Y, n) < n.$$

В случае успеха мы разложили n на множители (вероятность успеха была оценена в § 3.1). В случае неудачи мы возвращаемся на 1 шаг и ищем другие значения m_i.

Конец алгоритма.

Замечание 3.2. Факторизация чисел $Q(m_i)$ на 1 шаге алгоритма Диксона проводится пробными делениями на элементы факторной базы.

Замечание 3.3. Решение системы линейных уравнений на 2 шаге алгоритма можно проводить методом Гаусса.

Проанализируем сложность алгоритма Диксона, следуя работе [221]. При этом примем следующее соглашение. Вместо $L^{\text{const}+o(1)}$ будем писать просто L^{const}; величина $o(1)$ все равно будет присутствовать в итоговой оценке сложности $L_n\left[\frac{1}{2}; c\right]$. Очевидно, что $\text{const} \cdot L^{\text{const}} = L^{\text{const}}$ по нашему определению; также $\pi(L^{\text{const}}) = L^{\text{const}}$, где $\pi(x)$ — число простых чисел, не превосходящих x.

Для одного случайно выбранного значения m на 1 этапе алгоритма поиск разложения $Q(m)$ в нашей факторной базе составит L^a арифметических операций. Действительно, простых чисел $p \leqslant L^a$ в факторной базе будет $k = \pi(L^a) = L^a$ по нашему соглашению. Простое число p может входить в разложение $Q(m)$ с кратностью $\alpha_p \leqslant \log_p Q(m) \leqslant \log_2 n$. Следовательно, общее количество операций деления для разложения $Q(m)$ на множители не будет превосходить величины

$$L^a \cdot \log_2 n = L^a \cdot \frac{e^{\log\log n}}{\log 2} = L^a \cdot L^{o(1)} = L^a$$

по нашему соглашению.

Пусть мы делали случайный выбор m на 1 этапе L^b раз для некоторой постоянной b. Тогда будет потрачено L^{a+b} арифметических операций на разложение $Q(m)$ на множители. Если мы нашли $k+1$ значение m_i, то на 2-м этапе, решая систему линейных уравнений от $k+1 = L^a + 1 = L^a$ неизвестных методом Гаусса, мы потратим еще L^{3a} арифметических операций. В работе [221] с помощью теорем о распределении простых чисел показано, что при $b = a + \frac{1}{2a}$ за L^b выборов числа m мы с высокой вероятностью найдем $\pi(L^a) + 1 = L^a = k+1$ значений m_i таких, что $Q(m_i)$ раскладываются на множители из нашей факторной базы. Тогда суммарная оценка сложности алгоритма Диксона составит

$$L^{\max(a+b,3a)} = L^{\max\left(2a+\frac{1}{2a},3a\right)}$$

арифметических операций. Минимум величины $\max\left(2a + \frac{1}{2a}, 3a\right)$ на интервале $(0; +\infty)$ достигается и равен 2. В итоге получаем оценку

сложности алгоритма при $a = \frac{1}{2}$:

$$L^2 = L_n\left[\frac{1}{2};\, 2\right]$$

арифметических операций.

Вывод. Если факторная база состоит из всех простых чисел $p \leqslant L(n)^{1/2}$ и мы случайно выбираем $L(n)^{3/2}$ значений m, то с высокой вероятностью будет построена пара x, $y \in \mathbb{N}$ такая, что $x^2 = y^2 \pmod n$. При этом будет выполнено $L_n\left[\frac{1}{2};\, 2\right]$ арифметических операций (если на 2 этапе алгоритма Диксона использовать обычное гауссово исключение для решения линейных систем).

Теперь обсудим дополнительные стратегии ускоряющие работу алгоритма Диксона (см. [221]).

Стратегия LP (использование больших простых чисел)

Эта стратегия была впервые предложена в работе [79]. Обычно она всегда используется на практике в субэкспоненциальных алгоритмах факторизации целых чисел и алгоритмах дискретного логарифмирования в конечных простых полях.

Суть стратегии LP заключается в следующем. Пусть мы раскладываем на множители $Q(m) = m^2 \pmod n$; поделили на все простые $p \leqslant L^a$, и у $Q(m)$ еще осталась неразложенная часть s, то есть

$$Q(m) = s \cdot \prod_{p \leqslant L^a} p^{\alpha_p(m)}.$$

Очевидно, что $s > L^a$. Если дополнительно выполняется неравенство $s \leqslant L^{2a}$, то s — простое число (иначе у s был бы простой делитель, не превосходящий $\sqrt{s} \leqslant L^a$). Тогда мы включаем дополнительное большое простое число s в факторную базу и храним те значения m, для которых $Q(m)$ делится на данное s. Это увеличивает длину векторов-показателей на 1 этапе алгоритма. Для того, чтобы вернуться к исходной длине векторов k, после выполнения 1 этапа следует провести исключение дополнительных больших простых чисел. А именно, если в нашем списке есть дополнительное большое простое число s, и только одно $Q(m)$ делится на него, то мы вычеркиваем s и $Q(m)$ из списка. Если же, например, есть два значения $Q(m_1)$ и $Q(m_2)$, делящиеся на s, то значение $Q(m_1) \cdot Q(m_2) \equiv (m_1 m_2)^2 \pmod n$ будет делиться на s^2. Следовательно, показатель числа s войдет в вектор

показателей в четной степени и будет отсутствовать в системе линейных уравнений над $\mathbb{Z}/2\mathbb{Z}$.

Можно использовать стратегию LP с двумя, тремя и т. д. дополнительными большими простыми числами, то есть искать такие $m \in \mathbb{N}$, для которых $m^2 \equiv Q(m) \pmod{n}$ и

$$Q(m) = \prod_{p \leqslant L^a} p^{\alpha_p(m)} \cdot s_1 s_2 \dots s_t,$$

где $s_1, \dots, s_t$ — дополнительные большие простые числа, не содержащиеся в факторной базе. В этом случае для исключения дополнительных простых чисел используется теория графов: строится граф, в котором затем ищутся циклы, дающие четную степень произведений $s_1, \dots, s_t$.

Замечание 3.4. Теоретическая оценка сложности алгоритма Диксона с применением стратегии LP не улучшается и по-прежнему составляет $L_n\left[\frac{1}{2}; 2\right]$ арифметических операций.

Стратегия PS (применение алгоритма Полларда—Штрассена)

В §2.6 мы описали алгоритм Полларда—Штрассена, который для $z \in \mathbb{N}$ и $y = z^2$ находит наименьший простой делитель числа НОД$(t, y!)$ за $O(z \log^2 z \log^2 t)$ арифметических операций.

Применительно к алгоритму Диксона мы хотим в числе $Q(m)$ выделить часть разложения на простые множители, состоящую из простых чисел p, $p \leqslant L^a$. Тогда полагаем $z = [L^{a/2}] + 1$, $y = z^2 \asymp L^a$, $y \geqslant L^a$ и применяем алгоритм Полларда—Штрассена не более, чем $\log_2 n$ раз при $t = Q(m)$. В итоге мы выделим у числа $Q(m)$ разложение на $p \leqslant L^a$ за $O(\log n \cdot L^{a/2} \log^4 n) = L^{a/2}$ арифметических операций (по нашему соглашению). Следовательно, оценка сложности алгоритма Диксона составит

$$L^{\max(\frac{a}{2}+b, 3a)}$$

арифметических операций.

Теорема 3.5. *Сложность алгоритма Диксона со стратегией PS минимальна при* $a = \frac{1}{\sqrt{3}}$ *и* $b = a + \frac{1}{2a}$ *и составляет* $L_n\left[\frac{1}{2}; \sqrt{3}\right]$ *арифметических операций.*

Стратегия EAS (стратегия раннего обрыва)

Пусть a, c, θ — некоторые фиксированные постоянные, $0 < a$, c, $\theta < 1$. В алгоритме Диксона мы факторизовали $Q(m)$ пробными делени-

ями на $p \leqslant L^a$. В стратегии EAS мы сначала делаем пробные деления $Q(m)$ на $p \leqslant L^{a\theta}$, и если после этого неразложенная часть $Q(m)$ остается больше, чем n^{1-c}, то мы отбрасываем данное m и переходим к следующему. В работе [221] проведен детальный анализ стратегии EAS.

Теорема 3.6. *Алгоритм Диксона со стратегией* EAS *при* $a = \sqrt{\frac{2}{7}}$, $c = \frac{1}{7}$, $\theta = \frac{1}{2}$ *делает* $L_n\left[\frac{1}{2}; \sqrt{\frac{7}{2}}\right]$ *арифметических операций. При этом мы перебираем* m *в количестве* L^b *значений при*

$$b = a + \frac{c}{2\theta a} + \frac{1-c}{2a}.$$

Замечание 3.7. Можно проводить стратегию EAS с несколькими обрывами, то есть при некоторой возрастающей последовательности θ_i и убывающей последовательности c_i.

Методы исключения

Имеются более тонкие, чем алгоритм Гаусса, методы решения линейных систем уравнений над конечными полями. О них будет рассказано в последней главе книги. Например, в работе [98] предложен алгоритм решения систем линейных уравнений от k неизвестных в поле $\mathbb{Z}/2\mathbb{Z}$ за $O(k^{2{,}495548})$ арифметических операций.

Теорема 3.8. *Алгоритм Диксона со стратегиями* PS, EAS *с несколькими обрывами и алгоритмом решения линейной системы уравнений от* k *неизвестных за* $O(k^\lambda)$ *арифметических операций при* $\lambda < 5/2$ *имеет оценку сложности* $L_n\left[\frac{1}{2}; \sqrt{\frac{5}{2}}\right]$ *арифметических операций.*

Замечание 3.9. Алгоритм Диксона представляет собой очень удобную модель для получения теоретических оценок сложности без использования каких-либо эвристических и недоказанных гипотез. Однако, на практике он не применяется; вместо случайного выбора m используются другие подходы, о которых мы расскажем в следующих параграфах.

§ 3.3. Алгоритм Бриллхарта—Моррисона

В алгоритме Бриллхарта—Моррисона случайный выбор m из алгоритма Диксона заменяется на детерминированное определение очередного значения m, для которого мы ищем разложение $m^2 \pmod n$ на простые множители из факторной базы. Этот выбор m делается

с помощью непрерывных дробей для числа $\sqrt{n}$. Алгоритм впервые был описан в работе [79]; с его помощью было разложено на множители число Ферма F_7. Данный метод факторизации был наиболее популярным до появления в 1981 г. алгоритма квадратичного решета Померанса, о котором будет рассказано в следующем параграфе. Об алгоритме Бриллхарта—Моррисона см. также [25; 285].

Алгоритм основан на следующей теореме.

Теорема 3.10. *Пусть* $n \in \mathbb{N}$, $n > 16$, $\sqrt{n} \notin \mathbb{N}$. *Обозначим через* $\frac{p_i}{q_i}$, $i = 0, 1, 2, \ldots$ *подходящие дроби для разложения* $\sqrt{n}$ *в непрерывную дробь. Тогда абсолютно наименьший вычет* $p_i^2 \pmod n$ *равен* $p_i^2 - nq_i^2$ *и выполняется неравенство*

$$|p_i^2 - nq_i^2| < 2\sqrt{n}.$$

Доказательство. Обозначим $x = \sqrt{n} > 1$. Тогда по свойствам непрерывных дробей получим

$$|p_i^2 - nq_i^2| = q_i^2 \cdot \left|x - \frac{p_i}{q_i}\right| \cdot \left|x + \frac{p_i}{q_i}\right| < \\ < q_i^2 \cdot \left|\frac{p_{i+1}}{q_{i+1}} - \frac{p_i}{q_i}\right| \cdot \left(x + \frac{p_i}{q_i}\right) = \frac{q_i}{q_{i+1}}\left(x + \frac{p_i}{q_i}\right).$$

Далее,

$$x + \frac{p_i}{q_i} < x + x + \left|\frac{p_{i+1}}{q_{i+1}} - \frac{p_i}{q_i}\right| = 2x + \frac{1}{q_i q_{i+1}},$$

поскольку $\frac{p_{i+1}}{q_{i+1}}$ и $\frac{p_i}{q_i}$ расположены по разные стороны от x. Следовательно,

$$|p_i^2 - x^2 q_i^2| - 2x < 2x\left(-1 + \frac{q_i}{q_{i+1}} + \frac{1}{2xq_{i+1}^2}\right) < \\ < 2x\left(-1 + \frac{q_i}{q_{i+1}} + \frac{1}{q_{i+1}}\right) = 2x\left(-1 + \frac{q_i + 1}{q_{i+1}}\right) \leqslant 0,$$

если $i \geqslant 1$ (так как тогда $q_i + 1 \leqslant q_{i+1}$). Отсюда при $i \geqslant 1$ получим

$$|p_i^2 - nq_i^2| < 2x = 2\sqrt{n} < \frac{n}{2},$$

так как $n > 16$. Это означает, что при $i \geqslant 1$ выполнено утверждение теоремы 3.10.

При $i = 0$, $p_0 = [\sqrt{n}]$, $q_0 = 1$ и при $\varepsilon = \{\sqrt{n}\}$ получим

$$|[\sqrt{n}]^2 - n| = |(\sqrt{n} - \varepsilon)^2 - n| = |-2\varepsilon\sqrt{n} + \varepsilon^2| = \varepsilon(2\sqrt{n} - \varepsilon) < 2\sqrt{n}.$$

Теорема доказана. □

Разложение $\sqrt{n}$ в непрерывную дробь легко может быть найдено по следующей теореме.

Теорема 3.11. *Пусть* α *— квадратичная иррациональность,* $\alpha = \dfrac{\sqrt{D}-u}{v}$, *где* $D \in \mathbb{N}$, $\sqrt{D} \notin \mathbb{N}$, $v \in \mathbb{N}$, $u \in \mathbb{Z}$, $v \mid D^2 - u$. *Тогда для любого* $k \geqslant 0$ *справедливо разложение в непрерывную дробь*

$$\alpha = [a_0, a_1, \ldots, a_k, \alpha_{k+1}],$$

где $a_0 \in \mathbb{Z}$, $a_1, \ldots, a_k \in \mathbb{N}$, α_{k+1} *—* $(k+1)$*-й остаток. При этом справедливы соотношения*:

$$a_0 = [\alpha], \quad v_0 = v, \quad u_0 = u + a_0 v$$

и при $k \geqslant 0$

$$a_{k+1} = [\alpha_{k+1}], \quad \text{где} \quad v_{k+1} = \frac{D - u_k^2}{v_k} \in \mathbb{Z}, \quad v_{k+1} \neq 0, \quad \alpha_{k+1} = \frac{\sqrt{D}+u_k}{v_{k+1}} > 1,$$

а числа u_k *получаются с помощью рекуррентной формулы*

$$u_{k+1} = a_{k+1} v_{k+1} - u_k.$$

Доказательство. Проведем индукцию по k. При $k = 0$ имеем

$$\alpha_1 = \frac{1}{\alpha - a_0} = \frac{1}{\dfrac{\sqrt{D}-u}{v} - a_0} = \frac{v}{\sqrt{D} - u - v a_0} =$$

$$= \frac{v_0}{\sqrt{D} - u_0} = \frac{\sqrt{D}+u_0}{(D-u_0^2)/v_0} = \frac{\sqrt{D}+u_0}{v_1}.$$

Число v_1 — целое, поскольку

$$\frac{D-u_0^2}{v_0} = \frac{D-(u+a_0 v)^2}{v} = \frac{D-u^2}{v} - 2a_0 u - a_0^2 v \in \mathbb{Z}$$

по условию. Очевидно также, что $v_1 \neq 0$ так как $D \neq u^2$.

Пусть формулы теоремы 3.11 справедливы для номера $k+1$. Докажем, что они выполнены и для $k+2$. По определению $(k+2)$-го остатка выполнены равенства

$$\alpha_{k+2} = \frac{1}{\alpha_{k+1} - a_{k+1}} = \frac{1}{\dfrac{\sqrt{D}+u_k}{v_{k+1}} - a_{k+1}} = \frac{v_{k+1}}{\sqrt{D} + u_k - a_{k+1} v_{k+1}} =$$

$$= \frac{v_{k+1}}{\sqrt{D} - u_{k+1}} = \frac{\sqrt{D}+u_{k+1}}{(D-u_{k+1}^2)/v_{k+1}} = \frac{\sqrt{D}+u_{k+1}}{v_{k+2}}.$$

По определению неполного частного $a_{k+2} = [\alpha_{k+2}]$. Осталось лишь доказать, что $v_{k+2} = \dfrac{D - u_{k+1}^2}{v_{k+1}}$ является целым числом. Число

$$v_{k+2} = \frac{D - u_{k+1}^2}{v_{k+1}} = \frac{D - (a_{k+1}v_{k+1} - u_k)^2}{v_{k+1}}$$

будет целым тогда и только тогда, когда целым будет число $\dfrac{D - u_k^2}{v_{k+1}} = v_k$; но v_k — целое по предположению индукции. □

Следствие 3.12. *По теореме 3.11 мы найдем разложение $\sqrt{n}$ в непрерывную дробь, используя только операции с целыми числами и нахождения целой части чисел вида $\dfrac{\sqrt{D} + u}{v}$.*

Алгоритм Бриллхарта—Моррисона.

Будем обозначать через P_i/Q_i подходящие дроби разложения $\sqrt{n}$ в непрерывную дробь (чтобы не путать их с элементами факторной базы $\{p_i\}$). Факторную базу составляют $p_0 = -1$ и простые числа p_i, $p_i \leqslant L(n)^a = L^a$, где $a = \text{const}$; при этом мы берем лишь те простые числа, для которых $\left(\dfrac{n}{p_i}\right) = +1$. Алгоритм Бриллхарта—Моррисона работает так же как алгоритм Диксона, только вместо случайных значений m мы берем $m_i = P_i$, и по теореме 3.10 абсолютно наименьший вычет $m_i^2 \pmod n$ равен

$$Q(m_i) = P_i^2 - nQ_i^2,$$

причем $|Q(m_i)| < 2\sqrt{n}$. В последнем неравенстве и заключается основная идея алгоритма: если в методе Диксона $Q(m) < n$, то в методе Бриллхарта—Моррисона $|Q(m)|$ существенно меньше (не превосходит $2\sqrt{n}$). Следовательно, вероятность того, что $Q(m)$ будет гладким (т. е. разложится в нашей факторной базе), из эвристических соображений значительно выше.

Замечание 3.13. Если p — простое число из факторной базы и $p \mid Q(m_i)$ для некоторого номера i, то $p \nmid Q_i$, поскольку $(P_i, Q_i) = 1$. Тогда $Q_i^2 n \equiv P_i^2 \pmod p$, откуда и следует условие $\left(\dfrac{n}{p}\right) = +1$.

Приведем оценки сложности алгоритма Бриллхарта—Моррисона, следуя работе [221]. При получении этих оценок делаются некоторые эвристические допущения. Например, предполагают, что если с помощью алгоритма построено $1 + [\log^2 n]$ пар (x, y) таких, что $x^2 \equiv y^2 \pmod n$, то хотя бы для одной из них выполнены неравенства

$$1 < \text{НОД}(x \pm y, n).$$

Утверждение 3.14. *Если* $a = 1/\sqrt{2}$ *и факторная база состоит из* $p = -1$ *и всех простых чисел* p *таких, что*

$$2 \leqslant p \leqslant L^a, \quad \left(\frac{n}{p}\right) = +1,$$

то при вычислении L^b *подходящих дробей* $\left(\text{где } b = a + \frac{1}{4a}\right)$ *можно ожидать, что алгоритм разложит* n *на два множителя с эвристической оценкой сложности* $L_n\left[\frac{1}{2}; \sqrt{2}\right]$ *арифметических операций.*

Утверждение 3.15. *В условиях утверждения* 3.14 *использование стратегии LP дает ту же оценку сложности алгоритма (но на практике эту стратегию следует использовать обязательно).*

Утверждение 3.16. *Использование стратегии* PS *при* $a = \frac{1}{\sqrt{6}}$, $b = a + \frac{1}{4a}$ *дает эвристическую оценку сложности алгоритма* $L_n\left[\frac{1}{2}; \sqrt{\frac{3}{2}}\right]$ *арифметических операций.*

Утверждение 3.17. *Использование стратегий* PS, EAS *с* k *обрывами и алгоритмом решения линейной системы уравнений от* k *неизвестных в* $\mathbb{Z}/2\mathbb{Z}$ *за* $O(k^\lambda)$ *арифметических операций при* $\lambda \leqslant 5/2$ *дает эвристическую оценку сложности алгоритма* $L_n\left[\frac{1}{2}; \sqrt{\frac{5}{4}}\right]$ *арифметических операций.*

Метод Бриллхарта—Моррисона существенно менее эффективен, чем метод квадратичного решета, о котором мы расскажем в следующем параграфе.

§ 3.4. Квадратичное решето

Алгоритм квадратичного решета был предложен К. Померансом в начале 1981 г. (см. [221; 222; 225]). Ряд усовершенствований этого метода был предложен впоследствии в работах [66; 86; 93; 213; 229; 258; 266], см. также [89].

Эвристическая оценка сложности усовершенствованного алгоритма квадратичного решета составляет $L_n\left[\frac{1}{2}; 1\right]$ арифметических операций. Рекордное значение для факторизованных этим методом чисел составляет 129-значное RSA-число n (см. [58]).

Опишем схему исходного алгоритма квадратичного решета Померанса. Мы по-прежнему будем строить соотношения $X^2 = Y^2 \pmod n$ и проверять выполнение неравенства:

$$1 < \text{НОД}(X \pm Y,\ n) < n.$$

Для этого рассмотрим многочлен

$$Q(x) = (x + [\sqrt{n}])^2 - n \equiv H(x)^2 \pmod n,$$

где $H(x) = x + [\sqrt{n}]$. Значения $Q(x)$ в целых точках, очевидно, являются квадратами по модулю n. Коэффициенты $Q(x)$ невелики, порядка $n^{1/2}$. В качестве факторной базы S рассматриваем $p_0 = -1$ и все простые числа p_i, $p_i \leqslant B$, такие, что $\left(\dfrac{n}{p_i}\right) = +1$. Затем с помощью некоторого просеивания мы находим значения x_i, для которых

$$A_i = Q(x_i) = \prod_{p \in S} p^{\alpha_{i_p}},$$

т. е. $Q(x_i)$ раскладывается в нашей факторной базе. Тогда, обозначая $B_i = H(x_i)$, получаем сравнение $B_i^2 \equiv A_i \pmod n$, и, накопив достаточно много таких соотношений, мы проводим исключение переменных и построим соотношение $X^2 \equiv Y^2 \pmod n$ так же, как в алгоритме Диксона.

Замечание 3.18. Условие $\left(\dfrac{n}{p}\right) = +1$ для простых чисел p из факторной базы следует из сравнения $(x + [\sqrt{n}])^2 \equiv n \pmod p$, которое должно будет выполняться для некоторого $x \in \mathbb{Z}$.

Просеивание. Значения $x_i \in \mathbb{Z}$, для которых $Q(x_i)$ являются гладкими, определяются следующим образом. Для каждого простого числа p из факторной базы находим решения $r_1^{(p)}$ и $r_2^{(p)}$ уравнения $Q(x) \equiv 0 \pmod p$ (например, вероятностным алгоритмом, см. гл. 6 настоящей книги). Затем мы меняем $x \in \mathbb{Z}$ в достаточно большом промежутке $[-M;\ M]$, $M \in \mathbb{N}$, заводим массив, пронумерованный этими значениями x, и в элемент массива с номером x помещаем достаточно грубо вычисленные значения $\log|Q(x)|$. Затем для каждого простого p из факторной базы S мы выполняем процедуру просеивания: из элементов массива, номера которых лежат в арифметических прогрессиях $x \equiv r_1^{(p)} \pmod p$ и $x \equiv r_2^{(p)} \pmod p$, мы вычитаем достаточно грубо вычисленное значение $\log p$. Смысл такого вычитания состоит в том, что для элементов x в этих прогрессиях значение $Q(x)$ будет делиться на p, но деление $Q(x)$ на p мы пока что заменяем вычитанием $\log|Q(x)| - \log p$. После окончания процедуры просеивания в элементе

массива с номером x будет содержаться значение

$$\log|Q(x)| - \sum_{p\in S,\ p|Q(x)} \log p.$$

На самом деле необходимо проводить просеивание также и по степеням простых чисел p^l для нескольких небольших l. То есть надо находить решения $r_1^{(p,l)}$, $r_2^{(p,l)}$ уравнения $Q(x)\equiv 0 \pmod{p^l}$, и затем в ходе просеивания из элементов массива с номерами $x\equiv r_1^{(p,l)} \pmod{p^l}$ и $x\equiv r_2^{(p,l)} \pmod{p^l}$ проводить вычитание $\log p$. Тогда после просеивания в элементе массива с номером x будет стоять значение

$$\log|Q(x)| - \sum_{p\in S,\ p^l|Q(x)} l\log p.$$

После завершения процедуры просеивания мы идем по массиву и берем те номера x, для которых значения элементов массива невелики по абсолютной величине. Для этих номеров x значение $Q(x)$ скорее всего разложится в нашей факторной базе и мы факторизуем теперь число $Q(x)$ пробными делениями и оставляем те x, для которых $A_i = Q(x_i)$ полностью разложится в нашей факторной базе.

Замечание 3.19. Смысл процедуры просеивания заключается в экономии большого количества операций деления больших целых чисел. То есть, вместо того, чтобы сразу для каждого $x\in[-M;\,M]$ пытаться разложить $Q(x)$ в факторной базе, мы предварительно с помощью простых операций сложения и вычитания существенно сокращаем множество тех x, для которых мы затем будем факторизовать числа $Q(x)$ пробными делениями. Эта экономия дает на практике очень существенный эффект, из-за которого метод квадратичного решета превзошел все предыдущие алгоритмы факторизации.

Замечание 3.20. Поскольку каждое второе целое число делится на 2, каждое третье на 3 и т. д., то можно не проводить просеивание по степеням маленьких простых, т. е. например, по $p^k \leqslant 100$. Вместо этого, после окончания процедуры просеивания надо брать не просто маленькие элементы итогового массива, а элементы, не превосходящие по абсолютной величине некоторой небольшой границы, которая учитывает невычтенные нами из $\log|Q(x)|$ несколько значений $\log 2$, $\log 3$, $\log 5$ и т. д.

Замечание 3.21. В алгоритме квадратичного решета следует обязательно использовать стратегию LP (в книге [89] рекомендуется использовать LP с двумя большими простыми числами).

Силвермен [258] предложил в методе квадратичного решета использовать не один, а несколько многочленов $Q(x)$ вида

$$Q(x) = Ax^2 + 2Bx + C,$$

где $A \in \mathbb{N}$, B, $C \in \mathbb{Z}$, $B^2 - AC > 0$, причем $n \mid B^2 - AC$. Мы опишем это усовершенствование квадратичного решета, следуя [89]. Для многочлена $AQ(x)$ выполнено соотношение

$$AQ(x) = (Ax + B)^2 - (B^2 - AC) \equiv (Ax + B)^2 \pmod{n}.$$

Для того, чтобы значения $Q(x)$ были невелики, мы проводим просеивание по интервалу

$$I = \left[-\frac{B}{A} - M;\ -\frac{B}{A} + M\right]$$

с центром $-\dfrac{B}{A}$ в вершине параболы (здесь M — некоторый выбираемый нами параметр). Очевидно, что для $x \in I$ справедливы неравенства

$$Q\Big(-\frac{B}{A}\Big) \leqslant Q(x) \leqslant Q\Big(-\frac{B}{A} + M\Big).$$

Значения $Q(x)$ будут невелики, если

$$Q\Big(-\frac{B}{A}\Big) \approx Q\Big(-\frac{B}{A} + M\Big);$$

в этом случае $Q(x) \in \left[-Q\Big(-\dfrac{B}{A} + M\Big);\ Q\Big(-\dfrac{B}{A} + M\Big)\right]$. Отсюда находим

$$A\frac{B^2}{A^2} - 2\frac{B^2}{A} + C \approx -\Big(A\Big(-\frac{B}{A} + M\Big)^2 + 2B\Big(-\frac{B}{A} + M\Big) + C\Big) =$$
$$= -AM^2 + \frac{B^2}{A} - C.$$

Следовательно,

$$AM^2 \approx 2\frac{B^2}{A} - 2C,$$

откуда $A \approx \dfrac{\sqrt{2(B^2 - AC)}}{M}$. Кроме того, $n \mid B^2 - AC$, и если $B^2 - AC = n$, то

$$\max_{x \in I} |Q(x)| \approx \Big|Q\Big(-\frac{B}{A}\Big)\Big| = \frac{B^2 - AC}{A} \approx \frac{n}{\sqrt{2n}/M} \approx M\sqrt{\frac{n}{2}}.$$

Данная величина имеет порядок $\sqrt{n}$, если M много меньше, чем $\sqrt{n}$.

Поэтому для выбора многочленов $Q(x)$ мы сперва выбираем M (M много меньше чем $\sqrt{n}$, обычно M вида $L_2\left[\frac{1}{2}; \text{const}\right]$). Затем выбираем простое число A,

$$A \approx \frac{\sqrt{2n}}{M}, \quad \left(\frac{n}{A}\right) = +1.$$

Далее находим $B \in \mathbb{Z}$,

$$B^2 \equiv n \pmod{A}$$

(алгоритмы решения квадратных уравнений в конечных простых полях мы описываем далее в гл. 6). Затем полагаем $C = \dfrac{B^2 - n}{A}$, и

$$Q(x) = Ax^2 + 2Bx + C.$$

Замечание 3.22. 1. Технические детали можно найти в [258].

2. Коен [89] не рекомендует слишком часто менять многочлены.

3. Можно брать A составным, состоящим из произведения нескольких простых q, для которых $\left(\frac{n}{q}\right) = +1$. Тогда для B существует несколько значений — решений уравнения $z^2 \equiv n \pmod{A}$. Такой выбор A и B упростит процедуру инициализации перед просеиванием массива $\{\log|Q(x)|\}$ для данного набора многочленов.

В работе [229] предложен несколько иной выбор многочленов. А именно, предлагается выбирать

$$u(x) = a^2x + b, \quad v(x) = a, \quad w(x) = a^2x^2 + 2bx + c$$

и затем с помощью просеивания искать значения $x_i \in [-M; M]$, для которых

$$u(x_i)^2 \equiv v(x_i)^2 w(x_i) \pmod{n}, \quad u(x_i)^2 \neq v(x_i)^2 w(x_i),$$

и при этом значения $w(x_i)$ являются гладкими. Потом с помощью некоторой техники исключения мы найдем множество индексов I такое, что $\prod\limits_{i\in I} w(x_i)$ является квадратом. Тогда при

$$X = \prod_{i\in I} u(x_i), \quad Y = \prod_{i\in I} v(x_i) \sqrt{\prod_{i\in\ I} w(x_i)}$$

мы получим искомое соотношение

$$X^2 \equiv Y^2 \pmod{n}.$$

В работе [229], в отличие от [89], предлагается часто менять многочлены $u(x)$, $v(x)$ и $w(x)$. При этом инициализация перед просеиванием массива значений $\{\log|w(x)|\}$ делается некоторым эффективным «трубочным» способом. Дальнейшее развитие метода [229] предложено в работе [213].

В работе [66] описано эффективное применение стратегии LP в методе квадратичного решета.

В работе [74] описана возможность эффективного распараллеливания алгоритма квадратичного решета на несколько компьютеров.

Вывод. Метод квадратичного решета с использованием нескольких многочленов является эффективным и достаточно легко реализуемым на компьютере алгоритмом. Он, по-видимому, является наилучшим из известных алгоритмов факторизации произвольных чисел $n \in \mathbb{N}$, $n < 10^{110}$, если не считать метода факторизации с помощью эллиптических кривых (см. далее гл. 4), который в некоторых случаях может сработать быстрее. Однако для чисел n, больших 10^{110}, алгоритмы решета числового поля работают быстрее метода квадратичного решета. Об этих алгоритмах мы расскажем далее в § 3.6. Заметим, что сравнение эффективности квадратичного решета и алгоритмов решета числового поля было проведено в работах [110—112], см. также [74].

§ 3.5. Методы Шнорра—Ленстры и Ленстры—Померанса

В этом параграфе мы вкратце опишем два субэкспоненциальных вероятностных алгоритма для факторизации целых чисел. Один из них принадлежит Ленстре и Померансу [169], другой — Ленстре и Шнорру, см. [89; 241].

Алгоритм Шнорра—Ленстры был первым субэкспоненциальным алгоритмом, для которого требуется лишь небольшой объем памяти компьютера. Объем памяти здесь составляет величину $O(\log n)$ битов, а в методах, описанных в предыдущих параграфах, этот объем субэкспоненциален. Сложность метода составляет в среднем $L_n\left[\frac{1}{2}; 1\right]$ арифметических операций; алгоритм является вероятностным. Алгоритм работает с бинарными квадратичными формами отрицательного дискриминанта и с помощью некоторого случайного выбора ищет амбигову форму в группе классов квадратичных форм. Эта форма может дать разложение n на множители; в случае неудачи следует сделать следующий случайный выбор.

Отметим, что на практике алгоритм Шнорра—Ленстры никогда активно не использовался. Алгоритм Ленстры для факторизации с помощью эллиптических кривых (о котором см. 4) имеет ту же оценку сложности и так же требует немного памяти, но групповые операции на кривой выполняются значительно быстрее, чем операции в группе классов квадратичных форм. Поэтому алгоритм Ленстры считается значительно более эффективным и активно используется на практике.

Алгоритм Ленстры—Померанса также является вероятностным и раскладывает n на множители в среднем за $L_n\left[\frac{1}{2}; 1\right]$ арифметических операций. Этот алгоритм, так же как и алгоритм Шнорра—Ленстры, работает с группой классов бинарных квадратичных форм отрицательно дискриминанта и также ищет амбигову форму. Однако оценка сложности алгоритма Ленстры—Померанса является строгой и не использует недоказанные гипотезы и эвристические рассуждения. Более того, в работе [169] показано, что для некоторой бесконечной достаточно плотной последовательности натуральных чисел n эвристические рассуждения, лежащие в основе оценки сложности алгоритма Шнорра—Ленстры, являются неверными. Поэтому фактически у нас нет сейчас оснований считать, что алгоритм Шнорра—Ленстры имеет указанную выше оценку сложности.

Заметим, что алгоритм Ленстры—Померанса, как и алгоритм Шнорра—Ленстры, никогда активно не использовался на практике.

§ 3.6. Алгоритмы решета числового поля

Алгоритм решета числового поля для факторизации целых чисел специального вида (special number field sieve, или SNFS) был впервые предложен в 1990 г. в работе [161]. С его помощью было разложено на множители число Ферма $F_9 = 2^{512} + 1$, записываемое 155 десятичными знаками (см. об этом [162]). Эвристическая оценка сложности составляет $L_n[1/3; c]$ арифметических операций при $c = (32/9)^{1/3} = 1{,}5263\ldots$ Числа n, к которым применяется SNFS, имеют вид $n = r^e - s$, где $r \in \mathbb{N}$, $s \in \mathbb{Z}$, r и $|s|$ невелики.

Впоследствии метод был обобщен и применен для факторизации произвольных целых чисел. Он получил название general number field sieve или, сокращенно, GNFS. Оценка сложности также составляет $L_n[1/3; c]$ при некоторой постоянной c.

В настоящее время рекордным значением для натуральных чисел, разложенных с помощью SNFS, является число специального вида,

записываемое 227 десятичными знаками, см. [265]. Для RSA-чисел n, не имеющих специального вида, рекордные разложения были найдены в 1999 г.: сначала было разложено 140-значное RSA-число (см. [87]), а затем 155-значное RSA-число (512 битов), см. [88]. На факторизацию этого последнего числа потребовалось около 8400 mips-year[1].

Метод решета числового поля и его усовершенствования описаны в ряде работ, см. [107; 110; 111; 112; 159; 162; 163; 203; 204; 205; 226; 227; 290]. В настоящее время, согласно [74], это самый эффективный метод факторизации для чисел $n > 10^{110}$ (сравнение NFS с методом квадратичного решета см. в конце § 3.4); см. об этом также обзоры [211] и [209].

Фактически решето числового поля не является алгоритмом. Это метод вычисления, состоящий из нескольких этапов, и каждый из этих этапов обслуживается несколькими алгоритмами. Подробное описание SNFS и GNFS выходит далеко за рамки данной книги. Для того, чтобы читатель получил представление о методе решета числового поля и об используемым этим методом средствах, мы схематически опишем факторизацию числа Ферма F_9, следуя работе [162].

Далее в этом параграфе мы предполагаем, что читатель знаком с основами алгебраической теории чисел в объеме, например, книги [263].

Обозначим $N = F_9 = 2^{512} + 1$. Еще в 1903 г. был найден простой делитель $p_7 = 2424833$. Мы обозначим $n = N/p_7$. Это число было далее разложено с помощью SNFS на два сомножителя: $n = p_{49} \cdot p_{99}$. Для этого потребовалось около 700 рабочих станций и один суперкомпьютер для решения системы линейных уравнений; работа заняла несколько месяцев. Предполагалось предварительно (хотя это и не проверялось), что n не есть степень простого числа; это допущение в конечном счете оказалось верным.

Схема метода SNFS для n.

1 этап. Выбор факторной базы.

Факторная база состоит из некоторого множества элементов $a_p \in \mathbb{Z}/n\mathbb{Z}$, $a_p \neq 0$, где p пробегает некоторое конечное множество индексов $\mathbb{P}_0$. Все a_p обратимы в $\mathbb{Z}/n\mathbb{Z}$ (иначе был бы найден делитель n). Обозначим через $\mathbb{Z}^{\mathbb{P}_0}$ множество $|\mathbb{P}_0|$-мерных векторов:

$$\mathbb{Z}^{\mathbb{P}_0} = \{(\upsilon_p)_{p \in \mathbb{P}_0} \mid \upsilon_p \in \mathbb{Z}\}.$$

[1] Mips (англ.) — сокр. от million instructions per second — миллион команд в секунду (единица измерения быстродействия ЭВМ); year — год, mips-year — единица измерения числа операций за год.

Рассмотрим отображение

$$f\colon \mathbb{Z}^{\mathbb{P}_0} \to (\mathbb{Z}/n\mathbb{Z})^*, \quad f((\upsilon_p)_{p\in\mathbb{P}_0}) = \prod_{p\in\mathbb{P}_0} a_p^{\upsilon_p} \pmod n.$$

Это отображение *на*, если $\{a_p\}$ порождают $(\mathbb{Z}/n\mathbb{Z})^*$; обычно так и бывает, хотя доказать это, как правило, достаточно сложно.

2 этап. Нахождение соотношений.

Здесь мы ищем векторы $v \in \operatorname{Ker} f$, т. е. такие $v = (v_p)_{p\in\mathbb{P}_0}$, для которых

$$\prod_{p\in\mathbb{P}_0} a_p^{v_p} \equiv 1 \pmod n.$$

Нам нужно найти достаточно большое множество $V = \{v \in \operatorname{Ker} f\}$ этих векторов, точнее, $|V|$ должно быть немного больше $|\mathbb{P}_0|$.

3 этап. Нахождение зависимостей.

Здесь мы ищем нетривиальную линейную зависимость по модулю 2 найденных векторов $v \in V$; их количество больше, чем их размерность, поэтому такая зависимость существует. Для ее нахождения мы решаем систему линейных уравнений

$$\sum_j z_j v_j \equiv \overrightarrow{0} \pmod 2,$$

где $V = \{v_j\}$. Это большая разреженная система линейных уравнений над полем $\mathbb{Z}/2\mathbb{Z}$. Решив ее, мы тем самым найдем непустое подмножество $W \subseteq V$, для которого

$$\sum_{v\in W} v = \overrightarrow{0} \pmod 2.$$

Тогда $w = \dfrac{1}{2}\sum\limits_{v\in W} v$ — есть вектор с целыми коэффициентами, причем $2w \in \operatorname{Ker} f$. Это означает, что при $X \equiv f(W) \pmod n$ выполнено сравнение

$$X^2 \equiv f(2W) \equiv 1 \pmod n.$$

Тогда мы проверяем, выполняется ли неравенство

$$1 < \text{НОД}(X \pm 1,\ n) < n.$$

Если да, то делитель n найден, и мы останавливаемся. Иначе возвращаемся либо на 2 этап (находим новые соотношения), либо на 1 этап (строим новую факторную базу).

Конец схемы.

Для факторизации нашего числа $n = F_9/p_7$ рассматривается числовое поле $\mathbb{K} = \mathbb{Q}(\sqrt[5]{2})$. Элементы β поля $\mathbb{K}$ имеют вид $\beta = \sum_{i=0}^{4} q_i(\sqrt[5]{2})^i$, где $q_i \in \mathbb{Q}$; им соответствуют векторы $(q_0, q_1, q_2, q_3, q_4) \in \mathbb{Q}^5$. Сложение таких элементов (векторов) проводится покоординатно, а умножение выполняется с помощью соотношения $(\sqrt[5]{2})^5 = 2$. Мы обозначаем через $\mathrm{Norm}\,\beta$ норму алгебраического числа $\beta \in \mathbb{K}$ в поле $\mathbb{K}$; символом $\beta^{(j)}$ мы обозначаем элемент $\sigma_j(\beta)$, где σ_1, σ_2, σ_3, σ_4, σ_5 — все изоморфизмы $\mathbb{K}$ в $\mathbb{C}$.

Лемма 3.23. *Пусть* $1 \leqslant l \leqslant 4$, $a, b \in \mathbb{R}$. *Тогда* $\mathrm{Norm}(a - (\sqrt[5]{2})^l b) = a^5 - 2^l b^5$.

Доказательство. Поскольку $\sigma_j(\sqrt[5]{2})^l$ — корни многочлена $x^5 - 2^l$, выполняются равенства

$$\mathrm{Norm}(a - (\sqrt[5]{2})^l b) = b^5 \,\mathrm{Norm}\Big(\frac{a}{b} - (\sqrt[5]{2})^l\Big) = \\ = b^5 \prod_{j=1}^{5}\Big(\frac{a}{b} - \sigma_j(\sqrt[5]{2})^l\Big) = b^5\Big(\Big(\frac{a}{b}\Big)^5 - 2^l\Big),$$

Лемма доказана. □

Обозначим через $\mathbb{Z}_{\mathbb{K}}$ кольцо целых алгебраических чисел поля $\mathbb{K}$; можно доказать, что $\mathbb{Z}_{\mathbb{K}} = \mathbb{Z}[\sqrt[5]{2}]$. Зафиксируем границу гладкости $B \in \mathbb{R}_{>0}$ (это достаточно большое число).

Определение 3.24. Число $\beta \in \mathbb{Z}_{\mathbb{K}}$ называется B-гладким, если $\mathrm{Norm}\,\beta \in \mathbb{Z}$ является B-гладким числом (т. е. $\mathrm{Norm}\,\beta = \pm \prod_{\substack{p \text{ — простое,} \\ p \leqslant B}} p^{r_p}$, где $r_p \in \mathbb{Z}_{\geqslant 0}$).

Пусть R — коммутативное кольцо с единицей, ψ — кольцевой гомоморфизм:

$$\psi\colon \mathbb{Z}[\sqrt[5]{2}] \to R, \quad \psi(1) = 1.$$

Тогда для $c = \psi(\sqrt[5]{2}) \in \mathbb{N}$ выполнено равенство $c^5 = \psi(2) = 2 \in R$.

Обратно, если найдется элемент $c \in R$ такой, что $c^5 = 2 \in R$, то мы можем определить кольцевой гомоморфизм $\psi\colon \mathbb{Z}[\sqrt[5]{2}] \to R$ соотношениями $\psi(1) = 1$, $\psi(\sqrt[5]{2}) = c$.

Пример 3.25. Для $n = F_9/p_7$ мы положим $R = \mathbb{Z}/n\mathbb{Z}$, $c \equiv 2^{205} \pmod n$. Так как $2^{512} \equiv -1 \pmod n$, то $c^5 \equiv 2^{1025} \equiv 2 \pmod n$. Поэтому существует гомоморфизм $\varphi\colon \mathbb{Z}[\sqrt[5]{2}] \to \mathbb{Z}/n\mathbb{Z}$ такой, что $\varphi(1) = 1$, $\varphi(\sqrt[5]{2}) = 2^{205} \pmod n$.

Лемма 3.26. *Пусть* p — *простое число*, $p \not\equiv 1 \pmod 5$. *Тогда в* $\mathbb{Z}/p\mathbb{Z}$ *существует единственное* c *такое, что* $c^5 \equiv 2 \pmod p$.

Доказательство. Существует единственное $k \in \mathbb{N}$ такое, что $5k \equiv 1 \pmod{p-1}$, $1 \leqslant k \leqslant p-1$. Тогда, если $f(x)$ и $g(x)$ — отображения $\mathbb{Z}/p\mathbb{Z}$ в себя, $f(x) = x^5$, $g(x) = x^k$, то $f(g(x))$ — тождественное отображение. Следовательно, $f(x)$ есть отображение *на*, откуда следует наше утверждение. □

Лемма 3.27. *Если* $p \equiv 1 \pmod 5$, *то в* $\mathbb{Z}/p\mathbb{Z}$ *либо нет ни одного такого* c, *что* $c^5 \equiv 2 \pmod p$, *либо есть* 5 *таких элементов* $c \in \mathbb{Z}/p\mathbb{Z}$, *что* $c^5 \equiv 2 \pmod p$. *Если обозначить их через* $c_0, \ldots, c_4$, *то* $c_j = c_0 \cdot a^{\frac{p-1}{5} j}$, *где* a — *первообразный корень по модулю* p.

Теперь рассмотрим ненулевые идеалы $\mathfrak{B}$ в дедекиндовом кольце $\mathbb{Z}_m = \mathbb{Z}[\sqrt[5]{2}]$. Норма идеала $\mathfrak{B}$ определяется равенством $\operatorname{Norm}\mathfrak{B} = |\mathbb{Z}_{\mathbb{K}}/\mathfrak{B}|$.

Норма мультипликативна, т. е. $\operatorname{Norm}\mathfrak{AB} = \operatorname{Norm}\mathfrak{A}\operatorname{Norm}\mathfrak{B}$. Можно доказать, что поле $\mathbb{K}$ одноклассное, т. е. группа классов идеалов состоит из одного элемента; другими словами, любой идеал в $\mathbb{Z}_{\mathbb{K}}$ является главным.

Пусть $\mathfrak{P} \neq (0)$ — простой идеал $\mathbb{Z}_{\mathbb{K}}$. Тогда $\mathbb{Z}_{\mathbb{K}}/\mathfrak{P} = GF(p^k)$ для некоторого простого числа p и натурального k.

Определение 3.28. Простой идеал $\mathfrak{P}$ называется *простым идеалом первой степени*, если $\mathbb{Z}_{\mathbb{K}}/\mathfrak{P} = GF(p)$, где p — простое число.

По теореме Куммера простые идеалы первой степени в $\mathbb{Z}_{\mathbb{K}}$ имеют вид $\mathfrak{P} = (p, \sqrt[5]{2} - c)$, где p — простое число, $c \in \mathbb{Z}$, $f(x) = x^5 - 2$, $f(c) \equiv 0 \pmod p$. При этом для естественного гомоморфизма $\mathbb{Z}_{\mathbb{K}} \to \mathbb{Z}_{\mathbb{K}}/\mathfrak{P} = GF(p)$ элемент $\sqrt[5]{2}$ переходит в $\sqrt[5]{2} \equiv c \pmod p$, и $c^5 \equiv 2 \pmod p$.

Лемма 3.29. *Если* $p \not\equiv 1 \pmod 5$, *то в кольце* $\mathbb{Z}_{\mathbb{K}}$ *существует единственный простой идеал* $\mathfrak{P}$ *нормы* p. *При этом для элемента* $\pi \in \mathbb{Z}_{\mathbb{K}}$ *равенство* $\mathfrak{P} = (\pi)$ *справедливо тогда и только тогда, когда* $|\operatorname{Norm}\pi| = p$.

Доказательство. По теореме Куммера простые идеалы $\mathbb{Z}_{\mathbb{K}}$, лежащие над p, имеют вид $\mathfrak{P}_i = (p, f_i(\sqrt[5]{2}))$, где $f_i(x) \in \mathbb{Z}[x]$ — унитарные многочлены, $f_i(x) \pmod p$ неприводимы в $\mathbb{Z}/p\mathbb{Z}[x]$, и $f(x) = \prod\limits_i f_i^{e_i}(x) \pmod p$. Кроме того, $\operatorname{Norm}\mathfrak{P}_i = p^{\deg f_i(x)}$. Следовательно, $\mathfrak{P}_i$ — простой идеал первой степени тогда и только тогда, когда $f_i(x)$ — линейный многочлен. Но $f(x) \pmod p$ по лемме 3.26 имеет единственный линейный множитель; из этого следует единственность $\mathfrak{P}$.

Далее, если $\mathfrak{P} = (\pi)$, то $\operatorname{Norm}\mathfrak{P} = |\operatorname{Norm}\pi| = p$. Обратно, если $|\operatorname{Norm}\pi| = \operatorname{Norm}(\pi) = p$, то из разложения идеала (π) в произведение простых идеалов, мультипликативности нормы и единственности $\mathfrak{P}$ следует, что $(\pi) = \mathfrak{P}$. □

Пример 3.30. Пусть $p=2$. Тогда $f(x)=x^5-2\equiv x^5 \pmod 2$, $(2)=(2,\sqrt[5]{2})^5=\mathfrak{P}^5$, где $\mathfrak{P}=(\sqrt[5]{2})$ — простой идеал первой степени. При этом $\mathbb{Z}/\mathfrak{P}=\mathbb{Z}/2\mathbb{Z}$.

Пример 3.31. Пусть $p=3$, $f(x)=x^5-2\equiv x^5+1 \pmod 3=(x+1)\times(x^4-x^3+x^2-x+1) \pmod 3$. Тогда $\mathfrak{P}=(3,\sqrt[5]{2}+1)$ — единственный простой идеал первой степени нормы 3. Далее, для элемента $\pi=1+\sqrt[5]{2}$ с помощью леммы 3.23 находим, что $\operatorname{Norm}\pi=3$. Поэтому $\mathfrak{P}=(1+\sqrt[5]{2})$.

Определение 3.32. Пусть p — простое число, $c\in\mathbb{Z}/p\mathbb{Z}$, $c^5\equiv 2 \pmod p$. Через $\Psi_{p,c}$ мы будем обозначать кольцевой гомоморфизм $\Psi_{p,c}\colon \mathbb{Z}_{\mathbb{K}}\to\mathbb{Z}/p\mathbb{Z}$, определяемый равенством $\Psi_{p,c}(\sqrt[5]{2})=c$.

Лемма 3.33. *Пусть $\mathfrak{P}$ — простой идеал первой степени в $\mathbb{Z}[\sqrt[5]{2}]$, $\mathfrak{P}=(p,\sqrt[5]{2}-c)$, где p — простое число, $c^5\equiv 2 \pmod p$. Пусть $\pi\in\mathbb{Z}[\sqrt[5]{2}]$. Равенство $\mathfrak{P}=(\pi)$ имеет место тогда и только тогда, когда $|\operatorname{Norm}\pi|=p$ и $\Psi_{p,c}(\pi)=0$.*

Доказательство. Очевидно, что

$$\mathbb{Z}_{\mathbb{K}}/\operatorname{Ker}\Psi_{p,c}\simeq\mathbb{Z}/p\mathbb{Z}.$$

Поэтому $\operatorname{Ker}\Psi_{p,c}$ — простой идеал в $\mathbb{Z}_{\mathbb{K}}$, содержащий p и $\sqrt[5]{2}-c$; следовательно, $\operatorname{Ker}\Psi_{p,c}=\mathfrak{P}$.

Если $|\operatorname{Norm}\pi|=p$ и $\Psi_{p,c}(\pi)=0$, то $\pi\in\mathfrak{P}$ и нормы (π) и $\mathfrak{P}$ совпадают, откуда $(\pi)=\mathfrak{P}$.

Если же $\mathfrak{P}=(\pi)$, то $\pi\in\mathfrak{P}$, откуда $\Psi_{p,c}(\pi)=0$. При этом равенство $|\operatorname{Norm}\pi|=p$ очевидно. $\square$

Пример 3.34. Пусть $\pi=1+(\sqrt[5]{2})^2-2(\sqrt[5]{2})^3$, $\operatorname{Norm}\pi=-151$, $p=151$ — простое число. Если $c=116$, то $c^5\equiv 2 \pmod{151}$. Нетрудно убедиться в том, что $\Psi_{151,116}(\pi)\equiv 0 \pmod{151}$. Поэтому $\mathfrak{P}=(151,\sqrt[5]{2}-116)=(\pi)$ — простой идеал первой степени в $\mathbb{Z}_{\mathbb{K}}$.

Следствие 3.35. *С помощью леммы 3.33 можно находить образующие простых идеалов первой степени в $\mathbb{Z}_{\mathbb{K}}$.*

Определение 3.36. Число $\varepsilon\in\mathbb{Z}_{\mathbb{K}}$ называется *единицей* поля $\mathbb{K}$, если $\varepsilon^{-1}\in\mathbb{Z}_{\mathbb{K}}$.

По теореме Дирихле о единицах любая единица ε поля $\mathbb{K}$ имеет вид

$$\varepsilon=\pm\varepsilon_1^{l_1}\varepsilon_2^{l_2},\qquad l_1,\,l_2\in\mathbb{Z},$$

где ε_1 и ε_2 — основные единицы поля $\mathbb{K}$.

Поскольку поле $\mathbb{K}$ одноклассное, то любой ненулевой простой идеал $\mathfrak{P}$ имеет вид $\mathfrak{P}=(\pi_{\mathfrak{P}})$ для некоторого $\pi_{\mathfrak{P}}\in\mathbb{Z}_{\mathbb{K}}$. Тогда для любого

элемента $\beta \in \mathbb{Z}_{\mathbb{K}} \setminus 0$ из разложения

$$(\beta) = \prod_{\mathfrak{P}} \mathfrak{P}^{m_{\mathfrak{P}}(\beta)}$$

следует равенство

$$\beta = \varepsilon \prod_{\mathfrak{P}} \pi_{\mathfrak{P}}^{m_{\mathfrak{P}}(\beta)},$$

где ε — единица поля $\mathbb{K}$. При этом также $|\operatorname{Norm}\beta| = \prod\limits_{\mathfrak{P}} |\operatorname{Norm}\pi_{\mathfrak{P}}|^{m_{\mathfrak{P}}(\beta)}$.

Пример 3.37. Пусть $\beta = -1 + (\sqrt[5]{2})^4$, $\operatorname{Norm}\beta = 15$. Тогда $(\beta) = \mathfrak{P}_1\mathfrak{P}_2$, где $\mathfrak{P}_1$, $\mathfrak{P}_2$ — простые идеалы нормы 3 и 5 соответственно. Такие идеалы единственны, причем $\mathfrak{P}_1 = (1 + \sqrt[5]{2})$ (пример 3.31), а $\mathfrak{P}_2 = (1 + (\sqrt[5]{2})^2)$. Поэтому $-1 + (\sqrt[5]{2})^4 = \varepsilon_1(1 + \sqrt[5]{2})(1 + (\sqrt[5]{2})^2)$, откуда мы находим, что $\varepsilon_1 = -1 + \sqrt[5]{2}$. Элемент ε_1 является единицей поля $\mathbb{K}$, причем $\operatorname{Norm}\varepsilon_1 = 1$.

Пример 3.38. Пусть $\beta = 1 + (\sqrt[5]{2})^3$. Тогда, аналогично примеру 3.37, находим $1 + (\sqrt[5]{2})^3 = \varepsilon_2(1 + \sqrt[5]{2})^2$, и $\varepsilon_2 = -1 + (\sqrt[5]{2})^2 - (\sqrt[5]{2})^3 + (\sqrt[5]{2})^4$ — еще одна единица поля $\mathbb{K}$, $\operatorname{Norm}\varepsilon_2 = 1$.

Далее в алгоритме решета числового поля для факторизации $n = F_9/p_7$ авторы [162] считали, что найденные в примерах 3.37 и 3.38 единицы ε_1 и ε_2 являются основными единицами поля $\mathbb{K}$. При этом никаких противоречий в ходе работы алгоритма не возникало, и он успешно завершил работу. Заметим, что существуют алгоритмы построения основных единиц числового поля (см. [217]), однако в данном случае они не применялись.

Пусть ε — произвольная единица поля $\mathbb{K}$. Тогда $\varepsilon = \varepsilon_0^{v_0}\varepsilon_1^{v_1}\varepsilon_2^{v_2}$, где $\varepsilon_0 = -1$, а единицы ε_1, ε_2 были найдены нами в примерах 3.37 и 3.38; $v_j \in \mathbb{Z}$. Нам далее нужно уметь находить v_0, v_1, v_2, зная ε. Очевидно, что $v_0 = 0$, если $\operatorname{Norm}\varepsilon > 0$; $v_0 = 1$, если $\operatorname{Norm}\varepsilon < 0$. Положим

$$\lambda_1 = e^{(\log 2)/5}, \quad \lambda_2 = e^{\frac{2\pi i + \log 2}{5}}, \quad \lambda_1, \lambda_2 \in \mathbb{C},$$

и рассмотрим кольцевые гомоморфизмы

$$\Psi_j \colon \mathbb{Z}[\sqrt[5]{2}] \to \mathbb{C},$$

$\Psi_j(\sqrt[5]{2}) = \lambda_j$, $j = 1, 2$. Тогда при $j = 1, 2$ справедливы равенства $\log|\Psi_j(\varepsilon)| = v_1 \log|\Psi_j(\varepsilon_1)| + v_2 \log|\Psi_j(\varepsilon_2)|$, и можно показать, что определитель матрицы $\left\|\log|\Psi_j(\varepsilon_k)|\right\|_{j,k=1,2}$ отличен от нуля. Поскольку v_1, v_2 — целые числа, то с помощью приближенных вычислений мы сможем их найти, зная ε.

Теперь осуществим первый этап алгоритма — построение факторной базы. Мы выбираем границу гладкости $B = 1294973$ для $n = F_9/p_7$. Рассматриваем все простые числа $p \leqslant B$. Для каждого p ищем $c \in \mathbb{Z}/p\mathbb{Z}$, $c^5 \equiv 2 \pmod p$, и составляем таблицу пар (p, c), соответствующих простым идеалам первой степени $\mathfrak{P} = (p, \sqrt[5]{2} - c)$. Для нахождения c можно применять вероятностный алгоритм решения уравнения $f(x) \equiv 0 \pmod p$ (см. гл. 6). Точнее, если $p \not\equiv 1 \pmod 5$, то $c \equiv 2^k \pmod p$, где $5k \equiv 1 \pmod{p-1}$. Если же $p \equiv 1 \pmod 5$, то сперва проверяем выполнение сравнения $2^{(p-1)/5} \equiv 1 \pmod p$. Если оно не выполняется, то уравнение $c^5 \equiv 2 \pmod p$ неразрешимо; в противном случае для решения $x^5 - 2 \equiv 0 \pmod p$ применяем вероятностный алгоритм.

После нахождения множества простых идеалов первой степени $\mathfrak{P} = (p, \sqrt[5]{2} - c)$ мы должны найти $\pi_{\mathfrak{P}} \in \mathbb{Z}_{\mathbb{K}}$, такие, что $\mathfrak{P} = (\pi_{\mathfrak{P}})$. Для этого можно использовать следующий перебор. Рассмотрим множество

$$T = \left\{ \sum_{i=0}^{4} r_i (\sqrt[5]{2})^i \;\middle|\; r_i \in \mathbb{Z},\ |r_i| \leqslant \mathrm{const} \right\}.$$

Теоретические оценки для значения const в определении T можно извлечь из результатов [8, гл. 2]. Для $\pi \in T$ вычисляем $\mathrm{Norm}\,\pi$. Если $\mathrm{Norm}\,\pi = p$ — простое число и $p \leqslant B$, то затем мы перебираем простые идеалы первой степени $\mathfrak{P}$, делящие данное p, т. е. $\mathfrak{P} = (p, \sqrt[5]{2} - c)$. Если такой идеал $\mathfrak{P}$ единственен, то $\pi = \pi_{\mathfrak{P}}$ — его образующий. Если же таких $\mathfrak{P}$ несколько, то π — образующий того из них, для которого $\Psi_{p,c}(\pi) = 0$ (по утверждению 3.33).

На деле авторы [162] использовали несколько более тонкий перебор: перебирали $\pi = \sum_{i=0}^{4} s_i \alpha^i$, где $\alpha = -(\sqrt[5]{2})^3$. Здесь s_i — взаимно простые целые числа, причем $s_i \geqslant 0$, если $s_{i+1} = \ldots = s_4 = 0$, и $\sum_{i=0}^{4} s_i^2 \cdot 2^{6i/5} \leqslant 15000$.

С помощью этого перебора было найдено 49 726 образующих простых идеалов первой степени. Всего затем было найдено 99 500 образующих простых идеалов первой степени для нашей таблицы пар $(p, c) \leftrightarrow \mathfrak{P}$. Оставшиеся образующие искали аналогичным перебором идеалов $\alpha\mathfrak{P}$ нормы $8\,\mathrm{Norm}\,\mathfrak{P}$ (здесь $\alpha = -(\sqrt[5]{2})^3$); т. е. находили элементы π' нормы $8p$ и, поделив π' на α, находили образующие $\pi_{\mathfrak{P}}$.

Далее можно считать (заменяя при необходимости $\pi_{\mathfrak{P}}$ на $-\pi_{\mathfrak{P}}$), что $\mathrm{Norm}\,\pi_{\mathfrak{P}} > 0$.

Теперь рассмотрим гомоморфизм φ:

$$\varphi : \mathbb{Z}[\sqrt[5]{2}] \to \mathbb{Z}/n\mathbb{Z},$$

$\varphi(1) \equiv 1 \pmod n$, $\varphi(\sqrt[5]{2}) \equiv 2^{205} \pmod n$, из примера 3.25. При $\alpha = -(\sqrt[5]{2})^3$

$$\varphi(\alpha) = -2^{615} \equiv 2^{103} \pmod n,$$

поскольку $2^{512} \equiv -1 \pmod n$. Величина 2^{103} имеет порядок $n^{1/5}$, и из этого в дальнейшем мы извлечем немалую пользу. Заметим, что при a, $b \in \mathbb{Z}$ верно сравнение

$$\varphi(a + b\alpha) \equiv a + 2^{103} b \pmod n.$$

Факторная база $\mathbb{P}_0$, которую мы строим сейчас, будет состоять из:

1) 99 700 простых чисел p, $p \leqslant B_1 = 1295377$;

2) $\varepsilon_0 = -1$, ε_1, ε_2 — из примеров 3.37 и 3.38;

3) $\pi_{\mathfrak{P}}$ — образующих для 99 500 простых идеалов первой степени $\mathfrak{P} \subseteq \mathbb{Z}_{\mathbb{K}}$, $\operatorname{Norm} \mathfrak{P} \leqslant B_2 = 1294973$.

Для любого индекса $\rho \in \mathbb{P}_0$ мы полагаем

$$a_\rho = \varphi(\rho) \in \mathbb{Z}/n\mathbb{Z},$$

и это есть завершение 1 этапа алгоритма.

Теперь опишем 2 этап алгоритма — нахождение соотношений. Тривиальные соотношения вида $\varepsilon_0^2 = 1$ и $2 = (\sqrt[5]{2})^5$ мы не используем. Далее, имеется 4944 простых числа p из нашей факторной базы, для которых $f(x) \equiv x^5 - 2 \pmod p$ разлагается на линейные множители. Для этих p справедливо равенство

$$p = \varepsilon \prod_{\mathfrak{P}|(p)} \pi_{\mathfrak{P}},$$

где $\pi_{\mathfrak{P}}$ — найденные нами образующие простых идеалов $\mathfrak{P}$, а ε — единицы поля $\mathbb{K}$. Для ε мы находим представление $\varepsilon = \varepsilon_1^{v_1} \cdot \varepsilon_2^{v_2}$ с помощью приближенных вычислений (как это было описано выше); $\varepsilon_0 = -1$ не входит в это представление, так как $p > 0$ и все $\pi_{\mathfrak{P}}$ положительны по построению. Отсюда получаем соотношения

$$\varphi(p) = \varphi(\varepsilon_1)^{v_1} \varphi(\varepsilon_2)^{v_2} \prod_{\mathfrak{P}|(p)} \varphi(\pi_{\mathfrak{P}}).$$

Эти соотношения составили приблизительно 2,5% от всех соотношений, построенных на 2 этапе.

Дальнейшее построение соотношений происходит с помощью методов решета. Мы перебираем пары $a,\ b \in \mathbb{Z},\ b > 0$, такие, что

1) НОД$(a,\ b) = 1$;

2) $|a + 2^{103}b| = |\varphi(a + b\alpha)|$ является B_1-гладким числом, за исключением, может быть, одного простого делителя p_1, $B_1 < p_1 < 10^8$;

3) $|a^5 - 8b^5|$ является B_2-гладким, за исключением, может быть, одного простого делителя p_2, $B_2 < p_2 < 10^8$.

Если p_1 и p_2 отсутствуют, то пара $a,\ b$ дает нам полное соотношение; в противном случае соотношение называется частичным. Заметим, что поскольку $\mathrm{Norm}(a + b\alpha) = a^5 - 8b^5$, то при $p_2 = 1$ число $a + b\alpha$ является B_2-гладким.

Лемма 3.39. *Пусть $a,\ b \in \mathbb{Z},\ (a,\ b) = 1$. Тогда любой простой идеал $\mathfrak{P}$ кольца $\mathbb{Z}_{\mathbb{K}}$, делящий $a + b\alpha$, является простым идеалом первой степени. При этом, если разложение* $\mathrm{Norm}(a + b\alpha)$ *на простые множители имеет вид*

$$\mathrm{Norm}(a + b\alpha) = \prod_{k=1}^{m} q_k^{e_k},$$

то идеал $(a + b\alpha)$ в кольце $\mathbb{Z}_{\mathbb{K}}$ раскладывается на простые идеалы следующим образом:

$$(a + b\alpha) = \prod_{k=1}^{m} \mathfrak{Q}_k^{e_k}.$$

Здесь $\mathfrak{Q}_k \mid (q_k)$ и при $k > 2$ определяется по формуле

$$\mathfrak{Q}_k = (q_k,\ \sqrt[5]{2} - 2^{-1}(ab^{-1})^2 \pmod{q_k}),$$

а при $q_k = 2$ идеал $\mathfrak{Q}_k = (\sqrt[5]{2})$.

Доказательство. Пусть $p > 2$ — простое число, $\mathfrak{P}$ — простой идеал, такой, что $\mathfrak{P} \mid (p)$ и $\mathfrak{P} \mid (a + b\alpha)$. Поскольку $(a,\ b) = 1$, то $p \nmid b$. Следовательно, $ab^{-1} \pmod{p} \in \mathfrak{P}$, откуда

$$(ab^{-1} \pmod{p})^2 - \alpha^2 = (ab^{-1} \pmod{p})^2 - 2\sqrt[5]{2} \in \mathfrak{P},$$
$$\sqrt[5]{2} - (2^{-1}(ab^{-1})^2) \pmod{p} \in \mathfrak{P}.$$

По утверждению 3.23, $\mathrm{Norm}(a + b\alpha) = a^5 - 8b^5$. Поскольку $a^5 - 8b^5 \equiv$ $\equiv 0 \pmod{p}$, то

$$(2^{-1}(ab^{-1})^2)^5 - 2 \equiv 0 \pmod{p}.$$

Положим $c \equiv 2^{-1}(ab^{-1})^2 \pmod p$. По теореме Куммера $\mathfrak{Q} = (p, \sqrt[5]{2} - c)$ является простым идеалом первой степени, причем $\mathfrak{Q} \in \mathfrak{P}$. Следовательно, $\mathfrak{Q} = \mathfrak{P}$ — простой идеал первой степени.

Мы показали также, что для каждого простого числа $p = q_k$, делящего $\mathrm{Norm}(a + b\alpha)$, простой идеал $\mathfrak{Q}_k$ в $\mathbb{Z}_{\mathbb{K}}$, делящий $a + b\alpha$ и q_k, определен однозначно по указанной в лемме 3.39 формуле. Отсюда следует равенство $(a + b\alpha) = \prod\limits_{k=1}^{m} \mathfrak{Q}_k^{e_k}$. $\square$

Следствие 3.40. *Лемма 3.39 сводит факторизацию идеала $(a + b\alpha)$ (где $(a, b) = 1$) на простые идеалы к целочисленной факторизации нормы $\mathrm{Norm}(a + b\alpha) = a^5 - 8b^5$ на простые числа следующим образом. Пусть*

$$(a + b\alpha) = \prod_{\mathfrak{P} \text{ — простые идеалы}} \mathfrak{P}^{k_{\mathfrak{P}}}.$$

Тогда по лемме 3.39 $\mathrm{Norm}\,\mathfrak{P}$ — простое число, и

$$\mathrm{Norm}(a + b\alpha) = \prod_{\mathfrak{P}} (\mathrm{Norm}\,\mathfrak{P})^{k_{\mathfrak{P}}}.$$

Пусть $\mathfrak{P}$ — простой идеал, делящий $a + b\alpha$, $\mathrm{Norm}\,\mathfrak{P} = p$ — простое число.

1) *Если $p \not\equiv 1 \pmod 5$, то для данного p идеал $\mathfrak{P}$ единственен по лемме 3.29; тогда $k_{\mathfrak{P}} = \nu_{\mathfrak{P}}((a + b\alpha)) = \nu_p(a^5 - 8b^5)$.*

2) *Если $p \equiv 1 \pmod 5$, то сначала надо выяснить, для какого c из имеющегося набора значений простой идеал $\mathfrak{P} = (p, \sqrt[5]{2} - c)$ входит в разложение идеала $(a + b\alpha)$. Воспользуемся отображением Ψ из доказательства леммы 3.39. Тогда*

$$c \equiv \Psi(\sqrt[5]{2}) \pmod{\mathfrak{P}},$$

и если $\mathfrak{P} \mid (a + b\alpha)$, то

$$\Psi(a + b\alpha) \equiv a\Psi(1) + b\Psi(\alpha) \equiv 0 \pmod p.$$

Отсюда

$$\Psi(\alpha) \equiv c^3 \pmod p \equiv \frac{a}{b} \pmod p.$$

Поэтому

$$c^6 \equiv 2c \equiv \left(\frac{a}{b}\right)^2 \pmod p,$$

откуда $c \equiv 2^{-1}\left(\frac{a}{b}\right)^2 \pmod p$ определяется по a, b и p однозначно. Тогда для этого c $\mathfrak{P} = (p, \sqrt[5]{2} - c)$ и $\nu_{\mathfrak{P}}(a + b\alpha) = \nu_p(a^5 - 8b^5)$.

Итак, мы можем находить разложение идеала $(a+b\alpha)$ в произведение простых идеалов кольца $\mathbb{Z}_{\mathbb{K}}$.

Вернемся теперь к нахождению соотношений на 2 этапе алгоритма. Пусть $(a,\,b)=1$ и пара $a,\,b$ дает полное соотношение, т. е. $|a+2^{103}b|$ является B_1-гладким числом и $\operatorname{Norm}(a+b\alpha)=a^5-8b^5$ является B_2-гладким числом. Тогда имеют место следующие соотношения.

1) $a+b\alpha=\varepsilon\prod\limits_{\mathfrak{P}}\pi_{\mathfrak{P}}^{u_{\mathfrak{P}}}$, где ε — единица, $\mathfrak{P}$ — простые идеалы, $\operatorname{Norm}\mathfrak{P}\leqslant$ $\leqslant B_2$. Описанным выше способом мы находим показатели $u_{\mathfrak{P}}$. Затем вычисляем

$$\varepsilon=(a+b\alpha)\Big/\prod_{\mathfrak{P}}\pi_{\mathfrak{P}}^{u_{\mathfrak{P}}}$$

и находим разложение для ε:

$$\varepsilon=\varepsilon_0^{v_0}\varepsilon_1^{v_1}\varepsilon_2^{v_2}.$$

Показатель v_0 определяется знаком ε, а v_1 и v_2 мы находим с помощью приближенных вычислений, как это было описано ранее.

2) $a+2^{103}b=\pm\prod\limits_{p\leqslant B_1}p^{w_p}$, и это разложение мы находим с помощью просеивания, о котором будет сказано чуть позже.

Теперь применим отображение φ из примера 3.25. Тогда

$$\pm\prod p^{w_p}=a+2^{103}b\equiv\varphi(a+b\alpha)\pmod n\equiv$$

$$\equiv\prod_{i=0}^{2}\varphi(\varepsilon_i)^{v_i}\prod_{\mathfrak{P}}\varphi(\pi_{\mathfrak{P}})^{u_{\mathfrak{P}}}\pmod n,$$

и это есть соотношение между элементами факторной базы, которое мы построили на 2 этапе алгоритма.

Мы ограничились рассмотрением пары $a,\,b$, дающей полное соотношение. Если соотношение частичное, т. е. появляются дополнительные простые числа $p_1,\,p_2$, то далее с помощью некоторой техники исключения из нескольких таких соотношений делают одно полное соотношение. Здесь используется теория графов; мы не рассматриваем эту технику в деталях.

Накопив достаточно много соотношений, мы переходим к 3 этапу алгоритма и находим $X\in\mathbb{Z}$, $X^2\equiv 1\pmod n$, с помощью решения системы линейных уравнений над полем $\mathbb{Z}/2\mathbb{Z}$. При факторизации $n=F_9/p_7$ использовалось структурированное гауссово исключение для решения систем линейных уравнений над $\mathbb{Z}/2\mathbb{Z}$ (см. об этом гл. 11).

Теперь объясним, как проводилось нахождение пар $a, b \in \mathbb{Z}$, $(a, b) = 1$, таких, что $a + 2^{103}b$ и $a^5 - 8b^5$ являются B_1- и B_2-гладкими числами соответственно. Для этого использовалось просеивание, аналогичное просеиванию в методе квадратичного решета из § 3.4.

Зафиксируем b, $0 < b < 2 \cdot 10^6$. Выберем некоторый отрезок $[-A; A]$ (где A зависит от b), в котором будут изменяться значения a. Заводим массив, пронумерованный элементами $a \in [-A; A]$, $a \in \mathbb{Z}$, $(a, b) = 1$. В элемент массива с номером a мы заносим достаточно грубо вычисленное значение $\log|a + 2^{103}b|$. Далее для каждого простого числа p, $p \leqslant B_1$, мы идем по арифметической прогрессии номеров a, для которых $a + 2^{103}b \equiv 0 \pmod{p}$, т. е. $a = a_0(p) + jp$, $j \in \mathbb{Z}$. Из элементов массива с номерами a вычитаем грубо вычисленное значение $\log p$. Аналогичное просеивание делаем для некоторых степеней p^k простого числа p: в прогрессии $a + 2^{103}b \equiv 0 \pmod{p^k}$ из элементов массива с номерами a также вычитаем $\log p$.

После того, как мы проведем просеивание по всем p, в нашем массиве образуются некоторые результирующие значения. Если элемент массива с номером a окажется маленьким, то, скорее всего, $a + 2^{103}b$ будет B_1-гладким числом. Тогда мы факторизуем $a + 2^{103}b$ пробными делениями на $p \leqslant B_1$. Если $a + 2^{103}b$ окажется B_1-гладким, за исключением, может быть, одного простого p_1, $B_1 < p_1 < 10^8$, то мы сохраняем пару a, b в некотором массиве M.

Замечание 3.41. На деле просеивание происходит более тонким способом, но мы не будем здесь это обсуждать. Читатель может обратиться к оригинальной работе [162] за дальнейшими разъяснениями.

Теперь для пар a, b из массива M мы проводим аналогичное просеивание для нахождения B_2-гладких чисел $a^5 - 8b^5 = \mathrm{Norm}(a + b\alpha)$. В итоге мы найдем некоторое множество пар a, b, дающих полное или частичное соотношение.

Для факторизации $n = F_9/p_7$ было найдено 44 106 полных соотношений и 2 903 999 частичных соотношения. Для этого использовалось около 700 рабочих станций в течение около 5 недель. Было затрачено примерно 340 mips-year. Затем решение системы линейных уравнений над полем $\mathbb{Z}/2\mathbb{Z}$ заняло около 6 недель на суперкомпьютере. В итоге n разложилось на два множителя, простота каждого из которых была проверена с помощью алгоритма Ленстры—Коена (см. гл. 1). На этом факторизация числа Ферма F_9 была завершена.

Замечание 3.42. Нам осталось лишь пояснить, каким образом было выбрано числовое поле $\mathbb{K} = \mathbb{Q}(\sqrt[5]{2})$ для факторизации числа $n = F_9/p_7$. Мы опишем выбор числового поля для факторизации

числа n вида $n = r^e - s$, следуя [159]. Выбираем небольшое $d \in \mathbb{N}$ (обычно $d = 3, 5, 7$) и затем полагаем k равным наименьшему натуральному числу, для которого $kd \geqslant e$. Пусть $t = s \cdot r^{kd-e}$; t — небольшое целое число. Положим

$$f(x) = x^d - t, \qquad m = r^k \approx n^{1/d}.$$

Тогда $f(m) = m^d - t = r^{kd} - sr^{kd-e} \equiv 0 \pmod{n}$. Если $f(x) \in \mathbb{Z}[x]$ является неприводимым многочленом, то мы полагаем $\mathbb{K} = \mathbb{Q}(\alpha)$, где $\alpha \in \mathbb{C}$, $f(\alpha) = 0$. Очевидно, $[\mathbb{K} : \mathbb{Q}] = d$.

Дальнейшая работа алгоритма проводится в кольце $\mathbb{Z}[\alpha] \subseteq \mathbb{Z}_{\mathbb{K}}$. В нашем распоряжении также имеется кольцевой гомоморфизм φ, $\varphi(1) \equiv 1 \pmod{n}$, $\varphi(\alpha) \equiv m \pmod{n}$. В случае, когда $\mathbb{Z}[\alpha] = \mathbb{Z}_{\mathbb{K}}$ и поле $\mathbb{K}$ — одноклассное (как это было для $n = F_9/p_7$), алгоритмы решета числового поля работают более эффективно.

На этом мы заканчиваем схематичное описание алгоритмов решета числового поля для факторизации целых чисел.

§ 3.7. Заключение

Подведем итоги. Если мы хотим факторизовать натуральное число n, то сначала перебором $p = 2, 3, 5, 7, \ldots$ до некоторой границы следует отделить маленькие простые делители нашего числа. Затем следует проверить, является ли наше число, которое мы хотим факторизовать, составным. Для этого лучше всего использовать вероятностный тест Миллера—Рабина из гл. 1. Если наше число — вероятно простое, то нужно попробовать доказать его простоту с помощью алгоритма Ленстры—Коена из гл. 1. Если наше число — составное, то можно попытаться получить его разложение на множители с помощью $(P-1)$-метода Полларда и ρ-метода Полларда из гл. 2, а также с помощью метода эллиптических кривых Ленстры (из гл. 4). После этого для факторизации следует применить метод квадратичного решета, если наше число n не превосходит 10^{110}. Для чисел большей величины следует использовать алгоритмы решета числового поля.

Применительно к криптографии с открытым ключом мы видим, что RSA-модули n, равные произведению двух простых чисел, не являются безопасными для шифрования при условии $n \approx 2^{512}$. Согласно [74] RSA-модули $n \approx 2^{1024}$ будут оставаться безопасными еще по крайней мере 15 лет с момента написания [74], если только не будут найдены принципиально новые алгоритмы факторизации или не будет создан эффективный квантовый компьютер.

Глава 4. Применение эллиптических кривых для проверки простоты и факторизации целых чисел

§ 4.1. Введение. Эллиптические кривые и их свойства

Уже более пятнадцати лет эллиптические кривые активно используются в различных криптосистемах, а также в теоретико-числовых алгоритмах проверки простоты и факторизации целых чисел. В данной главе мы опишем некоторые основные свойства эллиптических кривых и их алгоритмические приложения в теории чисел. Подробное изложение теории эллиптических кривых можно найти в книгах [30; 256; 257]. Применение эллиптических кривых в криптографии описано в книге [65], а также в работах [188; 181; 145; 146; 179].

Пусть $\mathbb{K}$ — поле, $\operatorname{char}\mathbb{K} \neq 2, 3$. Эллиптическая кривая над $\mathbb{K}$ задается уравнением

$$y^2 = x^3 + ax + b, \quad \text{где} \quad a, b \in \mathbb{K}, \quad 4a^3 + 27b^2 \neq 0.$$

Она обозначается символом E или $E_{a,b}$. Если $\mathbb{K}_1$ — поле, содержащее $\mathbb{K}$, то мы обозначаем множество точек кривой через

$$E_{a,b}(\mathbb{K}_1) = E(\mathbb{K}_1) = \{(x, y) \in \mathbb{K}_1^2 \colon y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}.$$

Здесь $\mathcal{O}$ — «ноль» кривой, или «бесконечно удаленная точка». Это формальная точка, не имеющая координат. Она становится обычной точкой, только если мы перейдем в проективное пространство. Положим $x = X/Z$, $y = Y/Z$ в уравнении кривой; тогда $Y^2Z = X^3 + aXZ^2 + bZ^3$. Мы получаем точки $(X : Y : Z)$ в проективном пространстве над полем $\mathbb{K}$ (или над $\mathbb{K}_1 \supset \mathbb{K}$), удовлетворяющие этому однородному уравнению. Точки, у которых $Z \neq 0$ (можно считать тогда, что $Z = 1$), соответствуют точкам $(X/Z, Y/Z)$ аффинной кривой $E_{a,b}(\mathbb{K})$ (или $E_{a,b}(\mathbb{K}_1)$). Если же $Z = 0$, то из уравнения находим, что и $X = 0$. Тогда $Y \neq 0$ и можно считать, что $Y = 1$. Эта точка $(0 : 1 : 0)$ в проективном пространстве соответствует бесконечно удаленной точке $\mathcal{O}$ аффинной кривой.

На эллиптической кривой над полем $\mathbb{K}$ можно определить операцию сложения $\oplus$, относительно которой множества $E_{a,b}(\mathbb{K}_1)$ для полей $\mathbb{K}_1 \supset \mathbb{K}$ становятся абелевыми группами. Правила, по которым осуществляется сложение, таковы.

1. $(x, y) \oplus \mathcal{O} = (x, y)$, $\mathcal{O} \oplus \mathcal{O} = \mathcal{O}$.

2. $(x, y) \oplus (x, -y) = \mathcal{O}$.

3. Пусть $P = (x_1, y_1)$, $Q = (x_2, y_2)$ и $x_1 \neq x_2$. Проведем «прямую» через P и Q:

$$y = y_1 + \lambda(x - x_1), \quad \lambda = \frac{y_2 - y_1}{x_2 - x_1}.$$

Найдем точки пересечения прямой и кривой. Это P, Q и R, где $R = (x_3, y_3)$, x_3 — третий корень уравнения

$$(y_1 + \lambda(x - x_1))^2 = x^3 + ax + b.$$

По теореме Виета

$$x_1 + x_2 + x_3 = \lambda^2, \quad x_3 = -x_1 - x_2 + \lambda^2.$$

Тогда $y_3 = y_1 + \lambda(x_3 - x_1)$. Мы полагаем по определению сумму $P \oplus Q$ точек P и Q равной

$$P \oplus Q = (x_3, -y_3).$$

4. Пусть $P = (x_0, y_1)$, $Q = (x_0, y_2)$, где $y_1 \neq -y_2$. Тогда из уравнения кривой следует, что $y_1 = y_2 \neq 0$. Обозначим $y_0 = y_1 = y_2$. Тогда удвоение точки $P = (x_0, y_0)$, т. е. точка $2P = P \oplus P$, определяется с помощью «касательной». Из уравнения кривой находим, что $2y\,dy = (3x^2 + a)\,dx$. Касательная к кривой в точке P имеет вид

$$y = y_0 + \lambda(x - x_0), \quad \lambda = \frac{3x_0^2 + a}{2y_0}.$$

Подставляя уравнение касательной в уравнение кривой, получим уравнение

$$(y_0 + \lambda(x - x_0))^2 = x^3 + ax + b;$$

x_0 — его корень кратности два. Поэтому по теореме Виета третий корень равен $x_3 = -2x_0 + \lambda^2$. Отсюда $y_3 = y_0 + \lambda(x_3 - x_0)$. По определению полагаем $2P = P \oplus P = (x_3, -y_3)$.

Можно показать, что относительно введенной операции сложения множество $E_{a,b}(\mathbb{K}_1)$ образует абелеву группу. Если поле $\mathbb{K}_1$ конечно, то это конечная абелева группа.

Теорема 4.1 (Теорема Хассе). *Пусть p — простое число, $p > 3$, $E_{a,b}$ — эллиптическая кривая над полем $\mathbb{Z}/p\mathbb{Z}$. Тогда*

$$||E_{a,b}(\mathbb{Z}/p\mathbb{Z})| - (p+1)| < 2\sqrt{p}.$$

Величину $|E_{a,b}(\mathbb{Z}/p\mathbb{Z})|$ можно найти по следующей очевидной формуле

$$|E_{a,b}(\mathbb{Z}/p\mathbb{Z})| = 1 + \sum_{x\in\mathbb{Z}/p\mathbb{Z}} \left(1 + \left(\frac{x^3+ax+b}{p}\right)\right) = \\ = p + 1 + \sum_{x\in\mathbb{Z}/p\mathbb{Z}} \left(\frac{x^3+ax+b}{p}\right)$$

(здесь $\left(\frac{z}{p}\right)$ — символ Лежандра).

Пусть $|E_{a,b}(\mathbb{Z}/p\mathbb{Z})| = p + 1 - t$. Тогда для всех $j \in \mathbb{N}$ величина $|E_{a,b}(GF(p^j))|$ находится по следующей формуле:

$$|E_{a,b}(GF(p^j))| = p^j + 1 - t_j,$$

где t_j удовлетворяют рекуррентному соотношению

$$t_{j+1} = t_1 t_j - p t_{j-1}, \quad j \geqslant 1, \quad t_1 = t, \quad t_0 = 2.$$

В § 4.3 мы опишем алгоритмы нахождения порядка группы точек эллиптической кривой над конечным простым полем.

Определение 4.2. *j-инвариантом* эллиптической кривой называется величина

$$j = j(E) = 1728 \cdot \frac{4a^3}{4a^3 + 27b^2}.$$

§ 4.2. Алгоритм Ленстры для факторизации целых чисел с помощью эллиптических кривых

Вероятностный алгоритм Ленстры [167; 166] для факторизации целых чисел с помощью эллиптических кривых имеет среднюю оценку сложности

$$e^{((2+o(1))\log p \log\log p)^{1/2}} \log^2 n$$

для количества выполняемых арифметических операций; здесь p — минимальный простой делитель n. Если мы заменим p на $n^{1/2}$, то получим субэкспоненциальную оценку сложности $L_n[1/2; 1]$. Метод был усовершенствован в работе [192], см. также [57; 72; 89, гл. 10; 106; 260].

С помощью этого метода в 1995 г. было разложено на множители число Ферма $F_{10} = 2^{1024} + 1$ (см. [73]). Разложение имеет вид

$$F_{10} = p_8 \cdot p_{10} \cdot p_{40} \cdot p_{252},$$

где p_j обозначает простое число, записываемое j десятичными знаками. Факторизация заняла приблизительно 240 mips-years.

Заметим, что поскольку приведенная выше оценка сложности алгоритма Ленстры зависит от величины минимального простого делителя факторизуемого числа, этот метод может быть эффективно использован для отделения небольших простых делителей. Преимуществом метода является также использование лишь небольшого объема памяти компьютера. Получение оценки сложности алгоритма Ленстры требует довольно глубоких знаний в области теории эллиптических кривых и модулярных форм. Однако сам алгоритм описывается достаточно просто и так же просто реализуется на компьютере. Описание алгоритма Ленстры, кроме указанных выше работ, можно найти в книге [144].

Для описания алгоритма Ленстры нам потребуются эллиптические кривые уже не над полем, а над кольцом $\mathbb{Z}/n\mathbb{Z}$, где n — нечетное не делящееся на 3 составное число, которое мы хотим разложить на множители. Рассмотрим тройки чисел $(x, y, z) \in (\mathbb{Z}/n\mathbb{Z})^3$, такие, что идеал, порожденный x, y и z, совпадает с $\mathbb{Z}/n\mathbb{Z}$. Это выполняется, например, при НОД$(x, n) = 1$. *Орбитой* такого элемента $(x, y, z) \in (\mathbb{Z}/n\mathbb{Z})^3$ называется множество

$$\{(ux, uy, uz) \mid u \in (\mathbb{Z}/n\mathbb{Z})^*\};$$

оно обозначается через $(x : y : z)$. Множество всех орбит мы обозначим $\mathbb{P}^2(\mathbb{Z}/n\mathbb{Z})$ — это аналог проективного пространства над полем.

Эллиптическая кривая $E = E_{a,b}$ над кольцом $\mathbb{Z}/n\mathbb{Z}$ задается уравнением

$$y^2 = x^3 + ax + b,$$

где $a, b \in \mathbb{Z}/n\mathbb{Z}$, $6(4a^3 + 27b^2) \in (\mathbb{Z}/n\mathbb{Z})^*$. Обозначим множество точек кривой через

$$E = E_{a,b}(\mathbb{Z}/n\mathbb{Z}) = \{(x : y : z) \in \mathbb{P}^2(\mathbb{Z}/n\mathbb{Z}) \mid y^2 z = x^3 + axz^2 + bz^3\}.$$

Это множество имеет естественный закон сложения, относительно которого оно является конечной абелевой группой. Однако мы будем использовать такие же групповые операции, как для конечного про-

стого поля (см. § 4.1). Именно, мы обозначим

$$\mathcal{O} = (0:1:0) \in \mathbb{P}^2(\mathbb{Z}/n\mathbb{Z}),$$
$$\mathbb{V}_n = \{(x:y:1) \mid x,\, y \in \mathbb{Z}/n\mathbb{Z}\} \cup \{\mathcal{O}\}.$$

Для $P \in \mathbb{V}_n$ и для любого простого числа p, делящего n, мы обозначаем через P_p точку из $\mathbb{P}^2(\mathbb{Z}/p\mathbb{Z})$, полученную приведением координат точки P по модулю p. Очевидно, что $P_p = \mathcal{O}_p$ тогда и только тогда, когда $P = \mathcal{O}$.

Сложение точек $P,\ Q \in \mathbb{V}_n$ мы будем проводить следующим образом (считаем, что a задано). При вычислении суммы $P \oplus Q$ мы либо найдем делитель d числа n (и тогда наша цель факторизации n будет достигнута), либо найдем точку $R \in \mathbb{V}_n$, для которой выполнено следующее условие:

если $p \mid n$, $\tilde{a} \equiv a \pmod{p}$, и если для p найдется $\tilde{b} \in \mathbb{Z}/p\mathbb{Z}$ такое, что $6(4\tilde{a}^3 + 27\tilde{b}^2) \neq 0$ в $\mathbb{Z}/p\mathbb{Z}$ и при этом $P_p,\ Q_p \in E_{\tilde{a},\tilde{b}}(\mathbb{Z}/p\mathbb{Z})$, то тогда $R_p = P_p \oplus Q_p$ в $E_{\tilde{a},\tilde{b}}(\mathbb{Z}/p\mathbb{Z})$; здесь сумма вычисляется по правилам для эллиптической кривой над полем, которые были описаны в § 4.1.

Замечание 4.3. Если у нас есть точка $P = (x:y:1)$, есть p и a, то $y^2 \equiv x^3 + ax + b \pmod{n}$. Отсюда $b \equiv y^2 - x^3 - ax \pmod{n}$ определяется по модулю n однозначно. Тогда однозначно определяется и значение $\tilde{b} \equiv b \pmod{p}$. Если для каждого $p \mid n$ редуцированная точка Q_p попадет на кривую $y^2 \equiv x^3 + ax + \tilde{b} \pmod{p}$ над полем $\mathbb{Z}/p\mathbb{Z}$, то мы можем складывать P_p и Q_p на кривой над $\mathbb{Z}/p\mathbb{Z}$ и вычислять сумму $P \oplus Q$ над $\mathbb{Z}/n\mathbb{Z}$. Если же для некоторого p точка Q_p не попадет на кривую $y^2 \equiv x^3 + ax + \tilde{b} \pmod{p}$, то складывать точки P и Q нельзя.

Сложение точек P и Q из множества $\mathbb{V}_n$ осуществляется следующим образом. Если $P = \mathcal{O}$, то $R = Q$; если $Q = \mathcal{O}$, то $R = P$. Пусть далее $P,\ Q \neq \mathcal{O}$, $P = (x_1 : y_1 : 1)$, $Q = (x_2 : y_2 : 1)$. Найдем $d = \text{НОД}(x_1 - x_2,\, n)$ с помощью алгоритма Евклида. Если $1 < d < n$, то мы нашли делитель n, и алгоритм останавливается. Если $d = 1$, то $x_1 \not\equiv x_2 \pmod{n}$, и $x_1 \not\equiv x_2 \pmod{p}$ для любого простого числа p, $p \mid n$. Тогда (с помощью обобщенного алгоритма Евклида) найдем $(x_1 - x_2)^{-1} \pmod{n}$. Далее положим

$$\lambda = (y_1 - y_2)(x_1 - x_2)^{-1} \pmod{n}, \quad \nu = y_1 - \lambda x_1 \pmod{n},$$
$$x_3 = -x_1 - x_2 + \lambda^2 \pmod{n}, \quad y_3 = -\lambda x_3 - \nu \pmod{n}.$$

Тогда по определению сумма P и Q равна $R = P \oplus Q = (x_3 : y_3 : 1)$.

Заметим, что в случае $d = 1$ сумму $P \oplus Q$ мы находим по формуле секущей из § 4.1.

Теперь рассмотрим случай $d = \text{НОД}(x_1 - x_2, n) = n$. Тогда $x_1 \equiv x_2 \pmod{n}$, и сложение надо проводить по формуле касательной. Найдем $d_1 = \text{НОД}(y_1 + y_2, n)$. Если $1 < d_1 < n$, то мы нашли делитель n, и алгоритм останавливается. Если $d_1 = n$, т. е. $y_1 \equiv -y_2 \pmod{n}$, то положим $R = P \oplus Q = \mathcal{O}$. Если $d_1 = 1$, то находим

$$\lambda_1 = (3x_1^2 + a)(y_1 + y_2)^{-1} \pmod{n}, \quad \nu = y_1 - \lambda x_1 \pmod{n},$$
$$x_3 = -2x_1 + \lambda^2 \pmod{n}, \quad y_3 = -\lambda x_3 - \nu \pmod{n}$$

и полагаем $R = P \oplus Q = (x_3 : y_3 : 1)$.

Итак, мы определили сложение точек $\mathbb{V}_n$. Теперь определим умножение точек $P \in \mathbb{V}_n$ на натуральные числа k. В результате этого умножения мы либо находим делитель d числа n, $1 < d < n$, либо получаем точку $R \in \mathbb{V}_n$, удовлетворяющую следующим условиям:

если p — простое число, $p \mid n$, $\bar{a} \equiv a \bmod p$, и если для p найдется $\tilde{b} \in \mathbb{Z}/p\mathbb{Z}$ такое, что $6(4\bar{a}^3 + 27\tilde{b}^2) \neq 0 \pmod{p}$, и при этом $P_p \in E_{\bar{a},\tilde{b}}(\mathbb{Z}/p\mathbb{Z})$, то тогда $R_p = kP_p$ в группе $E_{\bar{a},\tilde{b}}(\mathbb{Z}/p\mathbb{Z})$; здесь kP_p вычисляется по правилам сложения для эллиптической кривой над полем, описанным в § 4.1.

Замечание 4.4. Вычисление $R = kP \in \mathbb{V}_n$ проводится аналогично описанному выше вычислению $R = P \oplus Q \in \mathbb{V}_n$. При этом бинарный метод возведения в степень в алгоритме Ленстры не используется; это означает, что для бинарной записи $k = \sum\limits_i 2^{j_i}$ мы не вычисляем $kP = \bigoplus\limits_i 2^{j_i} P$, предварительно найдя множество точек $2^j P$. Вместо этого мы рассматриваем разложение $k = k_1 \dots k_t$ в произведение небольших натуральных чисел k_i, $k_1 > k_2 > \dots > k_t$. Именно такие k будут рассматриваться далее в алгоритме факторизации. Точку kP мы представляем в виде $k_1(k_2(\dots(k_t P)\dots))$ и вычисляем ее последовательными умножениями на k_i, $i = t, \dots, 1$; здесь уже допустим бинарный метод вычислений.

Алгоритм факторизации с одной эллиптической кривой.

На входе алгоритма задано факторизуемое число n и параметры $\upsilon, \varpi \in \mathbb{N}$, зависящие от n. Также даны $a, x, y \in \mathbb{Z}/n\mathbb{Z}$ такие, что $P = (x : y : 1) \in \mathbb{V}_n$ и для $b \equiv y^2 - x^3 - ax \pmod{n}$ выполнено условие $6(4a^3 + 27b^2) \in (\mathbb{Z}/n\mathbb{Z})^*$. Алгоритм находит натуральный делитель d числа n, $1 < d < n$.

Для каждого $r \in \mathbb{N}$, $2 \leqslant r \leqslant \varpi$, мы полагаем

$$e(r) = \max\{m \mid m \in \mathbb{Z}_{\geqslant 0},\ r^m \leqslant \upsilon + 2\sqrt{\upsilon} + 1\},$$

и затем

$$k = \prod_{\substack{2 \leqslant r \leqslant w \\ r \text{ — простое}}} r^{e(r)}.$$

Пусть $P = (x : y : 1) \in \mathbb{V}_n$. Тогда P лежит на эллиптической кривой $E_{a,b}$ над $\mathbb{Z}/n\mathbb{Z}$, определенной уравнением $Y^2 = X^3 + aX + b$. Мы вычисляем точку kP так, как было объяснено выше. Если в ходе вычисления найден делитель d числа n, $1 < d < n$, то мы разложили n на множители и алгоритм останавливается. Если мы нашли kP и делитель d при этом не найден, то алгоритм заканчивает работу и выдает сообщение о неудачной попытке факторизации.

Конец алгоритма.

Замечание 4.5. Как следует выбирать параметры v, w, a, x, y? Элементы $a, x, y \in \mathbb{Z}/n\mathbb{Z}$ выбираются случайно; тогда при $b \equiv y^2 - x^3 - ax \pmod n$ мы получим эллиптическую кривую $E_{a,b}$ над $\mathbb{Z}/n\mathbb{Z}$ и точку на ней. Параметр w теоретически равен $w = L(p)^{\frac{1}{\sqrt{2}}+o(1)}$, где $L(t) = \exp\sqrt{\log t \log\log t}$, p — минимальный простой делитель числа n. Поскольку p нам неизвестен, но $p \leqslant n^{1/2}$, то $L(p) \leqslant L(n^{1/2}) = \exp\left(\left(\frac{1}{\sqrt{2}} + o(1)\right)\sqrt{\log n \log\log n}\right)$, откуда получаем верхнюю оценку

$$w \leqslant L(n)^{\frac{1}{2}+o(1)}.$$

На практике следует испытать несколько последовательно увеличивающихся значений w. Параметр v, который оценивает степени небольших простых чисел r, входящих в значение k из алгоритма, на практике также следует выбирать эмпирически, используя некоторую возрастающую последовательность значений.

Замечание 4.6. На практике алгоритм факторизации при заданных n, v, w состоит в следующем. Случайно выбирают очередные $a, x, y \in \mathbb{Z}/n\mathbb{Z}$ и выполняют алгоритм факторизации с одной кривой. Так повторяется до тех пор, пока мы не разложим n на множители или пока не закончится отведенное для работы алгоритма время.

Замечание 4.7. Усовершенствование Монтгомери [192] заключается в одновременном вычислении обратных элементов для нескольких элементов $a_1 \pmod n, \ldots, a_l \pmod n$ в кольце $\mathbb{Z}/n\mathbb{Z}$ (см. также [89, гл. 10]). Это позволяет одновременно работать сразу с несколькими кривыми. Другое усовершенствование алгоритма Ленстры заключается в использовании только проективных координат; в этом случае

нам вообще не нужно будет проводить деление по модулю n. Однако Коен [89, гл. 10] все же рекомендует аффинные координаты в сочетании с усовершенствованием Монтгомери.

Замечание 4.8. Алгоритм Ленстры аналогичен $(P-1)$-методу Полларда, описанному в гл. 2. В нем, так же, как и в $(P-1)$-методе, возможна вторая стадия, см. [89; 192; 74].

§ 4.3. Вычисление порядка группы точек эллиптической кривой над конечным полем

Вычисление порядка группы точек на эллиптической кривой над конечным простым полем имеет важные приложения как в криптографии, так и в алгоритмах проверки простоты чисел, о которых будет рассказано в следующем параграфе.

Пусть p — простое число, $p>3$, эллиптическая кривая $E=E_{a,b}$ над полем $\mathbb{Z}/p\mathbb{Z}$ задана уравнением $y^2=x^3+ax+b$. Для нахождения $|E(\mathbb{Z}/p\mathbb{Z})|$ Р. Шуф в 1985 г. в работе [244] предложил алгоритм, имеющий полиномиальную сложность $O(\log^8 p)$ битовых операций (см. также [245]). Дальнейшие усовершенствования алгоритма Шуфа были предложены Аткином, Элкисом, Мюллером и другими авторами, см. [65; 113; 202; 171; 245]. Это позволило на практике вычислять порядки групп точек для простых полей, число элементов которых записывается несколькими сотнями десятичных цифр; рекордное значение p равно $10^{499}+153$.

В данном параграфе мы опишем первоначальный алгоритм Шуфа из работы [244].

Обозначим через $\overline{\mathbb{Z}/p\mathbb{Z}}$ алгебраическое замыкание поля $\mathbb{Z}/p\mathbb{Z}$. Отображение Фробениуса $\varphi\colon E(\overline{\mathbb{Z}/p\mathbb{Z}})\to E(\overline{\mathbb{Z}/p\mathbb{Z}})$ определяется соотношениями

$$\varphi(x,\,y)=(x^p,\,y^p),\qquad \varphi(\mathcal{O})=\mathcal{O}.$$

Нетрудно доказать, что φ является гомоморфизмом и вложением $E(\overline{\mathbb{Z}/p\mathbb{Z}})$ в себя; очевидно, что точки $E(\mathbb{Z}/p\mathbb{Z})$ остаются неподвижными при действии φ. Пусть

$$|E(\mathbb{Z}/p\mathbb{Z})|=p+1-t,$$

по теореме Хассе $|t|<2\sqrt{p}$. Целое число t называется *следом отображения Фробениуса*; отображение φ удовлетворяет уравнению

$$\varphi^2-t\varphi+p=0.$$

Для каждого натурального числа n обозначим через $E[n]$ подгруппу $E(\overline{\mathbb{Z}/p\mathbb{Z}})$, состоящую из точек, порядок которых делит n:

$$E[n] = \{P \in E(\overline{\mathbb{Z}/p\mathbb{Z}}) \mid nP = \mathcal{O}\}.$$

Теорема 4.9 (см. [256]). *Если $n > 1$ и p не делит n, то группа $E[n]$ изоморфна $\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/p\mathbb{Z}$.*

Пример 4.10. Рассмотрим x_1, x_2, x_3 — три различных корня уравнения $x^3 + ax + b = 0$ в $\overline{\mathbb{Z}/p\mathbb{Z}}$. Тогда

$$E[2] = \{(x_i, 0) : i = 1, 2, 3\} \cup \{\mathcal{O}\}.$$

Определим многочлены $\psi_n(x, y) \in \mathbb{Z}/p\mathbb{Z}[x, y]$, $n = -1, 0, 1, 2, \ldots$, следующими соотношениями:

$$\psi_{-1}(x, y) = -1, \quad \psi_0(x, y) = 0, \quad \psi_1(x, y) = 1, \quad \psi_2(x, y) = 2y,$$
$$\psi_3(x, y) = 3x^4 + 6ax^2 + 12bx - a^2,$$
$$\psi_4(x, y) = 4y(x^6 + 5ax^4 + 20bx^3 - 5a^2x^2 - 4abx - 8b^2 - a^3);$$

далее при $n \geqslant 3$

$$\psi_{2n}(x, y) = \psi_n(x, y)(\psi_{n+2}(x, y)\psi_{n-1}(x, y)^2 - \psi_{n-2}(x, y)\psi_{n+1}(x, y)^2)/(2y),$$

и при $n \geqslant 2$

$$\psi_{2n+1}(x, y) = \psi_{n+2}(x, y)\psi_n(x, y)^3 - \psi_{n+1}(x, y)^3\psi_{n-1}(x, y);$$

везде y^2 следует заменять на $x^3 + ax + b$.

Многочлены $\psi_n(x, y)$ называются *многочленами деления*. Нетрудно доказать по индукции, что $f_n(x)$, определенные равенством

$$f_n(x) = \begin{cases} \psi_n(x, y), & \text{если } n \text{ нечетное,} \\ \psi_n(x, y)/y, & \text{если } n \text{ четное,} \end{cases}$$

являются многочленами от x, т. е. $f_n(x) \in \mathbb{Z}/p\mathbb{Z}[x]$. Кроме того, если n — нечетно, $p \nmid n$, то $\deg f_n(x) = (n^2 - 1)/2$.

Теорема 4.11. *Пусть $P = (x, y) \in E(\overline{\mathbb{Z}/p\mathbb{Z}}) \setminus E[2]$. Пусть $n \geqslant 3$. Равенство $nP = \mathcal{O}$ выполнено тогда и только тогда, когда $f_n(x) = 0$.*

Теорема 4.12. *Пусть $P = (x, y) \in E(\overline{\mathbb{Z}/p\mathbb{Z}}) \setminus E[2]$, $n \geqslant 2$, причем $nP \neq \mathcal{O}$. Тогда*

$$nP = \left(x - \frac{\psi_{n-1}(x, y)\psi_{n+1}(x, y)}{\psi_n(x, y)^2}, \frac{\psi_{n+2}(x, y)\psi_{n-1}(x, y)^2 - \psi_{n-2}(x, y)\psi_{n+1}(x, y)^2}{4y\psi_n(x, y)^3}\right).$$

Далее в алгоритме Шуфа мы будем находить значение $t \pmod l$ для небольших простых чисел l. Если таких l будет достаточно много,

точнее, если $\Pi l > 4\sqrt{p}$, то, найдя затем по китайской теореме об остатках значение $t \pmod{\Pi l}$, мы получим, что искомое значение t равно абсолютно наименьшему вычету в классе $t \pmod{\Pi l}$. Это следует из теоремы Хассе (см. § 4.1). Тогда мы найдем искомое значение $|E(\mathbb{Z}/p\mathbb{Z})| = p + 1 - t$.

Рассмотрим сначала случай $l = 2$. Согласно приведенному выше примеру, в $E(\mathbb{Z}/p\mathbb{Z})$ найдется ненулевая точка $P = (x, 0)$ второго порядка, если и только если выполнено условие

$$\text{НОД}(x^p - x,\ x^3 + ax + b) \neq 1.$$

Это условие равносильно четности числа $|E(\mathbb{Z}/p\mathbb{Z})|$ что, в свою очередь, равносильно четности t, поскольку $p + 1$ четно. Итак, $t \equiv 0 \pmod 2$ тогда и только тогда, когда

$$\text{НОД}(x^p - x,\ x^3 + ax + b) \neq 1;$$

в противном случае $t \equiv 1 \pmod 2$.

Везде далее мы обозначаем через l нечетное фиксированное (небольшое) простое число, $l \neq p$. Реально в алгоритме следует рассматривать простые числа l порядка $O(\log p)$.

Рассмотрим группу $E[l]$; очевидно, что $\varphi(E[l]) \subset E[l]$. Легко показать, что φ является изоморфизмом $E[l]$. Обозначим $\varphi_l = \varphi|_{E[l]}$. Тогда φ_l удовлетворяет уравнению

$$\varphi_l^2 - t\varphi_l + p = 0.$$

Покажем, что если φ_l удовлетворяет уравнению

$$\varphi_l^2 - t'\varphi_l + p = 0$$

при некотором $t' \in \mathbb{Z}$, то $t \equiv t' \pmod l$. Действительно, вычитая второе уравнение из первого, получим

$$(t - t')\varphi_l = 0 \quad \text{на } E[l],$$

откуда $t \equiv t' \pmod l$, поскольку φ_l — изоморфизм.

Теперь нам надо найти значение τ, $0 \leqslant \tau \leqslant l - 1$, для которого

$$\varphi_l^2 - \tau\varphi_l + p = 0 \quad \text{на } E[l].$$

Тем самым мы определим значение $t \pmod l$. Другими словами, нам нужно найти $\tau \in \mathbb{Z}/l\mathbb{Z}$, для которого на $E[l]$ выполняется равенство $\varphi_l^2 + p = \tau\varphi_l$. Случай $\tau = 0$ рассматривается отдельно. Если же $\tau \neq 0$, то, для

$$k \equiv p \pmod l, \qquad 1 \leqslant k \leqslant l - 1,$$

мы, по теореме 4.12, примененной к точке $P=(x,\,y)\in E[l]\setminus\mathcal{O}$ можем переписать наше равенство $\varphi_l^2(P)+pP=\tau\varphi_l(P)$ в виде

$$(x^{p^2},\,y^{p^2})\oplus\Big(x-\frac{\psi_{k-1}(x,\,y)\psi_{k+1}(x,\,y)}{\psi_k(x,\,y)^2},\\ \frac{\psi_{k+2}(x,\,y)\psi_{k-1}(x,\,y)^2-\psi_{k-2}(x,\,y)\psi_{k+1}(x,\,y)^2}{4y\psi_k(x,\,y)^3}\Big)=\\ =\Big(x^p-\Big(\frac{\psi_{\tau-1}(x,\,y)\psi_{\tau+1}(x,\,y)}{\psi_\tau(x,\,y)^2}\Big)^p,\\ \Big(\frac{\psi_{\tau+2}(x,\,y)\psi_{\tau-1}(x,\,y)^2-\psi_{\tau-2}(x,\,y)\psi_{\tau+1}(x,\,y)^2}{4y\psi_\tau(x,\,y)^3}\Big)^p\Big).$$

Знак $\oplus$ обозначает операцию сложения точек на кривой. Последнее равенство приводится к виду

$$\begin{cases}H_1(x)\equiv 0\pmod{f_l(x)},\\ H_2(x)\equiv 0\pmod{f_l(x)},\end{cases}$$

где $H_1(x)$, $H_2(x)\in\mathbb{Z}/p\mathbb{Z}[x]$. В результате перебора $\tau=0,\,1,\,\ldots,\,l-1$ мы найдем истинное значение $t\equiv\tau\pmod l$. Так выглядит основная идея алгоритма, которую мы далее конкретизируем.

Алгоритм Шуфа.

Алгоритм находит $t\pmod l$ для таких простых чисел l, что $\prod l>4\sqrt{p}$. Затем с помощью китайской теоремы об остатках находится истинное значение t. Значение $t\pmod 2$ находим так, как было сказано выше. Пусть далее $l>2$, l — фиксированное простое число, $l\neq p$.

1 этап алгоритма для фиксированного l. Мы проверяем, существует ли точка $P=(x,\,y)\in E[l]\setminus\mathcal{O}$ такая, что

$$\varphi_l^2(P)=\pm kP,$$

где $k\equiv p\pmod l$, $1\leqslant k\leqslant l-1$. Сначала проверяем по первой координате; это означает, что должно выполняться равенство

$$x^{p^2}=x-\frac{\psi_{k-1}(x,\,y)\psi_{k+1}(x,\,y)}{\psi_k(x,\,y)^2}.$$

При четном k это равенство имеет вид

$$x^{p^2}=x-\frac{f_{k-1}(x)f_{k+1}(x)}{f_k(x)^2(x^3+ax+b)},$$

а при нечетном k —

$$x^{p^2} = x - \frac{f_{k-1}(x) f_{k+1}(x)(x^3 + ax + b)}{f_k(x)^2}.$$

Следовательно, по теореме 4.11 точка $P = (x, y) \in E[l] \setminus \mathcal{O}$, удовлетворяющая соотношению $\varphi_l^2(P) = \pm kP$, существует тогда и только тогда, когда

$$\text{НОД}((x^{p^2} - x) f_k(x)^2 (x^3 + ax + b) + f_{k-1}(x) f_{k+1}(x),\ f_l(x)) \neq 1$$

при четном k, и

$$\text{НОД}((x^{p^2} - x) f_k(x)^2 + f_{k-1}(x) f_{k+1}(x)(x^3 + ax + b),\ f_l(x)) \neq 1$$

при нечетном k. Если же соответствующий НОД (при k четном или нечетном) равен 1, то искомое τ не сравнимо с 0 по модулю l. Действительно, если $\tau \equiv 0 \pmod{l}$, то

$$(\varphi_l^2 + k)(P) = 0 \quad \text{для всех } P \in E[l],$$

а в нашем случае (при НОД $= 1$) таких точек нет, за исключением $\mathcal{O}$. В случае неразрешимости уравнения $\varphi_l^2(P) = \pm kP$ на $E[l] \setminus \mathcal{O}$ мы переходим на 2-й этап алгоритма.

Предположим, что существует точка $P \in E[l] \setminus \mathcal{O}$, для которой $\varphi_l^2(P) = \pm kP = \pm pP$.

1 случай. Если $\varphi_l^2(P) = -pP$, то $(\varphi_l^2 + p)P = \mathcal{O}$. Так как $(\varphi^2 - t\varphi + p)(P) = \mathcal{O}$ для любой точки P на кривой, то отсюда $(t\varphi_l)(P) = \mathcal{O}$. Поскольку φ_l — изоморфизм $E[l]$, получаем, что $t \equiv 0 \pmod{l}$. Таким образом, в данном случае мы нашли искомое $t \pmod{l}$.

2 случай. Если $\varphi_l^2(P) = pP$, то, пользуясь снова уравнением $\varphi^2 - t\varphi + p = 0$, получаем соотношение $(2p - t\varphi_l)(P) = \mathcal{O}$. Поскольку $2pP \neq \mathcal{O}$ для $P \in E[l] \setminus \mathcal{O}$, то $t \not\equiv 0 \pmod{l}$. Поэтому $\varphi_l(P) = \frac{2p}{t} \cdot P$ (здесь $\frac{1}{t}$ обозначает $t^{-1} \pmod{l}$). Применяя снова φ_l, получим

$$pP = \varphi_l^2(P) = \frac{2p}{t}\varphi_l(P) = \frac{4p^2}{t^2} \cdot P.$$

Отсюда

$$p \equiv 4p^2/t^2 \pmod{l},$$

или $t^2 \equiv 4p \pmod{l}$. В частности, в этом случае p является квадратичным вычетом по модулю l.

Теперь решим уравнение $w^2 \equiv p \pmod{l}$ (алгоритм решения таких уравнений см. далее в главе 6; для малых l возможен перебор). Для его

решения w будет выполнено сравнение $t \equiv \pm 2w \pmod{l}$, и нам останется лишь определить истинный знак + или −.

Подставляя $t \equiv \pm 2w \pmod{l}$ в уравнение для φ_l, получим

$$\varphi_l^2 \mp 2w\varphi_l + w^2 = 0,$$

или $(\varphi_l \mp w)^2 = 0$ тождественно на $E[l]$. Поэтому собственное значение линейного отображения φ_l на $E[l]$ может быть только $\pm w \pmod{l}$ (если r — собственное значение, то $(r \mp w)^2 \equiv 0 \pmod{l}$). Проверку существования решения $Q \in E[l]$ уравнения $(\varphi \mp w)Q = \mathcal{O}$ осуществляем так же, как выше мы осуществляли проверку существования решения уравнения $\varphi_l^2(P) = \pm kP$. Точнее, пусть $Q = (x, y)$. Тогда для выполнения равенства $\varphi_l(Q) = \pm wQ$ по первой координате должно выполняться равенство

$$x^p = x - \frac{\psi_{w-1}(x, y)\psi_{w+1}(x, y)}{\psi_w(x, y)^2}.$$

Отсюда при w четном

$$x^p = x - \frac{f_{w-1}(x)f_{w+1}(x)}{f_w(x)^2(x^3 + ax + b)},$$

а при w нечетном

$$x^p = x - \frac{f_{w-1}(x)f_{w+1}(x)(x^3 + ax + b)}{f_w(x)^2}.$$

Существование такой точки $Q \in E[l]$ равносильно тому, что

$$\text{НОД}((x^p - x)f_w(x)^2(x^3 + ax + b) + f_{w-1}(x)f_{w+1}(x),\ f_l(x)) \neq 1$$

при w четном, а при w нечетном

$$\text{НОД}((x^p - x)f_w(x)^2 + f_{w-1}(x)f_{w+1}(x)(x^3 + ax + b),\ f_l(x)) \neq 1.$$

Если теперь такая точка Q существует (т. е. соответствующий НОД не равен 1), то надо определить знак $\pm w$ по второй координате. Если

$$(x^p,\ y^p) = \varphi_l(w) = wQ = \Big(x - \frac{\psi_{w-1}(x, y)\psi_{w+1}(x, y)}{\psi_w(x, y)^2},$$
$$\frac{\psi_{w+2}(x, y)\psi_{w-1}(x, y)^2 - \psi_{w-2}(x, y)\psi_{w+1}(x, y)^2}{4y\psi_w(x, y)^3}\Big),$$

то отсюда

$$y^p = \frac{f_{w+2}(x) \cdot yf_{w-1}^2 - f_{w-2}(x) \cdot yf_{w+1}(x)^2}{4y^4 f_w(x)^3}$$

при w четном и

$$y^p = \frac{f_{w+2}(x)\cdot y^2 f_{w-1}^2 - f_{w-2}(x)\cdot y^2 f_{w+1}(x)^2}{4yf_w(x)^3}.$$

при w нечетном. Пользуясь соотношением $y^2 = x^3 + ax + b$, получаем, что при w четном будет выполняться соотношение $\varphi_l(Q) = wQ$, если

$$\text{НОД}\Big(4f_w(x)^3(x^3+ax+b)^{\frac{p+3}{2}} - f_{w+2}(x)f_{w-1}(x)^2 + f_{w-2}(x)f_{w+1}(x)^2,\ f_l(x)\Big) \neq 1.$$

Соответственно, при w нечетном

$$\text{НОД}\Big(4f_w(x)^3(x^3+ax+b)^{\frac{p-1}{2}} - f_{w+2}(x)f_{w-1}(x)^2 + f_{w-2}(x)f_{w+1}(x)^2,\ f_l(x)\Big) \neq 1.$$

В случае, когда $\varphi_l(Q) = -wQ$, вторая координата меняет знак. Тогда при w четном

$$\text{НОД}\Big(4f_w(x)^3(x^3+ax+b)^{\frac{p+3}{2}} + f_{w+2}(x)f_{w-1}(x)^2 - f_{w-2}(x)f_{w+1}(x)^2,\ f_l(x)\Big) \neq 1,$$

и

$$\text{НОД}\Big(4f_w(x)^3(x^3+ax+b)^{\frac{p-1}{2}} + f_{w+2}(x)f_{w-1}(x)^2 - f_{w-2}(x)f_{w+1}(x)^2,\ f_l(x)\Big) \neq 1.$$

при w нечетном.

Теперь предположим, что не существует точки $Q \in E[l] \setminus \mathcal{O}$ такой, что $\varphi_l(Q) = \pm wQ$. Кроме того, мы находимся в условиях 2 случая, т. е. существует $P \in E[l] \setminus \mathcal{O}$ такая, что $\varphi_l^2(P) = kP$. Покажем, что наше предположение невозможно. У нас $\varphi_l^2(P) = (\pm w)^2 P$; кроме того, выше мы показали, что $(\varphi_l \mp w)^2 P = \mathcal{O}$. Из этих двух соотношений следует, что

$$w^2 P \mp 2w\varphi_l(P) + w^2 P = \mathcal{O},$$

т. е. $2w^2 P = \pm 2w\varphi_l(P)$. Так как $2w \neq 0 \pmod{l}$, то отсюда $\varphi_l(P) = \pm wP$. Мы же предположили, что для всех точек $Q \in E[l] \setminus \mathcal{O}$ (включая и P), равенство $\varphi_l(Q) = \pm wQ$ невозможно. Это означает, что наше предположение было неверным.

Итак, если существует точка $P \in E[l] \setminus \mathcal{O}$ такая, что $\varphi_l^2(P) = kP$, то существует и точка $Q \in E[l] \setminus \mathcal{O}$ такая, что $\varphi_l(Q) = \pm wQ$.

В итоге 1 этап алгоритма реализуется следующим образом. Если не существует точки $P \in E[l] \setminus \mathcal{O}$, для которой $\varphi_l^2(P) = \pm kP$, то мы

переходим на 2 этап алгоритма. Если же такая точка P существует (т. е. соответствующий НОД в начале описания первого этапа не равен 1), то при $\left(\frac{p}{l}\right) = -1$ мы полагаем $t \equiv 0 \pmod{l}$ (поскольку 2 случай невозможен). Если же $\left(\frac{p}{l}\right) = +1$, то мы находим $w \in \mathbb{Z}$ такое, что $w^2 \equiv p \pmod{l}$ и $0 < w < l$. Затем проверяем, будет ли $+w$ или $-w$ собственным значением для φ_l на $E[l]$ (проверку осуществляем так, как это описано выше). Если $\pm w$ не является собственным значением φ_l на $E[l]$, то $t \equiv 0 \pmod{l}$ — искомое значение, как это было объяснено выше.

Если $\varphi_l(Q) = wQ$ для некоторой точки $Q \in E[l] \setminus \mathcal{O}$, то $\varphi_l^2(Q) = pQ$. Тогда, пользуясь соотношением $\varphi_l(Q) = \frac{2p}{t}Q$, найденным ранее, получим

$$w - \frac{2p}{t} \equiv 0 \pmod{l}.$$

Так как $p \equiv w^2 \not\equiv 0 \pmod{l}$, то отсюда $t \equiv 2w \pmod{l}$ есть искомое значение t по модулю l.

Если $\varphi_l(Q) = -wQ$ для некоторой точки $Q \in E[l] \setminus \mathcal{O}$, то аналогично получим

$$w + \frac{2p}{t} \equiv 0 \pmod{l},$$

откуда $t \equiv -2w \pmod{l}$.

Таким образом, если на 1 этапе окажется, что существует $P \in E[l] \setminus \mathcal{O}$, для которой $\varphi_l^2(P) = \pm kP$, то мы определим искомое значение $t \pmod{l}$, и тогда 2-й этап для данного l будет не нужен.

2 этап алгоритма для фиксированного l. Пусть на 1 этапе оказалось, что не существует точки $P \in E[l] \setminus \mathcal{O}$, для которой

$$\varphi_l^2(P) = \pm kP = \pm pP.$$

Тогда искомое значение τ, $\tau \equiv t \pmod{l}$, для которого на $E[l]$ выполняется равенство

$$\varphi_l^2 + p = \tau\varphi_l,$$

будет не равно нулю, $\tau \neq 0$, как это уже показано ранее.

Далее мы перебором τ, $1 \leqslant \tau \leqslant \frac{l-1}{2}$, ищем то значение, для которого соотношение

$$(\varphi_l^2 + p)(P) = \pm\tau\varphi_l(P)$$

выполняется на $E[l]$ тождественно. Для $P=(x,\,y)\in E[l]\setminus\mathcal{O}$ левая часть запишется при $k\equiv p\pmod l$, $1\leqslant k<l$ в виде

$$(x^{p^2},\,y^{p^2})\oplus$$
$$\oplus\Big(x-\frac{\psi_{k-1}(x,\,y)\psi_{k+1}(x,\,y)}{\psi_k(x,\,y)^2},\ \frac{\psi_{k+2}(x,\,y)\psi_{k-1}(x,\,y)^2-\psi_{k-2}(x,\,y)\psi_{k+1}(x,\,y)^2}{4y\psi_k(x,\,y)^3}\Big),$$

причем сложение $\oplus$ на кривой проводится по формуле секущей, поскольку сейчас $\varphi_l^2(P)\neq\pm pP$ для всех $P\in E[l]\setminus\mathcal{O}$. Правая часть, т. е. $\pm\tau\varphi_l(P)=\pm\varphi_l(\tau P)$, имеет вид

$$\Big(x^p-\Big(\frac{\psi_{\tau-1}(x,\,y)\psi_{\tau+1}(x,\,y)}{\psi_\tau(x,\,y)^2}\Big)^p,$$
$$\pm\Big(\frac{\psi_{\tau+2}(x,\,y)\psi_{\tau-1}(x,\,y)^2-\psi_{\tau-2}(x,\,y)\psi_{\tau+1}(x,\,y)^2}{4y\psi_\tau(x,\,y)^3}\Big)^p\Big).$$

Поскольку $(\varphi_l^2+p)(P)=\pm\tau\varphi_l(P)$ тождественно на $E[l]$, нам уже не надо вычислять наибольший общий делитель с $f_l(x)$, а надо проверять делимость на $f_l(x)$. Для написания соответствующих формул надо рассмотреть четыре случая в зависимости от четности k и τ.

Пусть, например, k четно и τ четно. Тогда

$$(x^{p^2},\,y^{p^2})\oplus\Big(x-\frac{f_{k-1}(x)f_{k+1}(x)}{f_k(x)^2(x^3+ax+b)},\ \frac{f_{k+2}(x)f_{k-1}(x)^2-f_{k-2}(x)f_{k+1}(x)^2}{4(x^3+ax+b)f_k(x)^3\cdot y}\Big)=$$
$$=\Big(x^p-\Big(\frac{f_{\tau-1}(x)f_{\tau+1}(x)}{f_\tau(x)^2(x^3+ax+b)}\Big)^p,\ \pm\Big(\frac{f_{\tau+2}(x)f_{\tau-1}(x)^2-f_{\tau-2}(x)f_{\tau+1}(x)^2}{4(x^3+ax+b)f_\tau(x)^3\cdot y}\Big)^p\Big).$$

Пользуясь соотношением $y^2=x^3+ax+b$, нетрудно показать, что левая часть этого равенства имеет вид

$$\Big(\frac{H_1(x)}{H_2(x)},\ \frac{H_3(x)}{H_4(x)}\cdot y^{\pm1}\Big),$$

а правая часть —

$$\Big(\frac{H_5(x)}{H_6(x)},\ \pm\frac{H_7(x)}{H_8(x)}\cdot y^{\pm1}\Big),$$

где $H_1(x),\,\ldots,\,H_8(x)\in\mathbb{Z}/p\mathbb{Z}[x]$. Тогда для проверки тождества

$$(\varphi_l^2+p)(P)=\pm\tau\varphi_l(P)$$

по первой координате нам нужно проверить, что $H_1(x)H_6(x)-H_5(x)H_2(x)$ делится на $f_l(x)$ в $\mathbb{Z}/p\mathbb{Z}[x]$. Если это не так, то мы переходим к следующему τ. Если это так, то мы выбираем знак $\pm$ числа τ, выполняя

аналогичную проверку по второй координате. Точнее, мы приводим разность

$$\frac{H_3(x)}{H_4(x)} \cdot y^{\pm 1} \mp \frac{H_7(x)}{H_8(x)} \cdot y^{\pm 1}$$

к виду $\dfrac{H_9(x)}{H_{10}(x)} \cdot y^{\pm 1}$, где $H_9(x)$, $H_{10}(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, и затем проверяем делимость $H_9(x)$ на $f_l(x)$. Три остальных возможности для четности k и τ рассматриваются аналогично. На этом заканчивается 2 этап для фиксированного l.

3 этап алгоритма. Пусть мы нашли $t \pmod l$ для простых l таких, что $\prod l > 4\sqrt{p}$. По китайской теореме об остатках находим $t \pmod{\prod l}$; абсолютно наименьший вычет в этом классе вычетов и есть искомое t. Тогда полагаем $|E(\mathbb{Z}/p\mathbb{Z})| = p + 1 - t$.

Конец алгоритма Шуфа.

Замечание 4.13. При фиксированном $l > 2$ вычисления с многочленами и рациональными функциями следует проводить по модулю многочлена $f_l(x)$. То есть нам не нужно, например, использовать x^p; это многочлен очень высокой степени, поскольку p велико. Вместо этого мы вычисляем $x^p \pmod{f_l(x)}$ — многочлен степени, не превосходящей $\deg f_l(x) - 1 = \dfrac{l^2 - 1}{2} - 1$.

Замечание 4.14. Основная трудоемкость алгоритма Шуфа заключается именно в вычислении высоких степеней x^p, y^p, x^{p^2}, y^{p^2} по модулю $f_l(x)$.

Замечание 4.15. В работе [133] описано эффективное сочетание алгоритма Шуфа с различными его усовершенствованиями для оптимального вычисления порядка группы точек эллиптической кривой над конечным простым полем.

§ 4.4. Тестирование чисел на простоту с помощью эллиптических кривых

В 1986 г. Голдвассер и Килиан [126] предложили вероятностный алгоритм проверки простоты чисел с помощью эллиптических кривых.

Теорема 4.16 (см. [126]). *Существует вероятностный алгоритм доказательства простоты натуральных чисел с помощью эллиптических кривых. При этом для любого натурального числа k доля k-значных простых чисел, для которых сред-*

нее время работы алгоритма полиномиально, будет не меньше, чем

$$1 - O(2^{-k^{c/\log\log k}}).$$

Замечание 4.17. В предположении некоторой недоказанной гипотезы о распределении простых чисел среднее время работы алгоритма Голдвассер—Килиана будет полиномиальным для всех простых чисел.

Замечание 4.18. Алгоритм случайным образом выбирает эллиптическую кривую и проверяет выполнение некоторых условий. Он либо выдает верный ответ, является ли данное число простым или составным, либо делает следующий случайный выбор. Алгоритм работает до тех пор, пока либо проверка простоты не будет проведена, либо не закончится отведенное для работы компьютера время.

Замечание 4.19. Чередуя тест Голдвассер—Килиана с тестом Соловея—Штрассена (или Миллера—Рабина) из гл. 1, мы получаем вероятностный метод доказательства того, что данное натуральное число является простым или составным. Среднее время его работы будет полиномиальным для всех k-значных чисел за исключением, может быть, указанной в теореме доли k-значных простых чисел.

Замечание 4.20. Если алгоритм Голдвассер—Килиана доказал простоту числа, то он также выдает «сертификат простоты». С помощью этого сертификата вторичная проверка простоты данного k-значного числа может быть детерминированно проведена за $O(k^{3+\varepsilon})$ битовых операций.

Замечание 4.21. В работе [47] улучшена оценка количества тех простых чисел, для которых среднее время работы алгоритма Голдвассер—Килиана полиномиально. Доказано, что количество простых чисел, не превосходящих x, для которых среднее время работы не является полиномиальным, не превосходит $O(x^{15/16})$.

Мы опишем схему работы алгоритма Голдвассер—Килиана для проверки простоты нечетного, не делящегося на 3 натурального числа n. Мы будем рассматривать эллиптическую кривую E_n над кольцом $\mathbb{Z}/n\mathbb{Z}$, определенную уравнением

$$y^2 \equiv x^3 + ax + b \pmod{n}, \qquad (4a^3 + 27b^2,\ n) = 1.$$

Для множества

$$E_n(\mathbb{Z}/n\mathbb{Z}) = \{(x,\ y) \mid x,\ y \in \mathbb{Z}/n\mathbb{Z},\ \ y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$$

мы будем использовать тот же закон сложения, который был описан в § 4.1 для случая простого n. При этом для $q \in \mathbb{N}$, $P \in E_n(\mathbb{Z}/n\mathbb{Z})$ крат-

ную точку qP мы вычисляем рекуррентно по следующему правилу:

$$qP = \begin{cases} 2\left(\frac{q}{2}P\right), & \text{если } q \text{ четно,} \\ P \oplus (q-1)P, & \text{если } q \text{ нечетно.} \end{cases}$$

Также мы будем использовать редукцию: если p — простое число, $p|n$, и $P = (x,\ y) \in E_n(\mathbb{Z}/n\mathbb{Z})$, то

$$(P)_p = (x \pmod p,\ y \pmod p) \in E_p(\mathbb{Z}/p\mathbb{Z}).$$

Здесь $E_p(\mathbb{Z}/p\mathbb{Z}) = \{(x,\ y) \mid x,\ y \in \mathbb{Z}/p\mathbb{Z},\ y^2 = x^3 + ax + b \pmod p\} \cup \mathcal{O}$ является уже настоящей эллиптической кривой над полем $\mathbb{Z}/p\mathbb{Z}$, поскольку $4a^3 + 27b^2 \neq 0 \pmod p$. Очевидно, что если $P,\ Q \in E_n(\mathbb{Z}/n\mathbb{Z})$ и точка $P \oplus Q$ определена, то $(P \oplus Q)_p = P_p \oplus Q_p$.

Схема алгоритма Голдвассер—Килиана.

1 шаг. Полагаем $p_0 = n$, $i = 0$. Выбираем $k \in \mathbb{N}$ такое, что $2^{k-1} < p_0 < 2^k$.

2 шаг. Случайно выбираем $A,\ B \in \mathbb{Z}/p_i\mathbb{Z}$ и проверяем условие $D = (4A^3 + 27B^2,\ p_i) = 1$. Если $i = 0$ и данный наибольший общий делитель лежит в интервале $(1;\ p_0)$, то $p_0 = n$ — составное, и алгоритм заканчивает работу. Если $i > 0$ и $1 < D < p_i$, то возвращаемся на 1-й шаг. Если $i > 0$ и $D = p_i$, то возвращаемся на 2-й шаг (т. е. выбираем другие A, B).

3 шаг. В предположении, что p_i — простое число, ищем для редуцированной кривой $y^2 = x^3 + ax + b \pmod{p_i}$ величину $|E_{p_i}(\mathbb{Z}/p_i\mathbb{Z})|$ (например, с помощью алгоритма Шуфа из § 4.3). Если найденное значение $|E_{p_i}(\mathbb{Z}/p_i\mathbb{Z})|$ нечетно, то возвращаемся на 2-й шаг. В противном случае полагаем

$$q = |E_{p_i}(\mathbb{Z}/p_i\mathbb{Z})|/2$$

и проверяем выполнение теоремы Хассе

$$|2q - p_i - 1| < 2\sqrt{p_i}.$$

Если это последнее неравенство не выполняется для $i > 0$, то мы возвращаемся на 1 шаг. Если оно не выполняется при $i = 0$, то $n = p_0$ — составное.

4 шаг. Делаем l проходов вероятностного теста Соловея—Штрассена (или Миллера—Рабина), описанного в гл. 1, для проверки простоты q. Если оказалось, что q — составное, то возвращаемся на 2 шаг. Значение l выбираем так, чтобы выполнялось неравенство $\left(\frac{1}{2}\right)^l \leqslant 1/p^3$.

5 шаг. Выбираем случайную точку $P=(x,\ y)$, $P\in E_{p_i}(\mathbb{Z}/p_i\mathbb{Z})$. То есть мы случайно выбираем $x\in\mathbb{Z}/p_i\mathbb{Z}$ и при $\left(\dfrac{x^3+ax+b}{p}\right)=1$ находим $y\equiv(x^3+ax+b)^{1/2}\pmod p$, затем полагаем $P=(x,\ y)$; иначе делаем следующий выбор x.

6 шаг. Найдя $P=(x,\ y)\in E_{p_i}(\mathbb{Z}/p_i\mathbb{Z})$, мы проверяем выполнение равенства $2qP=\mathcal{O}$ на $E_{p_i}(\mathbb{Z}/p_i\mathbb{Z})$. Если это равенство не выполняется и $i>0$, то мы возвращаемся на 1 шаг. Если оно не выполняется и $i=0$, то n составное. Если же оно выполнено, то мы полагаем $p_{i+1}=q$.

7 шаг. Проверяем выполнение неравенства

$$q\leqslant 2^{k^{c/\log\log k}}.$$

Здесь постоянная c взята из оценки сложности

$$O((\log n)^{c\log\log\log n})$$

алгоритма проверки простоты чисел Адлемана—Померанса—Румели или алгоритма Ленстры, о которых было рассказано в гл. 1. Если неравенство не выполняется, то мы полагаем $i:=i+1$ и возвращаемся на 2 шаг. Если же неравенство для q выполнено, то мы детерминированно проверяем простоту q с помощью алгоритма Адлемана—Померанса—Румели или алгоритма Ленстры. Если q окажется составным, то мы возвращаемся на 1 шаг. Иначе алгоритм заканчивает работу и выдает ответ, что число n — простое.

Конец алгоритма.

Обоснование корректности работы алгоритма основано на следующем утверждении.

Утверждение 4.22. *Пусть $n\in\mathbb{N}$, $n>1$, $(n,\ b)=1$, E_n — эллиптическая кривая над $\mathbb{Z}/n\mathbb{Z}$, $P=(x,\ y)\in E_n(\mathbb{Z}/n\mathbb{Z})$, $P\neq\mathcal{O}$. Пусть q — простое число, $q>n^{1/2}+2n^{1/4}+1$, $qP=\mathcal{O}$. Тогда n — простое число.*

Доказательство. Предположим, что n — составное, и обозначим через p простой делитель n, $p\leqslant\sqrt{n}$. При редукции по модулю p мы получим точку $M_p=(P)_p$, для которой

$$M_p\neq\mathcal{O}_p,\qquad qM_p=\mathcal{O}_p.$$

Отсюда по теореме Лагранжа

$$|E_p(\mathbb{Z}/p\mathbb{Z})|\geqslant q>n^{1/2}+2n^{1/4}+1\geqslant p+2\sqrt{p}+1.$$

Поэтому

$$|E_p(\mathbb{Z}/p\mathbb{Z})| - (p+1) \geqslant 2\sqrt{p},$$

что противоречит теореме Хассе. Утверждение доказано. □

В алгоритме Голдвассер—Килиана в случае успеха будет построена цепочка

$$n = p_0 > p_1 > \ldots > p_l,$$

и из простоты p_l будет следовать простота n, согласно доказанному утверждению. Действительно, очередное q удовлетворяет неравенству

$$|2q - p_i - 1| < 2p_i^{1/2},$$

это проверяется на 3 шаге. Тогда неравенство $q > p_i^{1/2} + 2p_i^{1/4} + 1$ также будет выполнено, так как при $p_i > 5$ справедливо неравенство

$$\frac{p_i - 2p_i^{1/2} + 1}{2} > p_i^{1/2} + 2p_i^{1/4} + 1.$$

В силу доказанного утверждения из простоты q следует простота p_i. Поэтому для построенной в алгоритме цепочки простых чисел из простоты последнего числа следует простота первого числа, т. е. числа n.

Замечание 4.23. Построенная цепочка

$$n = p_0 > p_1 > \ldots > p_l$$

и есть тот сертификат простоты n, о котором говорилось в замечании 4.20.

Алгоритм Голдвассер—Килиана оказался непрактичным из-за многократного использования алгоритма Шуфа для вычисления порядков групп точек выбираемых в алгоритме проверки простоты эллиптических кривых. Аткин и Морейн [56] предложили использовать эллиптические кривые с комплексным умножением. Для таких кривых порядок группы точек находится по несложной формуле; несколько более сложным оказывается процесс построения самих кривых с комплексным умножением. Алгоритм Аткина и Морейна был реализован на компьютере и в настоящее время успешно применяется для проверки простоты чисел, так же как и алгоритмы Адлемана—Померанса—Румели—Ленстры—Коена из гл. 1. Сравнение этих двух методов было проведено в § 2.8, см. также [89, гл. 9]. Дополнительную информацию об алгоритме Аткина—Морейна можно найти в [199].

§ 4.5. Заключение

В заключение сделаем еще несколько замечаний об алгоритмах, использующих эллиптические кривые.

Как мы видели выше, в ряде случаев приходится вычислять кратную точку kP для точки P на эллиптической кривой (здесь $k \in \mathbb{Z}$). Такие вычисления проводятся и во многих криптосистемах. Эффективные алгоритмы для решения этой задачи можно найти в работах [200; 261].

Вычисление порядка группы точек эллиптической кривой над полем $GF(2^l)$ описано в работе [182].

Построение кривых с комплексным умножением описано в работах [189; 154].

В работе [133] отмечено, что если криптосистема использует эллиптические кривые с некоторыми специальными свойствами (например, аномальные или суперсингулярные кривые), то на такую криптосистему возможны более эффективные атаки (см. [235]).

В работе [224] доказано, что с помощью эллиптических кривых можно получить сертификат простоты для каждого простого числа p, проверяемый за $O(\log p)$ арифметических операций. Однако отсутствует оценка на количество таких сертификатов для фиксированного p. Метод этой работы, скорее всего, неприменим для практической проверки простоты чисел.

Глава 5. Алгоритмы дискретного логарифмирования

§ 5.1. Введение. Детерминированные методы

Пусть G — мультипликативная абелева группа, a, $b \in G$. Задача нахождения решения уравнения

$$a^x = b$$

называется задачей дискретного логарифмирования в группе G. Ее решение x называется дискретным логарифмом элемента b по основанию a и обозначается $\log_a b$, если основание a фиксировано и если решение существует; $\log_a b \in \mathbb{Z}/|G|\mathbb{Z}$, если $|G| < \infty$.

Задача дискретного логарифмирования имеет важные приложения в криптографии. Особенно важен случай $G = GF(q)^*$, где $q = p^l$, p — простое число, $l \in \mathbb{N}$, а также случай, когда G является группой точек эллиптической кривой над конечным полем.

Рассмотрим уравнение

$$a^x \equiv b \pmod{p} \tag{5.1}$$

в группе $(\mathbb{Z}/p\mathbb{Z})^*$, где p — простое число. Мы будем предполагать, что порядок $a \pmod{p}$ равен $p-1$. Тогда уравнение разрешимо, и решение x является элементом $\mathbb{Z}/(p-1)\mathbb{Z}$. В данном параграфе мы опишем детерминированные методы решения (5.1).

С помощью перебора можно решить уравнение (5.1) за $O(p)$ арифметических операций.

Решение $\log_a b$ уравнения (5.1) можно находить по формуле

$$\log_a b \equiv \sum_{j=1}^{p-2} (1 - a^j)^{-1} b^j \pmod{p-1},$$

см. [210]. Однако сложность вычисления по этой формуле хуже, чем для перебора.

Следующий алгоритм решения (5.1) имеет сложность $O(p^{1/2} \log p)$ арифметических операций (см. [36, гл. 6]).

Алгоритм согласования.

1 шаг. Присвоить $H := [p^{1/2}] + 1$.

2 шаг. Найти $c \equiv a^H \pmod p$.

3 шаг. Составить таблицу значений $c^u \pmod p$, $1 \leqslant u \leqslant H$, и упорядочить ее.

4 шаг. Составить таблицу значений $b \cdot a^v \pmod p$, $0 \leqslant v \leqslant H$, и упорядочить ее.

5 шаг. Найти совпавшие элементы из первой и второй таблиц. Для них

$$c^u \equiv b \cdot\ a^v \pmod p,$$

откуда $a^{Hu-v} \equiv b \pmod p$.

6 шаг. Выдать $x \equiv Hu - v \pmod{p-1}$.

Конец алгоритма.

Докажем, что алгоритм работает корректно. Любое целое число x, $0 \leqslant x \leqslant p-2$, можно представить в виде $x \equiv Hu - v \pmod{p-1}$, где $1 \leqslant u \leqslant H$, $0 \leqslant v \leqslant H$. Действительно, набор чисел H, $H-1$, $H-2$, …, $H-H$, $2H$, $2H-1$, …, $2H-H$, …, H^2, H^2-1, …, H^2-H содержит в себе набор чисел $0, 1, \ldots, p-2$, поскольку $H^2 > p$. Из этого следует корректность алгоритма. Оценка сложности также очевидна, поскольку набор из N элементов можно упорядочить за $O(N \log N)$ арифметических операций, см. [5, гл. 3].

Замечание 5.1. Некоторые усовершенствования алгоритма согласования см. в [36, гл. 6]. Все они также имеют экспоненциальную сложность.

Предположим теперь, что известно разложение $p-1$ на простые множители:

$$p - 1 = \prod_{i=1}^{s} q_i^{\alpha_i}.$$

Тогда решение (5.1) можно найти за $O\Bigl(\sum\limits_{i=1}^{s} \alpha_i(\log p + q_i)\Bigr)$ арифметических операций с помощью следующего алгоритма, см. [215]. (Для некоторого усовершенствования алгоритма справедлива аналогичная оценка с заменой q_i на $q_i^{1/2}$.)

Алгоритм Полига—Хеллмана.

1 шаг. Для каждого простого числа q, $q \mid p-1$, составляем таблицу чисел

$$r_{q,j} \equiv\ a^{j(p-1)/q} \pmod p, \quad j = 0, \ldots, q-1.$$

2 шаг. Для каждого простого q, $q^\alpha \parallel p-1$, находим $\log_a b \pmod{q^\alpha}$.

Пусть

$$x \equiv \log_a b \pmod{q^\alpha} \equiv x_0 + x_1 q + \ldots + x_{\alpha-1} q^{\alpha-1} \pmod{q^\alpha},$$

где $0 \leqslant x_i \leqslant q-1$. Тогда из (5.1) следует, что

$$b^{(p-1)/q} \equiv a^{x_0(p-1)/q} \pmod{p}.$$

С помощью таблицы 1 шага находим x_0. Тогда выполнено сравнение

$$(ba^{-x_0})^{(p-1)/q^2} \equiv a^{x_1(p-1)/q} \pmod{p}.$$

По таблице находим x_1, и т. д. Значение x_i находится из сравнения

$$(ba^{-x_0-x_1q-\ldots-x_{i-1}q^{i-1}})^{(p-1)/q^{i+1}} \equiv a^{x_i(p-1)/q} \pmod{p}.$$

3 шаг. Найдя $\log_a b \pmod{q_i^{\alpha_i}}$, $i = 1, \ldots, s$, находим $\log_a b \pmod{p-1}$ по китайской теореме об остатках.

Конец алгоритма.

Докажем оценку сложности алгоритма. Набор элементов $a^{(p-1)/q_i} \pmod{p}$ вычисляется за $\sum\limits_{i=1}^{s} O(\log p)$ арифметических операций. Затем набор $r_{q_i,j}$ для всех q_i, j вычисляется за $\sum\limits_{i=1}^{s} O(q_i)$ арифметических операций. Для нахождения очередного x_i на шаге 3 надо возвести в степень (т. е. найти $a^{x_{i-1}q^{i-1}}$), найти обратный элемент, умножить, возвести в степень и пройти по таблице. Обратный элемент находится с помощью обобщенного алгоритма Евклида за $O(\log p)$ операций. Все вместе дает указанную выше оценку сложности алгоритма Полига—Хеллмана.

Замечание 5.2. Алгоритм Полига—Хеллмана имеет полиномиальную сложность $O((\log p)^{c_1})$ в случае, когда все простые делители q_i числа p не превосходят $(\log p)^{c_2}$, где c_1, c_2 — положительные постоянные. Это имеет место, например, для простых чисел p вида $p = 2^\alpha + 1$, $p = 2^{\alpha_1}3^{\alpha_2} + 1$. Если же у $p-1$ есть простой делитель q, $q \geqslant p^c$, где $c > 0$, то алгоритм Полига—Хеллмана будет иметь экспоненциальную сложность.

§ 5.2. ρ-метод Полларда для дискретного логарифмирования

В § 2.3 мы описали ρ-метод Полларда для факторизации целых чисел. В работе [220] аналогичный метод был предложен для дискретного

логарифмирования по простому модулю p. Мы хотим решить уравнение $a^x \equiv b \pmod p$. Для этого рассмотрим три числовые последовательности

$$\{u_i\},\ \{v_i\},\ \{z_i\}, \qquad i = 0,\ 1,\ 2,\ \ldots,$$

определенные следующим образом:

$$u_0 = v_0 = 0, \qquad z_0 = 1;$$

$$u_{i+1} \equiv \begin{cases} u_i + 1 \pmod{p-1}, & \text{если} \quad 0 < z_i < p/3; \\ 2u_i \pmod{p-1}, & \text{если} \quad p/3 < z_i < \dfrac{2}{3}p; \\ u_i \pmod{p-1}, & \text{если} \quad \dfrac{2}{3}p < z_i < p; \end{cases}$$

$$v_{i+1} \equiv \begin{cases} v_i \pmod{p-1}, & \text{если} \quad 0 < z_i < p/3; \\ 2v_i \pmod{p-1}, & \text{если} \quad p/3 < z_i < \dfrac{2}{3}p; \\ v_i + 1 \pmod{p-1}, & \text{если} \quad \dfrac{2}{3}p < z_i < p; \end{cases}$$

$$z_{i+1} \equiv b^{u_{i+1}} a^{v_{i+1}} \pmod{p-1}.$$

Здесь под $c \pmod p$ мы понимаем наименьший неотрицательный вычет в данном классе вычетов.

Далее мы рассматриваем наборы $(z_i, u_i, v_i, z_{2i}, u_{2i}, v_{2i})$, $i = 1, 2, 3, \ldots$, и ищем номер i, для которого $z_i = z_{2i}$. Из последнего равенства следует, что

$$b^{u_{2i} - u_i} \equiv a^{v_i - v_{2i}} \pmod p.$$

Если окажется, что $(u_{2i} - u_i,\ p-1) = 1$, то при $l \in \mathbb{Z}$, $l(u_{2i} - u_i) \equiv 1 \pmod{p-1}$ мы получим

$$b \equiv a^{l(v_i - v_{2i})} \pmod p,$$

откуда искомый x равен $\log_a b \equiv l(v_i - v_{2i}) \pmod{p-1}$.

Дальнейшие детали, в частности, методы для нахождения совпавших элементов z_i, z_{2i}, см. в [220]. Эвристическая оценка сложности метода составляет $O(p^{1/2})$ операций.

Заметим, что в [276] с помощью некоторого усовершенствования ρ-метода Полларда было проведено дискретное логарифмирование по простому модулю, записываемому 22 десятичными цифрами.

В работе [267] также было предложено некоторое усовершенствование описанного выше метода.

§ 5.3. Дискретное логарифмирование в простых полях

В данном параграфе мы рассмотрим алгоритмы решения уравнения

$$a^x \equiv b \pmod{p}, \tag{5.2}$$

где p — простое число, имеющие эвристическую оценку сложности $L_p\left[\frac{1}{2}; c\right]$ при некоторых значениях постоянной c. Мы будем считать, что $a \pmod{p}$ имеет порядок $p-1$. Первый такой алгоритм предложил Адлеман в работе [44]. Мы опишем некоторую модификацию его метода.

Алгоритм Адлемана.

1 этап. Сформировать факторную базу, состоящую из всех простых чисел q, $q \leqslant B = e^{\text{const}\sqrt{\log p \log\log p}}$.

2 этап. С помощью некоторого перебора найти натуральные числа r_i такие, что

$$a^{r_i} \equiv \prod_{\substack{q \leqslant B,\\ q \text{ — простое}}} q^{\alpha_{iq}} \pmod{p}$$

Отсюда следует, что

$$r_i \equiv \sum_{\substack{q \leqslant B,\\ q \text{ — простое}}} \alpha_{iq} \log_a q \pmod{p-1}. \tag{5.3}$$

3 этап. Набрав достаточно много соотношений (5.3), решить получившуюся систему линейных уравнений относительно неизвестных $\log_a q$ — дискретных логарифмов элементов факторной базы.

4 этап. С помощью некоторого перебора найти одно значение r, для которого

$$a^r \cdot b \equiv \prod_{q \leqslant B} q^{\beta_q} \cdot p_1 \ldots p_k \pmod{p},$$

где $p_1, \ldots, p_k$ — простые числа «средней» величины, т. е. $B < p_i < B_1$, где B_1 — также некоторая субэкспоненциальная граница, $B_1 = e^{\text{const}\sqrt{\log p \log\log p}}$.

5 этап. С помощью вычислений, аналогичных 2 и 3 этапам алгоритма, найти дискретные логарифмы $\log_a p_i$ для фиксированных простых чисел средней величины $p_1, \ldots, p_k$ из 4 этапа.

6 этап. Определить искомый $\log_a b$:

$$\log_a b \equiv -r + \sum_{q \leqslant B} \beta_q \log_a q + \sum_{i=1}^{k} \log_a p_i \pmod{p-1}.$$

Конец алгоритма.

Замечание 5.3. Идея использования факторной базы для нахождения дискретных логарифмов применялась и ранее, например, в [280]. Описания алгоритма Адлемана см. также в [95; 176; 210]. Отметим, что на практике алгоритм Адлемана все же недостаточно эффективен.

В 1986 г. Копперсмит, Одлыжко и Шреппель предложили алгоритм дискретного логарифмирования с эвристической оценкой сложности $L_p\left[\frac{1}{2}; 1\right]$ арифметических операций. В работе [151] в 1991 г. с помощью метода [97] (версия с гауссовыми целыми) было проведено логарифмирование по модулю $p \asymp 10^{58}$. В 1997 г. Вебер [276] провел логарифмирование по модулю $p \asymp 10^{85}$ также с помощью версии алгоритма [97] с гауссовыми целыми. Он, кроме того, показал, что метод [97] с гауссовыми целыми лучше, чем решето числового поля, для данного $p \asymp 10^{85}$. В работе [134] экспериментально показано, что для $p \leqslant 10^{90}$ метод [97] лучше решета числового поля. Однако для $p > 10^{100}$ алгоритмы решета числового поля работают быстрее алгоритма Копперсмита—Одлыжко—Шреппеля, что также было показано в [134]. Об алгоритмах решета числового поля для дискретного логарифмирования, о версии [97] с гауссовыми целыми и о работе [276] мы расскажем более подробно далее, в § 5.5 этой главы. Здесь же мы опишем простейшую версию алгоритма Копперсмита—Одлыжко—Шреппеля.

Алгоритм COS.

1 этап. Положим $H = [p^{1/2}] + 1$, $J = H^2 - p > 0$. Сформируем множество

$$S = \{q \mid q \text{ — простое, } q < L^{1/2}\} \cup \{H + c \mid 0 < c < L^{1/2+\varepsilon}\},$$

где L и ε — постоянные величины, $L = L_p\left[\frac{1}{2}; 1\right]$, $0 < \varepsilon < 1$.

2 этап. С помощью некоторого просеивания мы ищем пары целых чисел c_1, c_2 таких, что $0 < c_i < L^{\frac{1}{2}+\varepsilon}$, $i = 1, 2$, и абсолютно наименьший вычет элемента $(H + c_1)(H + c_2) \pmod p$ гладок по отношению к границе

гладкости $L^{1/2}$, т. е.

$$(H+c_1)(H+c_2) \equiv \prod_{\substack{q<L^{1/2},\\ q \text{ — простое}}} q^{\alpha_q(c_1,c_2)} \pmod p.$$

При этом, поскольку $J = O(p^{1/2})$, то

$$(H+c_1)(H+c_2) \equiv J + (c_1+c_2)H + c_1c_2 \pmod p,$$

причем абсолютно наименьший вычет в этом классе вычетов равен $J + (c_1+c_2)H + c_1c_2$ и имеет величину $O(p^{1/2+\varepsilon})$. Поэтому вероятность его гладкости выше, чем для произвольных чисел на отрезке $[1,\ p-1]$. Логарифмируя по основанию a, получим соотношение

$$\log_a(H+c_1) + \log_a(H+c_2) \equiv \sum_{\substack{q<L^{1/2},\\ q \text{ — простое}}} \alpha_q(c_1,\ c_2)\log_a\ q \pmod{p-1}.$$

Это однородное уравнение относительно неизвестных величин $\log_a(H+c)$, $\log_a q$. Мы можем считать, что a также является $L^{1/2}$-гладким, $a = \prod\limits_{q<L^{1/2}} q^{\beta_q}$, откуда получим неоднородное уравнение

$$1 \equiv \sum_q \beta_q \log_a\ q \pmod{p-1}.$$

3 этап. Набрав на 2-м этапе достаточно много уравнений, мы решим получившуюся систему линейных уравнений в кольце $\mathbb{Z}/(p-1)\mathbb{Z}$ и найдем значения $\log_a(H+c)$, $\log_a q$.

4 этап. Для нахождения конкретного логарифма $x = \log_a b$ мы введем новую границу гладкости L^2. Случайным перебором находим одно значение w такое, что

$$a^w b \equiv \prod_{\substack{q<L^{1/2},\\ q \text{ — простое}}} q^{g_q} \prod_{\substack{L^{1/2}\leqslant u<L^2,\\ u \text{ — простое}}} u^{h_u} \pmod p.$$

В этом соотношении участвуют несколько новых простых чисел u средней величины.

5 этап. С помощью методов, аналогичных 2 и 3 этапам, мы находим логарифмы нескольких простых чисел u средней величины, возникших на 4 этапе.

6 этап. Находим ответ

$$x = \log_a b \equiv -w + \sum_{\substack{q<L^{1/2},\\ q \text{ — простое}}} g_q \log_a q + \sum_u h_u \log_a\ u \pmod{p-1}.$$

Конец алгоритма.

Замечание 5.4. Значения $\log_a(H+c)$, возникающие на 2 этапе, являются несущественными. Они не нужны впоследствии для нахождения индивидуальных логарифмов $\log_a b$. Поэтому на 3 этапе их сперва исключают из системы линейных уравнений; техника такого исключения описана в [103]. Получившуюся систему относительно неизвестных $\log_a q$ решают методом структурированного гауссова исключения с последующим применением алгоритма Ланцоша, как в работе [151], или сразу методом Ланцоша, как в работе [276].

Замечание 5.5. Опишем просеивание, применяемое на 2 этапе. Зафиксируем значение c_1. Пусть q — простое число, f — небольшое натуральное число, причем q^f делит $J+(c_1+c_2)H+c_1c_2$. Тогда

$$c_2 \equiv -(J+c_1H)(H+c_1)^{-1} \pmod{q^g}.$$

Отсюда следует, что значения c_2 для данных g и f лежат в арифметической прогрессии. Теперь уже ясно, что просеивание мы можем проводить аналогично методу квадратичного решета, описанному в гл. 3. Мы заводим массив из $L^{1/2+\varepsilon}$ элементов, номер элемента — это значение c_2. Сначала элементы массива равны 0. Затем мы перебираем степени простых чисел q^f и для фиксированного q^f к элементам массива с номерами c_2, $c_2 \equiv -(J+c_1H)(H+c_1)^{-1} \pmod{q^f}$, прибавляем достаточно грубо вычисленные значения $\log q$. После окончания просеивания по всем $q < L^{1/2}$ и некоторым небольшим f элементы нашего массива будут приближенно равны логарифмам от гладких частей элементов $J+(c_1+c_2)H+c_1c_2$. Если значение элемента массива с номером c_2 окажется примерно равно значению $\log|J+(c_1+c_2)H+c_1c_2|$, то число $J+(c_1+c_2)H+c_1c_2$ скорее всего будет гладким, и мы факторизуем его пробными делениями на a, $a < L^{1/2}$. Смысл применения просеивания, как и в методе квадратичного решета для факторизации, заключается в сокращении количества делений.

Замечание 5.6. В работе [134] описано логарифмирование по простому модулю $p = [10^{89}\pi] + 156137$, $\dfrac{p-1}{2}$ — простое. Использовалась версия алгоритма COS с гауссовыми целыми. Для работы алгоритма потребовалось 60 mips-years для нахождения соотношений, и около трех недель ушло на решение системы линейных уравнений. При решении системы линейных уравнений сначала использовалось структурированное гауссово исключение, а затем к уплотненной системе меньшего размера применялся алгоритм Ланцоша.

§ 5.4. Дискретное логарифмирование в полях Галуа

Фиксируем простое число p, натуральное число $n > 1$, и обозначим $q = p^n$. Пусть a — образующий элемент циклической группы $GF(q)^*$. Мы хотим решить уравнение

$$a^x = b \tag{5.4}$$

в $GF(q)$. Для этого также используются алгоритмы с факторными базами. В случае, когда p невелико, такой алгоритм описан в [144, гл. 4]. Он имеет эвристическую оценку сложности $L_q\left[\frac{1}{2}; \text{const}\right]$ арифметических операций.

Алгоритм index-calculus.

1 этап. (Предварительные вычисления.) Поле $GF(q)$ изоморфно $GF(p)[y]/(f(y))$, где $f(y) \in GF(p)[y]$ — неприводимый унитарный многочлен степени n. Поэтому элементы поля $GF(q)$ представимы в виде многочленов степени не более $n-1$. Умножение таких многочленов производится по модулю $f(y)$. В частности, $a = a(y)$ — некоторый многочлен. Элемент $a_1 = a^{(q-1)/(p-1)}$ имеет порядок $p-1$ и образует $GF(p)^*$. С его помощью мы составляем таблицу логарифмов «констант» — т. е. элементов простого поля $GF(p) \subseteq GF(q)$. Для этого мы вычисляем $a_1^0 = 1,\ a_1,\ a_1^2,\ \ldots,\ a_1^{p-2}$; это делается быстро, так как мы предположили, что p невелико.

2 этап. (Выбор факторной базы.) Факторная база $\mathbb{B} \subseteq GF(q)$ состоит из всех неприводимых многочленов g степени не выше t, где t — некоторый параметр, $t < n$ (выбор параметра t связан с оценкой сложности алгоритма).

3 этап. (Нахождение соотношений.) Случайно перебирая m, $1 \leqslant m \leqslant q-2$, мы находим те значения, для которых выполнено соотношение

$$a^m \equiv c_0 \prod_{g \in \mathbb{B}} g^{\alpha_g(m)} \pmod{f(y)},$$

где $c_0 \in GF(p)$. Для факторизации нам приходится использовать деление многочленов над конечным полем. Из найденного соотношения следует, что

$$m \equiv \log_a c_0 + \sum_{g \in \mathbb{B}} \alpha_g(m) \log_a g \pmod{q-1}.$$

Здесь $\log_a c_0$ нам уже известны, а $\log_a g$ — неизвестные величины.

4 этап. (Нахождение алгоритмов элементов факторной базы.) Найдя на 3 этапе достаточно много соотношений (больше, чем $|\mathbb{B}|$), мы решаем систему линейных уравнений в кольце $\mathbb{Z}/(q-1)\mathbb{Z}$ и находим $\log_a g$ для $g \in \mathbb{B}$.

5 этап. (Нахождение индивидуального логарифма.) В простейшей версии алгоритма мы ищем одно значение m, для которого

$$b \cdot a^m \equiv c_1 \prod_{g \in \mathbb{B}} g^{\beta_g} \pmod{f(x)},$$

где $c_1 \in GF(p)$. Отсюда находим искомое значение

$$\log_a b \equiv -m + \log_a c_1 + \sum_{g \in \mathbb{B}} \beta_q \log_a g \pmod{q-1}.$$

Конец алгоритма.

В случае, когда p велико, описанный выше алгоритм неэффективен. Для логарифмирования в поле $GF(p^2)$ Эль Гамаль в работе [108] предложил алгоритм, имеющий оценку сложности $L_p\left[\frac{1}{2}; \text{const}\right]$ арифметических операций. В его методе используется вложение поля $GF(p^2)$ в кольцо целых алгебраических чисел $\mathbb{Z}_{\mathbb{K}}$ некоторого мнимого квадратичного одноклассного поля $\mathbb{K}$

$$f\colon GF(p^2) \to \mathbb{Z}_{\mathbb{K}}.$$

Одноклассность поля $\mathbb{K}$ означает, что в $\mathbb{Z}_{\mathbb{K}}$ есть однозначное разложение на простые множители, и каждый идеал является главным. Наше уравнение $a^x = b$ в $GF(p^2)$ тогда перейдет в уравнение $\alpha^x = \beta$ в $\mathbb{Z}_{\mathbb{K}}$, где $\alpha = f(a)$, $\beta = f(b)$. Далее используется стандартная схема логарифмирования с факторной базой, неоднократно описанная нами выше. При этом в качестве элементов факторной базы используются простые (неразложимые) элементы кольца $\mathbb{Z}_{\mathbb{K}}$ с небольшой нормой.

Заметим, что алгоритм Эль Гамаля явился, по сути, предвестником алгоритма решета числового поля.

Еще один алгоритм был предложен Эль Гамалем в работе [109]. Это алгоритм дискретного логарифмирования в поле $GF(p^n)$, где p — большое простое число, $n > 2$. Он также имеет эвристическую субэкспоненциальную оценку сложности $L_{p^n}\left[\frac{1}{2}; \text{const}\right]$. В нем используется представление $GF(p^n)$ в виде $\mathbb{Z}_{\mathbb{K}}/\mathfrak{P}$, где $\mathbb{Z}_{\mathbb{K}}$ — кольцо целых алгебраических чисел числового поля $\mathbb{K}$, $\mathfrak{P}$ — простой идеал $\mathbb{Z}_{\mathbb{K}}$ с нормой, равной p^n.

Теперь рассмотрим случай $p=2$, $q=2^n$. В 1984 г. в работе [95] Д. Копперсмит предложил алгоритм, который имеет эвристическую оценку сложности $L_q\left[\frac{1}{3}; \text{const}\right]$. Это был первый субэкспоненциальный алгоритм с показателем $\frac{1}{3}$; для факторизации алгоритмы с такой оценкой сложности появились лишь в 1990 г. (решето числового поля; см. гл. 3).

Опишем идею алгоритма Копперсмита на примере. Этот алгоритм основан на нахождении хорошего представления поля $GF(q)$ в виде $GF(q)=GF(2)[x]/(P(x))$, где неприводимый многочлен $P(x)$ имеет вид $P(x)=x^n+Q(x)$, $\deg Q(x)<n^{2/3}$. Пусть $n=127$; можно положить $P(x)=x^{127}+x+1$ — неприводимый многочлен в $GF(2)[x]$. Мы будем рассматривать поле $GF(2^{127})=GF(2)[x]/(P(x))$. Пусть $a=a(x)$ — образующий циклической группы $GF(2^{127})^*$. Предположим, что мы выбрали $A(x)$, $B(x)\in GF(2)[x]$, $\deg A(x)\leqslant 10$, $\deg B(x)\leqslant 10$, НОД$(A(x), B(x))=1$. Мы рассматриваем $A(x)$, $B(x)$ как элементы $GF(2^{127})$. Пусть $C(x)=x^{32}A(x)+B(x)$, тогда $\deg C(x)\leqslant 42$. Если мы рассмотрим многочлен $D(x)$,

$$D(x)\equiv C(x)^4 \pmod{P(x)}, \qquad \deg D(x)<127,$$

то

$$D(x)\equiv x^{128}A(x)^4+B(x)^4 \pmod{P(x)},$$

и поскольку

$$x^{128}\equiv x^2+x \pmod{P(x)},$$

то

$$D(x)\equiv (x^2+x)A(x)^4+B(x)^4 \pmod{P(x)}.$$

Таким образом,

$$C(x)^4\equiv D(x) \pmod{P(x)}, \qquad \deg D(x)\leqslant 42, \qquad \deg C(x)\leqslant 42.$$

Поскольку степени $C(x)$ и $D(x)$ невелики, то с достаточно высокой вероятностью они разложатся в произведение неприводимых многочленов малой степени, которые составляют факторную базу. То есть

$$C(x)\equiv \prod_j g_j(x)^{e_j} \pmod{P(x)},$$

$$D(x)\equiv \prod_j g_j(x)^{f_j} \pmod{P(x)}.$$

Тогда сравнение

$$4\sum_j e_j \log_a g_j(x) \equiv \sum_j f_j \log_a g_j(x) \pmod{q-1}$$

есть соотношение для неизвестных $\log_a g_j(x)$ — дискретных логарифмов элементов факторной базы нашего поля $GF(2^{127})$. Это однородные уравнения. Однако можно считать, что наше основание $a = a(x)$ в уравнении (5.4) само есть неприводимый многочлен малой степени, или что оно раскладывается в произведение таких многочленов:

$$a = a(x) \equiv \prod_j g_j(x)^{v_j} \pmod{P(x)}.$$

Это дает нам неоднородное уравнение

$$1 = \log_a a \equiv \sum_j v_j \log_a g_j(x) \pmod{q-1},$$

и мы получаем неоднородную систему линейных уравнений относительно неизвестных величин $\log_a g_j(x)$. В остальном схема алгоритма Копперсмита аналогична схеме алгоритма Адлемана, описанного в § 5.3.

В работах [269; 270] с помощью некоторого усовершенствования алгоритма Копперсмита было проведено дискретное логарифмирование в поле $GF(2^{607})$. Этап нахождения соотношений занял около 19 000 mips-years. Для решения систем линейных уравнений было использовано структурированное гауссово исключение, и затем для уплотненной системы был применен алгоритм Видемана (см. об этом алгоритме гл. 11). На решение системы линейных уравнений ушло более двух месяцев. Автор [269] также делает вывод о том, что использованный им метод не позволяет осуществлять в настоящее время дискретное логарифмирование в $GF(2^n)$ для $n \geqslant 997$.

О дискретном логарифмировании в поле $GF(p^n)$ см. также [173; 32].

§ 5.5. Дискретное логарифмирование и решето числового поля

В данном параграфе мы расскажем об алгоритмах решета числового поля для дискретного логарифмирования по простому модулю. Алгоритмы решета числового поля для факторизации появились несколько раньше (см. об этом § 3.6). Основываясь на идеях этих алгоритмов, Д. Гор-

дон в 1993 г. в работе [127] предложил алгоритм решения уравнения

$$a^x \equiv b \pmod{p}, \tag{5.5}$$

где p — простое число; сложность алгоритма составляет эвристически $L_p\left[\frac{1}{3}; 3^{2/3}\right]$ арифметических операций.

Метод Гордона оказался непрактичным. Широкауер в работе [236] предложил свою версию алгоритма решета числового поля для решения (5.5) со сложностью $L_p\left[\frac{1}{3}; (64/9)^{1/3}\right]$ арифметических операций. Его алгоритм был реализован Вебером, и в работе [274] описано логарифмирование по модулю $p \asymp 10^{40}$, а в работе [238] 1996 г. — по модулю $p \asymp 10^{65}$. Также Вебер в работе [275] провел некоторые предварительные вычисления, связанные с дискретным логарифмированием по предложенному МакКарли модулю $p \asymp 10^{129}$; позднее в [278] было найдено решение задачи (5.5) для данного p. Однако это число p имеет специальный вид, и поэтому используется специальное решето числового поля, работающее быстрее. Вебер в 1997 г. в своей диссертации [276] провел логарифмирование по модулю $p \asymp 10^{85}$, не имеющему специального вида, как решетом числового поля (метод Широкауера), так и с помощью алгоритма Копперсмита—Одлыжко—Шреппеля (версия с гауссовыми целыми). При этом оказалось, что алгоритм решета числового поля работает медленнее для данного p (см. [277; 276]), чем алгоритм Копперсмита—Одлыжко—Шреппеля. Позднее Жу и Лерсье показали, что для $p > 10^{100}$ алгоритмы решета числового поля работают быстрее, чем алгоритм Копперсмита—Одлыжко—Шреппеля, см. [134]. В январе 2001 г. Жу и Лерсье провели логарифмирование по модулю $p \asymp 10^{110}$ (см. [135]), а в апреле 2001 г. — по модулю $p \asymp 10^{120}$ (см. [136]). В настоящее время это — рекордное значение p для дискретного логарифмирования по простому модулю, не имеющему специального вида.

Прежде чем описать схему алгоритмов решета числового поля для решения (5.5), скажем еще несколько слов о других результатах в этой области.

Широкауер в работе [237] обобщил алгоритм работы [236] на случай поля $GF(p^n)$. При $q = p^n$ оценка сложности составляет $L_q[1/3; (64/9)^{1/3}]$ арифметических операций. Фактически это верно при фиксированном p и $n \to \infty$. Согласно [237], указанная оценка справедлива и при $\log p > n^{2+\varepsilon}$ для некоторого $\varepsilon > 0$. Семаев в работе [247] также получил ряд результатов о дискретном логарифмировании в конечных непростых полях.

В работе [33] идеи Копперсмита и Широкауера применены для получения алгоритма дискретного логарифмирования в простом поле $GF(p)$ с наилучшей известной оценкой сложности $L_p\left[\frac{1}{3}; c\right]$, где $c = (92 + 26\sqrt{13})^{1/3}/3 \approx 1{,}902$.

Для чисел специального вида в работе [127] была предложена версия алгоритма решета числового поля со сложностью $L_p\left[\frac{2}{5}; c\right]$, где $c \approx 1{,}00475$. За счет того, что c близка к 1, такой алгоритм в некоторых случаях работает быстрее алгоритмов со сложностью $L_p\left[\frac{1}{3}; c\right]$, как показывает пример работы [278].

См. также [208; 152].

Теперь мы опишем, следуя [276], общую схему алгоритмов решета числового поля для решения задачи (5.5). Мы опишем также, в виде некоторого частного случая этой схемы, версию алгоритма Копперсмита—Одлыжко—Шреппеля с гауссовыми целыми. Принято рассматривать алгоритм работы [97] как отдельный метод, отличный от решета числового поля, поскольку он был придуман раньше и имеет собственную оценку сложности.

Далее в этом параграфе мы снова, как и в § 3.6, предполагаем, что читатель знаком с алгебраической теорией чисел в объеме книги [263].

Схема алгоритма.

1 этап. На этом этапе мы сводим решение уравнения (5.5) к решению уравнений

$$a^x \equiv s \pmod{p}, \qquad s \in S,$$

где S — некоторое конечное множество достаточно малых натуральных чисел. Грубо говоря, мы ищем одно $z \in \mathbb{N}$, такое, что

$$a^z \cdot b \equiv \prod_j s_j \pmod{p},$$

где s_j — не очень большие простые числа, скажем, $s_j \leqslant L_p\left[\frac{2}{3}; \text{const}\right]$. Факторизацию $a^z \cdot b \pmod{p}$ мы можем проводить методом эллиптических кривых Ленстры (см. гл. 4). Тогда $S = \{s_j\}$, $\log_a b \equiv -z + \sum_j \log_a s_j \pmod{p-1}$.

2 этап. С помощью некоторой техники мы выбираем два многочлена $g_1(x), g_2(x) \in \mathbb{Z}[x]$, $\deg g_i(x) = n_i$, $i = 1, 2$, имеющих общий корень

$m \pmod p$. Мы обозначаем для $j = 1, 2$:

$\alpha_j \in \mathbb{C}$ — фиксированный корень $g_j(x)$,

$h_j \in \mathbb{N}$ — старший коэффициент $g_j(x)$,

$\mathbb{K}_j \equiv \mathbb{Q}(\alpha_j)$,

$\mathcal{O}_j = \mathbb{Z}_{\mathbb{K}_j}$ — кольцо целых алгебраических чисел поля $\mathbb{K}_j$.

Замечание 5.7. Версия алгоритма Копперсмита—Одлыжко—Шреппеля с гауссовыми целыми в рамках данной схемы выглядит следующим образом:

1) $n_1 = 2$; $g_1(x)$ — неприводимый многочлен второй степени; поле $\mathbb{K}_1$ есть мнимое квадратичное одноклассное поле;

2) $n_2 = 1$, $g_2(x)$ — линейный многочлен вида $Ux + V$, U, $V \in \mathbb{Z}$.

3 этап. (Выбор факторной базы.) Для $j = 1, 2$ мы находим факторные базы

$$F_j = \{\wp \mid \wp \text{ — простые идеалы } \mathcal{O}_j, \operatorname{Norm} \wp < B_j\} \cup \{h_j\}.$$

Здесь B_j — некоторые постоянные, субэкспоненциально зависящие от p.

4 этап. С помощью некоторого просеивания мы находим множество пар $C = \{(c, d)\} \in \mathbb{Z}^2$ такое, что для $i = 1, 2$ идеалы $(h_i(c + d\alpha_i))$ в кольце $\mathcal{O}_j$ гладки по отношению к факторной базе F_j. При этом множество C должно быть достаточно велико, $|C| > |F_1| + |F_2|$.

5 этап. Для каждого $s \in S$ мы находим специальные соотношения. Для каждого простого идеала $\wp_s \in \mathcal{O}_1$, лежащего над s, мы находим пару чисел c, d такую, что идеал $(h_1(c + \alpha d))/\wp_1$ гладок по отношению к F_1 и идеал $(h_2(c + d\alpha_2))$ гладок по отношению к F_2.

6 этап. Для каждого большого простого числа q, делящего $p - 1$ (мы считаем, что факторизация $p - 1$ нам известна), делаем следующее.

1. Вычисляем так называемые аддитивные характеры Широкауера (определение см. ниже) от элементов $h_j(c + d\alpha_j)$, $j = 1, 2$, $(c, d) \in C$.

2. Находим матрицу A с элементами из поля $\mathbb{Z}/q\mathbb{Z}$. Ее столбцы состоят из векторов показателей в разложении $h_j(c + d\alpha_j)$ на простые идеалы и из значений аддитивных характеров.

3. Путем решения системы линейных уравнений $AX \equiv 0 \pmod q$ находим элементы $\gamma_i \in \mathcal{O}_i$, $i = 1, 2$, такие, что $\gamma_i = \delta_i^q$, $\delta_i \in \mathcal{O}_i$, $i = 1, 2$.

4. С помощью кольцевых эндоморфизмов

$$\varphi_j \colon \mathbb{Z}[h_j\alpha_j] \to \mathbb{Z}/p\mathbb{Z}, \quad \varphi_j(h_j\alpha_j) \equiv h_j m \pmod p, \quad j = 1, 2,$$

мы переходим от q-х степеней в кольцах $\mathcal{O}_j$ к целым числам и находим k, $l \in \mathbb{Z}$ такие, что $a^k \cdot b^l \equiv d^q \pmod p$. Из этого следует, что

$k + lx \equiv 0 \pmod{q}$, где $x \pmod{p-1}$ — решение (5.5). Отсюда мы находим значение $x \pmod{q}$.

Замечание 5.8. Для версии [97] с гауссовыми целыми нам не нужно вычислять аддитивные характеры Широкауера, поскольку мы работаем в кольцах $\mathcal{O}_i$ с однозначным разложением на множители и с конечной группой единиц.

7 этап. На 6 этапе мы нашли значение $x \pmod{q}$ для больших простых делителей q числа $p-1$. Предположим, что $p-1$ не делится на квадрат большого простого числа. Тогда недостающие значения $x \pmod{q^{\alpha_q}}$, где q — небольшие простые числа, $q^{\alpha_q} \| p-1$, мы найдем с помощью алгоритма Полига—Хеллмана. Затем с помощью китайской теоремы об остатках мы найдем искомое значение $x \pmod{p-1}$.

Конец алгоритма.

Определим теперь аддитивные характеры Широкауера, возникающие на 6 этапе алгоритма. Пусть α — алгебраическое число, $\deg \alpha = n$, $f(x) = a_n x^n + \ldots + a_0 \in \mathbb{Z}[x]$ — минимальный многочлен для α, $\mathbb{K} = \mathbb{Q}(\alpha)$, $\mathcal{O} = \mathbb{Z}_{\mathbb{K}}$ — кольцо целых алгебраических чисел поля $\mathbb{K}$. Положим

$$\Gamma = \{\gamma \in \mathcal{O} \mid l \nmid \mathrm{Norm}_{\mathbb{K}/\mathbb{Q}}(\gamma)\},$$

$$\mathcal{E} = \mathrm{HOK}\big\{|(\mathcal{O}/\mathfrak{b})^*| \;\big|\; \mathfrak{b} \text{ — простые идеалы } \mathcal{O},\ \mathfrak{b}|(l)\big\}.$$

Очевидно, что Γ является подгруппой относительно умножения. Рассмотрим отображение

$$\lambda\colon (\Gamma, \cdot) \to l\mathcal{O}/l^2\mathcal{O}, \qquad \lambda(\gamma) = \gamma^{\mathcal{E}} - 1 \pmod{l^2\mathcal{O}}.$$

Нетрудно видеть, что λ является гомоморфизмом мультипликативной полугруппы в аддитивную группу $l\mathcal{O}/l^2\mathcal{O}$. Пусть

$$\mathcal{O} = \mathbb{Z}\omega_1 \oplus \ldots \oplus \mathbb{Z}\omega_n,$$

где $\omega_1, \ldots, \omega_n$ — целый базис кольца $\mathcal{O}$. Тогда $l\mathcal{O}/l^2\mathcal{O}$ является линейным пространством над $\mathbb{Z}/l\mathbb{Z}$ с базисом $l\omega_1 \pmod{l^2\mathcal{O}}, \ldots$ $\ldots, l\omega_n \pmod{l^2\mathcal{O}}$. Пусть $\Omega_i \equiv l\omega_i \pmod{l^2\mathcal{O}}$, и пусть $\lambda_i(\gamma) = \sum_{i=1}^{n} b_i \Omega_i$, где $b_i \in \mathbb{Z}/l\mathbb{Z}$. Рассмотрим отображения $\lambda_i\colon \Gamma \to \mathbb{Z}/l\mathbb{Z}$, $\lambda_i(\gamma) \equiv b_i \pmod{l}$. Эти отображения называются аддитивными характерами Широкауера, причем λ однозначно определяется набором $\lambda_1, \ldots, \lambda_n$.

В работе [236] показано, что при некоторых условиях из равенства $\lambda(\gamma) = 0$ следует, что $\gamma = \delta^l$, где $\delta \in \mathcal{O}$. Это обеспечивает нам нахождение элементов $\gamma_i = \delta_i^q$ на 6 этапе алгоритма.

Теперь вкратце опишем просеивание, применяемое на 4 этапе алгоритма. Условие гладкости идеала $(h_i(c + d\alpha_i))$ в кольце $\mathcal{O}_i$ равносильно

гладкости нормы $\mathrm{Norm}_{\mathbb{K}/Q}(h_i(c + d\alpha_i))$ в кольце целых чисел. Данная норма представляет собой некоторый однородный многочлен $f_i(c, d)$, где $f_i(X, Y) \in \mathbb{Z}[X, Y]$. Предположим, что мы хотим найти те значения c, d, для которых $f_i(c, d)$ гладко по отношению к некоторой границе гладкости B. Если мы фиксируем d, простое число q и натуральное число h, то значения c, такие, что $q^h | f(c, d)$, будут лежать в арифметической прогрессии

$$c \equiv d \cdot r_j \pmod{q^h},$$

где $r_j \pmod{q^h}$ — какой-либо из корней уравнения $f(Z, 1) \equiv 0 \pmod{q^h}$. Теперь ясно, что мы можем организовать просеивание аналогично алгоритмам, описанным в §3.4, 3.6. Этот вид просеивания называется линейным просеиванием. Существует также просеивание по векторам (тогда мы движемся не по арифметической прогрессии, а по некоторой решетке в $\mathbb{Z}^2$) и решетчатое просеивание, см. [107].

На этом мы закончим описание алгоритмов решета числового поля для дискретного логарифмирования. Подробное описание этих алгоритмов может, пожалуй, составить предмет для написания отдельной монографии.

Замечание 5.9. При помощи некоторого развития методов решета числового поля в работе [45] получен алгоритм дискретного логарифмирования в конечных полях $GF(p^n)$ со сложностью $L_{p^n}\left[\frac{1}{3}; \mathrm{const}\right]$ арифметических операций. Однако эта оценка сложности справедлива не для всех значений p^n, см. [209]. Точнее, должно выполняться неравенство $\log p < n^{1/2}$.

§ 5.6. Частное Ферма и дискретное логарифмирование по составному модулю

В данном параграфе мы описываем некоторые методы проверки разрешимости и решения задачи дискретного логарифмирования в кольцах вычетов $\mathbb{Z}/m\mathbb{Z}$ по составному модулю m, а также в кольцах вида $GF(p)[x]/(f(x))$, где $f(x) \in GF(p)[x]$ — многочлен, не являющийся неприводимым. Ряд результатов в этом направлении был получен в работах [234; 15; 14].

Определение 5.10. Пусть $r \in \mathbb{N}$, $r = 2^{\alpha_0} p_1^{\alpha_1} \dots p_t^{\alpha_t}$, есть разложение r на простые множители, $2 < p_1 < \dots < p_t$. Определим функцию Кармайкла $\lambda(r)$:

$$\lambda(r) = \mathrm{HOK}(\varphi_0(2^{\alpha_0}), \varphi(p_1^{\alpha_1}), \dots, \varphi(p_t^{\alpha_t})),$$

где φ — функция Эйлера, $\varphi_0(1)=\varphi_0(2)=1$, $\varphi_0(4)=2$, $\varphi_0(2^{\alpha_0})=2^{\alpha_0-2}$ при $\alpha_0\geqslant 3$.

Нетрудно видеть, что при $a\in\mathbb{Z}$, $(a,\,r)=1$, выполнено сравнение $a^{\lambda(r)}\equiv 1 \pmod r$.

Определение 5.11. Пусть $r\in\mathbb{Z}_{>1}$, $a\in\mathbb{Z}$, $(a,\,r)=1$. *Частное Ферма* $Q(a,\,r)$ определяется соотношением

$$Q(a,\,r)\equiv\frac{a^{\lambda(r)}-1}{r} \pmod r$$

$\left(\text{здесь } \dfrac{a^{\lambda(r)}-1}{r} \text{ обозначает результат деления } a^{\lambda(r)}-1 \text{ на } r \text{ в кольце } \mathbb{Z}\right)$.

Опишем некоторые свойства частного Ферма; более подробно см. [234; 114; 80; 170].

Лемма 5.12. *Пусть* $a,\,b\in\mathbb{Z}$, $(a,\,r)=(b,\,r)=1$. *Тогда*

$$Q(ab,\,r)=Q(a,\,r)+Q(b,\,r) \pmod r.$$

Доказательство. Поскольку

$$a^{\lambda(r)}\equiv 1+rQ(a,\,r) \pmod{r^2},\qquad b^{\lambda(r)}\equiv 1+rQ(b,\,r) \pmod{r^2},$$

то

$$(ab)^{\lambda(r)}\equiv 1+r(Q(a,\,r)+Q(b,\,r)) \pmod{r^2}.$$

Из этого сравнения следует утверждение леммы. $\square$

Замечание 5.13. Мы показали, что частное Ферма обладает тем же свойством, что и логарифм: оно переводит произведение в сумму.

Определение 5.14. Рассмотрим $Q(r,\,x)$ как функцию на множестве $x\in\mathbb{Z}$, $(x,\,r)=1$. Назовем $m\in\mathbb{Z}$, $m\neq 0$, *периодом* $Q(r,\,x)$, если

1) m делится на все простые числа, делящие r;

2) $Q(a+m,\,r)\equiv Q(a,\,r) \pmod r$ для всех $a\in\mathbb{Z}$, $(a,\,r)=1$.

Лемма 5.15. *Число* $R=\dfrac{r^2}{(\lambda(r),\,r)}$ *является периодом* $Q(r,\,x)$.

Доказательство. Поскольку $R=r\cdot\dfrac{r}{(\lambda(r),\,r)}$, то $r\mid R$; следовательно, первое условие в определении периода выполнено. Далее, поскольку $\dfrac{R^2}{r}\equiv 0 \pmod r$, то

$$Q(a+R,\,r)\equiv\frac{(a+R)^{\lambda(r)}-1}{r} \pmod r\equiv\frac{a^{\lambda(r)}-1}{r}+\lambda(r)a^{\lambda(r)-1}\cdot\frac{R}{r} \pmod r.$$

Далее,

$$\frac{\lambda(r)R}{r}=\frac{\lambda(r)r^2}{r(\lambda(r),\,r)}\equiv 0 \pmod r.$$

Поэтому $Q(a+R,\,r)\equiv Q(a,\,r) \pmod r$, что и требовалось доказать. $\square$

Замечание 5.16. Если m — период $Q(x, r)$, то частное Ферма является гомоморфизмом мультипликативной группы $(\mathbb{Z}/m\mathbb{Z})^*$ в аддитивную группу $\mathbb{Z}/r\mathbb{Z}$.

Рассмотрим теперь задачу дискретного логарифмирования $a^x \equiv b \pmod m$, где m — период для частного Ферма $Q(x, r)$. Из равенств $(a, m) = (b, m) = 1$ следует, что $(a, r) = (b, r) = 1$. Поэтому в силу периодичности справедливо соотношение

$$Q(b, r) \equiv Q(a^x, r) \pmod r \equiv xQ(a, r) \pmod r.$$

Если выполняется условие $(Q(a, r), r) = 1$, то

$$x \equiv Q(b, r)Q(a, r)^{-1} \pmod r.$$

Таким образом, мы найдем $x \pmod r$; нам же надо найти x по модулю порядка элемента $a \pmod m$ в $(\mathbb{Z}/m\mathbb{Z})^*$. В некоторых случаях это возможно; они описаны в работе [234]. Эти случаи связаны с так называемыми λ-низкими числами. О λ-низких числах см. также [28].

Пример 5.17. $m = 1600 = 2^6 5^2$, $a = 3$. Рассмотрим уравнение

$$3^x \equiv b \pmod{1600}.$$

Положим $r = 80 = 2^4 5$; $\lambda(r) = 4$, $R = \dfrac{r^2}{(\lambda(r), r)} = \dfrac{6400}{4} = 1600 = m$. По лемме 5.15 число m является периодом $Q(x, r)$. Далее,

$$Q(3, r) \equiv \frac{3^4 - 1}{80} \pmod{80} \equiv 1 \pmod r.$$

Поэтому в силу периодичности

$$Q(b, r) \equiv Q(3^x, r) \equiv xQ(3, r) \equiv x \pmod{80} \equiv x \pmod{\lambda(m)},$$

поскольку $\lambda(m) = 80$. Так как порядок $3 \pmod m$ в $(\mathbb{Z}/m\mathbb{Z})^*$ делит $\lambda(m)$, то решение уравнения $3^x \equiv b \pmod{1600}$ задается формулой

$$x \equiv \frac{b^4 - 1}{80} \pmod{80}.$$

Для найденного по этой формуле x следует сделать проверку, так как не для каждого $b \in (\mathbb{Z}/m\mathbb{Z})^*$ уравнение $3^x \equiv b \pmod m$ разрешимо. В этом примере мы пользуемся тем, что для данных m и r выполнено условие $\lambda(m) \mid r$; это условие выполняется далеко не всегда.

С помощью частного Ферма можно проводить подъем решения уравнения $a^x \equiv b \pmod{p^\alpha}$, и находить решение уравнения $a^x \equiv b \pmod{p^{\alpha+1}}$.

Теорема 5.18. *Пусть p — нечетное простое число, $\alpha \in \mathbb{Z}_{\geqslant 2}$, $m = p^\alpha$. Пусть $g \in \mathbb{Z}$, $g \pmod m$ — первообразный корень по модулю m, $b \in \mathbb{Z}$, $(b, p) = 1$. Обозначим $x = [\log b]_\alpha \in \mathbb{Z}/\varphi(p^\alpha)\mathbb{Z}$ — решение уравнения $g^x \equiv b \pmod m$. Тогда $[\log b]_\alpha$ есть единственное по модулю $\varphi(p^\alpha)$ решение системы уравнений*

$$\begin{cases} Q(g, p^{\alpha-1})x \equiv Q(b, p^{\alpha-1}) \pmod{p^{\alpha-1}}, \\ x = [\log b]_1 \pmod{p-1}, \end{cases}$$

где $[\log b]_1$ означает решение уравнения $g^y \equiv b \pmod p$.

Следствие 5.19. *Из теоремы следует, что дискретное логарифмирование в группе $(\mathbb{Z}/p^\alpha\mathbb{Z})^*$ при $p > 2$ сводится к дискретному логарифмированию в $(\mathbb{Z}/p\mathbb{Z})^*$.*

Доказательство теоремы. Обозначим $r = p^{\alpha-1}$. Тогда

$$R = \frac{r^2}{(\lambda(r),\, r)} = \frac{p^{2\alpha-2}}{(p^{\alpha-2}(p-1),\, p^{\alpha-1})} = p^\alpha = m.$$

По лемме 5.15 число m является периодом $Q(x, r)$. Поэтому из $g^x \equiv b \pmod m$ следует, что $xQ(g, r) \equiv Q(b, r) \pmod r$; т. е. выполнено первое уравнение системы. Также из $g^x \equiv b \pmod m$ следует, что $g^x \equiv b \pmod p$. Это означает, что выполнено и второе уравнение.

Для доказательства единственности достаточно показать, что $Q(g, p^{\alpha-1}) \not\equiv 0 \pmod p$. Поскольку $g \pmod{p^\alpha}$ является первообразным корнем, то $g^{p-1} \not\equiv 1 \pmod{p^2}$. Отсюда по индукции получаем, что для $u = 1, 2, 3, \ldots$ выполнено равенство

$$g^{(p-1)p^u} = 1 + p^{u+1}A_u,$$

где $A_u \in \mathbb{Z}$, $A_u \not\equiv 0 \pmod p$. Положим $u = \alpha - 2$; тогда число $g^{\varphi(p^{\alpha-1})} - 1$ делится на $p^{\alpha-1}$ и не делится на p^α. Так как $\lambda(p^{\alpha-1}) = \varphi(p^{\alpha-1})$, то из равенства

$$Q(g, p^{\alpha-1}) \equiv \frac{g^{\lambda(p^{\alpha-1})} - 1}{p^{\alpha-1}} \pmod{p^{\alpha-1}}$$

следует, что $Q(g, p^{\alpha-1}) \not\equiv 0 \pmod p$. □

Теорема 5.20. *Пусть $m = 2^\alpha$, где $\alpha \in \mathbb{Z}_{\geqslant 5}$. Пусть $b \in \mathbb{Z}$, b нечетно,*

$$b \equiv (-1)^{k_0} 5^{k_1} \pmod{2^\alpha},$$

где $k_0 = 0$ при $b \equiv 1 \pmod 4$, $k_0 = 1$ при $b \equiv 3 \pmod 4$, и $0 \leqslant k_1 \leqslant 2^{\alpha-2} - 1$. Положим $[\log b]_\alpha = k_1$. Тогда $[\log b]_\alpha$ есть единственное

по модулю $2^{\alpha-2}$ *решение уравнения*

$$xQ(5,\,2^{\alpha-2}) \equiv Q(b,\,2^{\alpha-2}) \pmod{2^{\alpha-2}}.$$

Следствие 5.21. *Из теоремы* 5.20 *следует, что дискретное логарифмирование в группе* $\mathbb{Z}/2^{\alpha}\mathbb{Z}^*$ *при* $\alpha \geqslant 5$ *сводится к вычислению* $Q(b,\,2^{\alpha-2})Q(5,\,2^{\alpha-2})^{-1} \pmod{2^{\alpha-2}}$.

Доказательство теоремы. Положим $r = 2^{\alpha-2}$. Тогда

$$R = \frac{r^2}{(\lambda(r),\,r)} = \frac{2^{2\alpha-4}}{(2^{\alpha-4},\,2^{\alpha-2})} = 2^{\alpha} = m$$

является периодом $Q(x,\,r)$. Если $b \equiv (-1)^{k_0}5^{k_1} \pmod{2^{\alpha}}$, то

$$Q(b,\,r) \equiv Q((-1)^{k_0}5^{k_1},\,r) \equiv \frac{((-1)^{k_0}5^{k_1})^{\lambda(r)}-1}{r} \equiv \frac{5^{k_1\lambda(r)}-1}{r} \pmod{r},$$

поскольку $\lambda(r)$ четно в силу условия $\alpha \geqslant 5$. Значит,

$$Q(b,\,r) \equiv Q(5^{k_1},\,r) \equiv k_1 Q(5,\,r) \pmod{r}.$$

Для доказательства единственности достаточно проверить, что $Q(5,\,r)$ нечетно. Поскольку

$$Q(5,\,2^{\alpha-2}) \equiv \frac{5^{\lambda(2^{\alpha-2})}-1}{2^{\alpha-2}} \equiv \frac{5^{2^{\alpha-4}}-1}{2^{\alpha-2}} \pmod{2^{\alpha-2}},$$

то достаточно показать, что $5^{2^{\alpha-4}} \not\equiv 1 \pmod{2^{\alpha-1}}$ при $\alpha \geqslant 5$; это легко проверяется индукцией по α. Теорема 5.20 доказана. □

Рассмотрим теперь уравнение

$$a^x \equiv b \pmod{s}, \qquad (a,\,s) = (b,\,s) = 1, \tag{5.6}$$

где $s \in \mathbb{N}$, $s > 1$, s — нечетно, и известно полное разложение s на простые множители q_i:

$$s = \prod_{i=1}^{k} q_i^{u_i}, \qquad k \geqslant 2. \tag{5.7}$$

Пусть также известно полное разложение $q_i - 1$ на простые множители:

$$q_i - 1 = \prod_{j=1}^{v_i} p_{ij}^{\alpha_{ij}}, \qquad i = 1,\,\ldots,\,k. \tag{5.8}$$

Мы будем изучать вопрос о проверке разрешимости уравнения дискретного логарифмирования (5.6).

Обсудим сначала, как можно решать уравнение (5.6). Пусть g_i — первообразные корни по модулям $q_i^{u_i}$, $i = 1,\,\ldots,\,k$; их можно найти,

зная (5.8). Пусть c_i при $i = 1, \ldots, k$ удовлетворяют сравнениям

$$\begin{cases} c_i \equiv g_i \pmod{q_i^{u_i}}, \\ c_i \equiv 1 \pmod{q_j^{u_j}} \qquad \text{при } j \neq i. \end{cases}$$

Тогда $c_i \pmod s$ имеют порядки $\varphi(q_i^{u_i}) = M_i$, и

$$(\mathbb{Z}/s\mathbb{Z})^* = \langle c_1 \pmod s \rangle_{M_1} \times \ldots \times \langle c_k \pmod s \rangle_{M_k}$$

есть разложение $(\mathbb{Z}/s\mathbb{Z})^*$ в прямое произведение циклических групп. Пусть

$$\begin{cases} a \equiv c_1^{A_1} \ldots c_k^{A_k} \pmod s, \\ b \equiv c_1^{B_1} \ldots c_k^{B_k} \pmod s. \end{cases} \tag{5.9}$$

Числа A_i, $0 \leqslant A_i \leqslant M_i - 1$, мы находим, решая уравнения $g_i^y \equiv a \pmod{q_i^{u_i}}$ (аналогично находим B_i). Решение таких уравнений с помощью теоремы 5.18 сводится к решению уравнений $g_i^z \equiv a \pmod{q_i}$. То есть для нахождения A_i, B_i нам нужно уметь решать задачу дискретного логарифмирования по простому модулю. В предыдущих параграфах мы видели, что это трудная задача, если модуль велик.

Допустим, что нам все же известны числа A_i, B_i в (5.9). Тогда из (5.6) и (5.9) следует, что

$$g_i^{A_i x} \equiv g_i^{B_i} \pmod{q_i^{u_i}}, \quad i = 1, \ldots, k.$$

Отсюда

$$A_i x \equiv B_i \pmod{\varphi(q_i^{u_i})}, \quad i = 1, \ldots, k. \tag{5.10}$$

Для того, чтобы система уравнений (5.10) имела решение, необходимо, чтобы числа

$$D_i = \text{НОД}(A_i, \varphi(q_i^{u_i})) \tag{5.11}$$

делили B_i. Если это выполняется, то из (5.10) следует, что

$$x \equiv (A_i/D_i)^{-1}(B_i/D_i) \pmod{\varphi(q_i^{u_i})/D_i}, \quad i = 1, \ldots, k. \tag{5.12}$$

В свою очередь, для разрешимости системы (5.12) необходимо и достаточно, чтобы при всех $i \neq j$ правые части i-го и j-го уравнений из (5.12) были сравнимы по модулю $\text{НОД}(\varphi(q_i^{u_i})/D_i, \varphi(q_j^{u_j})/D_j)$. В этом случае мы найдем решение системы (5.12); обозначим его

$$x_0 \pmod{\text{НОК}(\varphi(q_1^{u_1})/D_1, \ldots, \varphi(q_k^{u_k})/D_k)}. \tag{5.13}$$

Лемма 5.22. *Для определенных по формулам* (5.11) *чисел* D_i *выполнено равенство*

$$\operatorname{ord} a \pmod s = \operatorname{НОК}(\varphi(q_1^{u_1})/D_1, \ldots, \varphi(q_k^{u_k})/D_k),$$

где $\operatorname{ord} a \pmod s$ *обозначает порядок элемента* $a \pmod s \in (\mathbb{Z}/s\mathbb{Z})^*$

Доказательство. Поскольку

$$\operatorname{ord} a \pmod s = \operatorname{НОК}_{i=1,\ldots,k}(\operatorname{ord}(a \pmod{q_i^{u_i}})),$$

то достаточно доказать, что

$$\operatorname{ord}(a \pmod{q_i^{u_i}}) = \varphi(q_i^{u_i})/D_i, \quad i = 1, \ldots, k.$$

По определению чисел A_i справедливо равенство $a \equiv g_i^{A_i} \pmod{q_i^{u_i}}$. Пусть p — простое число, $\alpha = \nu_p(\varphi(q_i^{u_i})) \geqslant 1$. Если $\nu_p(A_i) \geqslant \alpha$, то $\nu_p(\operatorname{ord}(a \pmod{q_i^{u_i}})) = 0$ и $\nu_p(D_i) = \alpha$. Если же $\beta = \nu_p(A_i)$ и $0 \leqslant \beta < \alpha$, то очевидно, что $\nu_p(\operatorname{ord}(a \pmod{q_i^{u_i}})) = \alpha - \beta$, и $\nu_p(D_i) = \beta$. Во всех случаях

$$\nu_p(\operatorname{ord}(a \pmod{q_i^{u_i}})) = \nu_p(\varphi(q_i^{u_i})/D_i).$$

Лемма 5.22 доказана. □

Замечание 5.23. Числа $D_i = \operatorname{НОД}(A_i, \varphi(q_i^{u_i}))$ нетрудно найти, зная $\operatorname{ord}(a \pmod{q_i^{u_i}})$, даже если мы не знаем A_i. Это следует из доказательства леммы 5.22. Действительно, пусть p и α те же, что в доказательстве леммы 5.22, $E_i = \operatorname{ord}(a \pmod{q_i^{u_i}})$. Если $p \nmid E_i$, то $p^\alpha \mid A_i$, откуда $\nu_p(D_i) = \alpha$. Если же $p^e \parallel E_i$, где $1 \leqslant e \leqslant \nu_p(\varphi(q_i^{u_i}))$, то $\nu_p(A_i) = \nu_p(\varphi(q_i^{u_i})) - e$; поэтому $\nu_p(D_i) = \nu_p(\varphi(q_i^{u_i})) - e$. Заметим также, что E_i можно вычислить, зная разложение (5.8); алгоритм нахождения порядка элемента см. в § 1.5.

Замечание 5.24. Из леммы 5.22 вытекает, что (5.13) есть искомое решение уравнения (5.6) в случае, когда система уравнений (5.12) разрешима.

Мы показали, как можно решить уравнение (5.6), зная числа $A_1, \ldots, A_k, B_1, \ldots, B_k$. Далее мы докажем две теоремы, дающие необходимые и достаточные условия для разрешимости (5.6). В этих теоремах мы не предполагаем, что A_i и B_i нам известны.

Лемма 5.25. *Пусть* q *— нечетное простое число,* $q - 1 = \prod_{j=1}^{n} p_j^{\alpha_j}$ *— разложение* $q - 1$ *на простые множители. Пусть* $u \in \mathbb{N}$, $a, b \in (\mathbb{Z}/q^u\mathbb{Z})^*$, $a \not\equiv 1 \pmod{q^u}$. *Пусть также* g *— первообразный корень по модулю* q^u. *Тогда выполнены следующие утверждения.*

1) *Если*

$$\operatorname{ord}(a(\operatorname{mod} q^u)) = q^{u-1-\beta_0}\prod_{j=1}^{n} p_j^{\alpha_j-\beta_j}, \quad \text{где } \beta_j \in \mathbb{Z}_{\geqslant 0},$$

то

$$a \equiv g^{q^{\beta_0}\prod\limits_{j=1}^{n} p_j^{\beta_j}\cdot l} \pmod{q^u},$$

где $0 < l < \varphi(q^u)$. *При этом, если* $u > 1$ *и* $\beta_0 < u-1$ *или если* $u = 1$, *то* $q \nmid l$; *если* $j > 0$ *и* $\beta_j < \alpha_j$, *то* $p_j \nmid l$.

2) *Сравнение* $a^x \equiv b \pmod{q^u}$ *разрешимо тогда и только тогда, когда* $\operatorname{ord}(b(\operatorname{mod} q^u))$ *делит* $\operatorname{ord}(a(\operatorname{mod} q^u))$.

Доказательство. Первое утверждение леммы очевидно, а второе вытекает из цикличности группы $(\mathbb{Z}/q^u\mathbb{Z})^*$. □

Вернемся к вопросу о разрешимости (5.6), используя обозначения (5.7), (5.8), (5.9).

Лемма 5.26. *Уравнение* (5.6) *разрешимо тогда и только тогда, когда разрешима система сравнений*

$$\begin{cases} A_i x \equiv B_i \pmod{q_i^{u_i-1}}, \\ A_i x \equiv B_i \pmod{p_{ij}^{\alpha_{ij}}}, \quad j = 1, \ldots, v_i, \quad i = 1, \ldots, k. \end{cases} \tag{5.14}$$

Доказательство. Сравнение $a^x \equiv b \pmod{s}$ выполнено тогда и только тогда, когда

$$c_i^{A_i x} \equiv c_i^{B_i} \pmod{q_i^{u_i}}, \quad i = 1, \ldots, k.$$

Для этого необходимо и достаточно, чтобы система $A_i x \equiv B_i \pmod{\varphi(q_i^{u_i})}$, $i = 1, \ldots, k$, была разрешима. Очевидно, что ее разрешимость эквивалентна разрешимости (5.14). □

Теорема 5.27. *Пусть в формуле* (5.8) $p_{i1} = 2, \alpha_{i1} = 1$ *для* $i = 1, \ldots, k$. *Пусть также все простые числа* q_i *и* p_{ij} *для* $j = 2, \ldots, v_i$, $i = 1, \ldots, k$, *различны. Предположим, что* $a \not\equiv 1 \pmod{q_i^{u_i}}$,

$$\operatorname{ord}(a \pmod{q_i^{u_i}}) = q_i^{u_i-1-\beta_{i0}}\prod_{j=1}^{v_i} p_{ij}^{\alpha_{ij}-\beta_{ij}}, \quad i = 1, \ldots, k,$$

причем $\beta_{i0} < u_i - 1$, *если* $u_i > 1$, *и* $\beta_{ij} < \alpha_{ij}$ *для всех* $j = 1, \ldots, v_i$, $i = 1, \ldots, k$. *Пусть также* $\operatorname{ord}(b(\operatorname{mod} q_i^{u_i})) \mid \operatorname{ord}(a(\operatorname{mod} q_i^{u_i}))$ *для* $i = 1, \ldots, k$. *Уравнение* (5.6) *разрешимо тогда и только тогда, когда*

$$b^{q_i^{u_i-1}(q_i-1)/2} \equiv F \pmod{q_i^{u_i}}, \quad i = 1, \ldots, k,$$

где $F = 1$ *или* $F = -1$.

Доказательство. Мы предположили, что $k \geqslant 2$ в (5.7). Тогда из (5.9) и п. 1 леммы 5.25 следует, что

$$A_i = q_i^{\beta_{i0}} \prod_{j=1}^{v_i} p_{ij}^{\beta_{ij}} \cdot L_i, \quad i = 1, \ldots, k,$$

где $L_i \in \mathbb{N}$, $(L_i, \varphi(q_i^{u_i})) = 1$. Заметим также, что если $u_i = 1$, то $\beta_{i0} = 0$; кроме того, $q_i \nmid L_i$. Пусть

$$\operatorname{ord}(b \pmod{q_i^{u_i}}) = q_i^{u_i - 1 - \gamma_{i0}} \prod_{j=1}^{v_i} p_{ij}^{\alpha_{ij} - \gamma_{ij}}, \quad i = 1, \ldots, k.$$

Тогда из условия теоремы следует, что $\gamma_{ij} \geqslant \beta_{ij}$ при всех $j = 1, \ldots, v_i$, $i = 1, \ldots, k$. Из леммы 5.25 получаем, что

$$B_i = q_i^{\gamma_{i0}} \prod_{j=1}^{v_i} p_{ij}^{\gamma_{ij}} N_i, \quad i = 1, \ldots, k,$$

где $N_i \in \mathbb{N}$. Применим лемму 5.26 и сведем разрешимость (5.6) к разрешимости системы (5.14). Если мы рассмотрим уравнение

$$A_i x \equiv B_i \pmod{q_i^{u_i - 1}}$$

из этой системы, то оно будет равносильно уравнению

$$\prod_{j=1}^{v_i} p_{ij}^{\beta_{ij}} \cdot L_i x \equiv q_i^{\gamma_{i0} - \beta_{i0}} \prod_{j=1}^{v_i} p_{ij}^{\gamma_{ij}} N_i \pmod{q_i^{u_i - 1 - \beta_{i0}}}. \tag{5.15}$$

Данное уравнение разрешимо; при $u_i = 1$ это очевидно, а при $u_i > 1$ разрешимость следует из того, что $q_i \nmid L_i$. Если мы рассмотрим уравнение

$$A_i x \equiv B_i \pmod{p_{ij}^{\alpha_{ij}}}, \quad j > 1, \tag{5.16}$$

из системы (5.14), то оно также будет разрешимо, поскольку $\nu_{p_{ij}}(A_i) = \beta_{ij} \leqslant \gamma_{ij} \leqslant \nu_{p_{ij}}(B_i)$. Поскольку модули в уравнениях (5.15) и (5.16) нечетны и различны по условию теоремы, то для разрешимости (5.14) необходимо и достаточно, чтобы была разрешима подсистема уравнений

$$A_i x \equiv B_i \pmod 2, \quad i = 1, \ldots, k. \tag{5.17}$$

Так как $\alpha_{i1} = 1$, то $\beta_{i1} = 0$ для $i = 1, \ldots, k$, и числа L_i нечетны. Поэтому $A_i \equiv 1 \pmod 2$, $i = 1, \ldots, k$. Следовательно, (5.17) примет вид

$x \equiv B_i \pmod 2$, $i = 1, \ldots, k$. Теперь для разрешимости (5.17) необходимо и достаточно, чтобы выполнялось условие

$$B_i \equiv E \pmod 2, \quad i = 1, \ldots, k,$$

где $E = 0$ или $E = 1$. Далее,

$$b^{q_i^{u_i-1}(q_i-1)/2} \equiv \left(g_i^{q_i^{u_i-1}(q_i-1)/2}\right)^{B_i} \equiv (-1)^{B_i} \pmod{q_i^{u_i}}.$$

Из последнего соотношения следует утверждение теоремы. □

Теорема 5.28. *Пусть $q_1, \ldots, q_k$ — различные нечетные простые числа,*

$$q_i - 1 = 2\prod_{j=2}^{v_i} p_{ij}^{\alpha_{ij}}, \quad i = 1, \ldots, k,$$

где p_{ij} — нечетные простые числа. Пусть числа p_{ij} при $j = 2, \ldots, v_i$, $i = 1, \ldots, k$, различны. Обозначим через $s = q_1 \ldots q_k$. Пусть $a, b \in \mathbb{N}$, $(a, s) = (b, s) = 1$, $t \in \mathbb{N}$, $1 \leqslant t \leqslant k$. Предположим, что $a \pmod{q_i}$ имеет порядок $q_i - 1$ для $i = 1, \ldots, t$ и порядок $(q_i - 1)/2$ для $i = t + 1, \ldots, k$. Уравнение (5.6) разрешимо тогда и только тогда, когда

$$\left(\frac{b}{q_1}\right) = \ldots = \left(\frac{b}{q_1}\right), \quad \left(\frac{b}{q_{t+1}}\right) = \ldots = \left(\frac{b}{q_k}\right) = 1,$$

где $\left(\frac{b}{q}\right)$ — символ Лежандра.

Доказательство. Из (5.9) и условия теоремы следует, что A_i взаимно просты с $\varphi(q_i) = q_i - 1$ для $i = 1, \ldots, t$, а для $i = t + 1, \ldots, k$ числа A_i четны и взаимно просты с $(q_i - 1)/2$. Из леммы 5.10 получаем систему

$$\begin{cases} A_i x \equiv B_i \pmod 2, \\ A_i x \equiv B_i \pmod{p_{ij}^{\alpha_{ij}}}, \end{cases}$$

где $j = 2, \ldots, v_i$, $i = 1, \ldots, k$. Эта система разрешима тогда и только тогда, когда разрешима подсистема

$$A_i x \equiv B_i \pmod 2, \quad i = 1, \ldots, k.$$

Эта подсистема имеет вид

$$\begin{cases} x \equiv B_i \pmod 2, & i = 1, \ldots, t, \\ B_i \equiv 0 \pmod 2, & i = t + 1, \ldots, k. \end{cases}$$

Заметим, что $B_i \equiv 0 \pmod 2$ тогда и только тогда, когда $\left(\frac{b}{q_i}\right) = 1$. Утверждение теоремы теперь очевидно. □

Замечание 5.29. В теоремах 5.27 и 5.28 наибольшим общим делителем чисел $q_i - 1$ и $q_j - 1$ является число 2. Это позволяет получить необходимые и достаточные условия для разрешимости (5.6). Если $q_i - 1$ и $q_j - 1$ имеют в качестве общего делителя небольшое простое число $p > 2$, то для проверки разрешимости (5.6) можно предложить алгоритм, аналогичный алгоритму Полига—Хеллмана. Мы предлагаем читателю описать этот алгоритм в качестве упражнения.

Зафиксируем простое число p и рассмотрим задачу дискретного логарифмирования $a^x = b$ в группе обратимых элементов $(\mathbb{Z}/p\mathbb{Z}[x]/(F(x)))^*$ факторкольца $\mathbb{Z}/p\mathbb{Z}[x]/(F(x))$, где многочлен $F(x) \in \mathbb{Z}/p\mathbb{Z}[x]$ — приводим.

Зафиксируем неприводимый многочлен $f = f(x) = x^n + A_{n-1}x^{n-1} + \ldots + A_0 \in \mathbb{Z}/p\mathbb{Z}[x]$ степени n, $n > 1$. Приведем несколько вспомогательных утверждений о свойствах колец $R_k = \mathbb{Z}/p\mathbb{Z}[x]/(f^k(x))$, $k = 1, 2, 3, \ldots$ Хорошо известно, что $R_1 = GF(p^n)$ и что существует $g(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $\deg g(x) < n$, такой, что $g(x) \pmod f$ имеет порядок $p^n - 1$ (см., например, [31, гл. 2]). Зафиксируем $g(x)$.

Лемма 5.30.

1) *Найдется многочлен* $g_1(x)$ (*равный либо* $g(x)$, *либо* $g(x) + f(x)$), *такой, что* $\deg g_1(x) \leqslant n$ *и* $g_1(x)^{p^n-1} \not\equiv 1 \pmod{f^2}$.

2) *Если* $h = h(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $f \nmid h$, *то при всех* $j \geqslant 0$

$$h^{p^j(p^n-1)} \equiv 1 \pmod{f^{p^j}}.$$

3) *Если* $j \geqslant 1$ $p^{j-1} < k \leqslant p^j$, *то порядок любого обратимого по умножению элемента кольца* R_n *делит* $p^j(p^n - 1)$, *и существует элемент такого порядка.*

Доказательство. Пусть $g(x)^{p^n-1} = 1 + f^l(x)t(x)$, где $t(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $f(x) \nmid t(x)$, $l \in \mathbb{N}$. Если $l = 1$, то $g_1(x) = g(x)$. Если $l \geqslant 2$, то

$$(g(x) + f(x))^{p^n-1} = g(x)^{p^n-1} + (p^n - 1)g(x)^{p^n-2}f(x) + \ldots =$$
$$= 1 - g(x)^{p^n-2}f(x) + t_1(x)f^2(x),$$

где $t_1(x) \in \mathbb{Z}/p\mathbb{Z}[x]$. Следовательно, $g_1(x) = g(x) + f(x)$ удовлетворяет первому утверждению леммы при $l \geqslant 2$. Далее, пусть $h(x)^{p^n-1} = 1 + f(x)t(x)$. Тогда $h(x)^{(p^n-1)p^j} = 1 + t(x)^{p^j}f(x)^{p^j}$, что доказывает второе утверждение. Легко видеть, что элемент $g_1(x) \pmod{f^k(x)}$, где $g_1(x)$ определен выше, имеет порядок $p^j(p^n - 1)$ в R_k; третье утверждение леммы теперь также очевидно. □

Лемма 5.31. *Пусть* $k > 1$. *Тогда* $|R_k| = p^{nk}$, $|R_k^*| = p^{n(k-1)}(p^n - 1)$. *Далее, при* $p^{j-1} < k \leqslant p^j$ *справедливо разложение* R_k^* *в прямое произведение циклических групп*

$$R_k^* = \langle \xi_{k,0} \rangle_{p^n-1} \times \langle \xi_{k,1} \rangle_{p^{l_{k,1}}} \times \ldots \times \langle \xi_{k,s_k} \rangle_{p^{l_{k,s_k}}},$$

где $l_{k,1} + \ldots + l_{k,s_k} = n(k-1)$ *и* $j = l_{k,1} \geqslant l_{k,2} \geqslant \ldots \geqslant l_{k,s_k}$. *В частности, при* $1 < k \leqslant p$ $s_k = n(k-1)$ *и*

$$R_k^* = \langle \xi_{k,0} \rangle_{p^n-1} \times \langle \xi_{k,1} \rangle_p \times \ldots \times \langle \xi_{k,n(k-1)} \rangle_p.$$

Доказательство. Элементы $R_k \setminus R_k^*$ — это многочлены вида $f(x) \cdot (a_0 + a_1 x + \ldots + a_{n(k-1)-1} x^{n(k-1)-1})$. Отсюда следует утверждение о $|R_k^*|$. Утверждение о разложении R_k^* в прямое произведение следует из теоремы о разложении конечной абелевой группы в прямое произведение циклических групп и леммы 5.12. □

Лемма 5.32. *Пусть* $k \geqslant 2$, $\xi \in R_k$. *Тогда элемент* ξ *единственным образом представим в виде*

$$\xi = a_0(x) + a_1(x) f(x) + \ldots + a_{k-1}(x) f^{k-1}(x) \pmod{f^k(x)},$$

где $a_i(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $\deg < n$. *При этом* $\xi \in R_k^*$ *тогда и только тогда, когда* $a_0(x) \neq 0$. *Далее, порядок* ξ *равен степени числа* p *тогда и только тогда, когда* $a_0(x) = 1$.

Доказательство. Докажем только последнее утверждение леммы. Достаточность следует из равенства $\xi^{p^k} \equiv a_0(x)^{p^k} \pmod{f^k}$. Необходимость следует из леммы 5.31, так как количество элементов порядка, равного степени p, равно $p^{n(k-1)}$ — как раз столько, сколько имеется элементов ξ, у которых $a_0(x) = 1$. □

Лемма 5.33. *Пусть* $2 \leqslant k \leqslant p$, $M = n(k-1)$, *и* $\eta_1, \ldots, \eta_M$ *есть следующий набор элементов порядка* p *в* R_k^*:

$$\begin{aligned}
&1 + f(x),\ 1 + x f(x),\ \ldots,\ 1 + x^{n-1} f(x);\\
&1 + f(x)^2,\ 1 + x f(x)^2,\ \ldots,\ 1 + x^{n-1} f(x)^2;\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&1 + f(x)^{k-1},\ 1 + x f(x)^{k-1},\ \ldots,\ 1 + x^{k-1} f(x)^{n-1}.
\end{aligned}$$

Тогда $R_k^* = \langle \eta_0 \rangle_{p^n-1} \times \langle \eta_1 \rangle_p \times \ldots \times \langle \eta_M \rangle_p$ *есть разложение* R_k^* *в прямое произведение* (*здесь* η_0 *— какой-либо элемент порядка* $p^n - 1$ *в* R_k^*).

Доказательство. Очевидно, что при $j = 1, \ldots, M$ элементы η_j имеют порядок p. Теперь для доказательства леммы достаточно показать,

что из равенства

$$\eta_0^{y_0}\eta_1^{y_1}\dots\eta_M^{y_M} \equiv \eta_0^{z_0}\eta_1^{z_1}\dots\eta_M^{z_M} \pmod{f^k},$$

где $0 \leqslant y_0,\ z_0 < p^n - 1$, $0 \leqslant y_j,\ z_j < p$, $j = 1, \dots, M$, следует, что $y_j = z_j$, $j = 1, \dots, M$. Действительно, так как $\eta_{lj} \equiv 1 \pmod{f(x)}$ при $j \geqslant 1$, то $\eta_{l0}^{y_0} \equiv \eta_{l0}^{z_0} \pmod{f(x)}$. Отсюда порядок элемента η_{l0} делит $(y_0 - z_0)p^k$. Поэтому $y_0 = z_0$. Сократив наше равенство на $\eta_{l0}^{y_0}$ и редуцируя его по mod f^2, получим

$$\eta_{l1}^{y_1}\dots\eta_{ln}^{y_n} \equiv \eta_{l1}^{z_1}\dots\eta_{ln}^{z_n} \pmod{f^2}.$$

Но

$$\begin{aligned}\eta_{l1}^{y_1}\dots\eta_{ln}^{y_n} &\equiv (1 + y_1 f(x))(1 + y_2 x f(x))\dots(1 + y_n x^{n-1} f(x)) \equiv \\ &\equiv (1 + (y_1 + y_2 x + \dots + y_n x^{n-1}) f(x)) \pmod{f^2};\\ \eta_{l1}^{z_1}\dots\eta_{ln}^{z_n} &\equiv (1 + (z_1 + z_2 x + \dots + z_n x^{n-1}) f(x)) \pmod{f^2}.\end{aligned}$$

Следовательно, $y_1 = z_1$, …, $y_n = z_n$ и т. д. □

Следующая лемма общеизвестна.

Лемма 5.34. *Пусть* $m_1, m_2 \in \mathbb{N}$, $d = (m_1, m_2)$. *Тогда* $(p^{m_1} - 1, p^{m_2} - 1) = p^d - 1$.

Лемма 5.35. *Пусть* $k \in \mathbb{N}$, $k \geqslant 2$, $h \in R_k^*$, *порядок* h *делит* p. *Рассмотрим функцию*

$$Q_0(h) = \frac{h^p - 1}{f^k} \pmod{f^k}.$$

Тогда $Q_0(h)$ *корректно определена и*

$$Q_0(h_1 h_2) = Q_0(h_1) + Q_0(h_2) \pmod{f^k}.$$

Доказательство. По лемме 5.32 h представим в виде

$$h \equiv 1 + a_1 f + \dots + a_{k-1} f^{k-1} \pmod{f^k}, \quad \deg a_i < n.$$

По условию

$$h^p \equiv 1 + a_1^p f^p + \dots + a_{k-1}^p f^{p(k-1)} \pmod{f^{kp}} \equiv 1 \pmod{f^k}.$$

Пусть $j \geqslant 1$, $(j-1)p < k \leqslant jp$; тогда очевидно, что $a_1 = \dots = a_{j-1} = 0$. Поэтому

$$h^p - 1 \equiv a_j^p f^{pj} + \dots + a_{k-1}^p f^{p(k-1)} \pmod{f^{kp}} \equiv 1 \pmod{f^{kp}}.$$

Отсюда следует корректность определения $Q_0(h)$. Далее,

$$h^p \equiv 1 + f^k Q_0(h) \pmod{f^{2k}}.$$

Поэтому

$$(h_1h_2)^p \equiv 1 + f^k \cdot (Q_0(h_1) + Q_0(h_2)) \pmod{f^{2k}}.$$

Лемма доказана. □

Теорема 5.36. *Пусть $k \geqslant 2$, $h_1, h_2 \in R_k^*$. Если порядок h_1 делит $p^n - 1$, то для разрешимости уравнения $h_1 = h_2^y$ необходимо и достаточно, чтобы порядок h_1 делил порядок h_2.*

Доказательство. Следует из леммы 5.31 и цикличности подгруппы $\langle \xi_{k,0} \rangle_{p^n-1} \ni h_1, h_2$. □

Теорема 5.37. *Пусть $p/2 < k \leqslant p$, элементы h_1, h_2 группы R_k^* имеют порядок p. Пусть (по лемме 5.32)*

$$h_1 \equiv 1 + a_1(x)f(x) + \ldots + a_{k-1}(x)f(x)^{k-1} \pmod{f^k},$$
$$h_2 \equiv 1 + b_1(x)f(x) + \ldots + b_{k-1}(x)f(x)^{k-1} \pmod{f^k},$$

$\deg a_i$, $\deg b_j < n$. *Если $b_1(x) \neq 0$ и уравнение $h_1 = h_2^y$ разрешимо, то $y_1 \equiv (a_1(x)b_1(x)^{-1})^p \pmod{f(x)}$.*

Доказательство. По лемме 5.35 выполнено сравнение $Q_0(h_1) \equiv yQ_0(h_2) \pmod{f^k}$. Далее, так как $p \geqslant k$, то

$$Q_0(h_1) \equiv a_1^p f^{p-k} \pmod{f^k}, \qquad Q_0(h_2) \equiv b_1^p f^{p-k} \pmod{f^k}.$$

Поскольку $2k - p \geqslant 1$, то получаем сравнение $a_1^p \equiv yb_1^p \pmod{f^{2k-p}}$, из которого следует утверждение теоремы. □

Замечание 5.38. Если заданы элементы h_1, h_2, имеющие порядок p, и $b_1(x) \neq 0$, то, найдя y по формуле теоремы 5.37, можно затем проверкой убедиться, выполнено ли равенство $h_1 = h_2^y$.

Теорема 5.39. *Пусть $p/2 < k \leqslant p$. Пусть h_1, $h_2 \in R_k^*$, порядок h_1 равен pd_1, порядок h_2 равен pd_2, и $d_1 | d_2 | p^n - 1$. Пусть (по лемме 5.22)*

$$h_1^{p^n-1} \equiv 1 + a_1(x)f(x) + \ldots + a_{k-1}(x)f(x)^{k-1} \pmod{f^k},$$
$$h_2^{p^n-1} \equiv 1 + b_1(x)f(x) + \ldots + b_{k-1}(x)f(x)^{k-1} \pmod{f^k},$$

$\deg a_i$, $\deg b_j < n$ *и* $b_1(x) \neq 0$.

1) *Предположим, что найдется $y_0 \in \mathbb{Z}/p\mathbb{Z}$, для которого выполнено сравнение $y_0 \equiv (a_1 b_1^{-1})^p \pmod{f(x)}$. Если*

$$h_1^{p^n-1} \equiv (h_2^{p^n-1})^{y_0},$$

то уравнение $h_1 = h_2^y$ разрешимо.

2) *Если класс вычетов* $(a_1 b_1^{-1})^p \pmod{f(x)}$ *не содержит элемента* $y_0 \in \mathbb{Z}/p\mathbb{Z}$, *или он содержит* $y_0 \in \mathbb{Z}/p\mathbb{Z}$, *но* $h_1^{p^n-1} \neq (h_2^{p^n-1})^{y_0}$, *то уравнение* $h_1 = h_2^y$ *неразрешимо.*

Доказательство. Докажем первое утверждение. По лемме 5.31

$$h_1 \equiv \xi_{k0}^{u_0} \xi_{k1}^{u_1} \dots \xi_{k,n(k-1)}^{u_{n(k-1)}} \pmod{f^k}, \qquad h_2 \equiv \xi_{k0}^{v_0} \xi_{k1}^{v_1} \dots \xi_{k,n(k-1)}^{v_{n(k-1)}} \pmod{f^k}.$$

Если $d_1 | d_2$, то найдется y_1, $0 \leqslant y_1 < p^n - 1$, такое, что $\xi_{k0}^{u_0} \equiv (\xi_{k0}^{v_0})^{y_1} \pmod{f^k}$. Кроме того, из условия следует, что

$$\left(\xi_{k0}^{u_0} \xi_{k1}^{u_1} \dots \xi_{k,n(k-1)}^{u_{n(k-1)}}\right)^{p^n-1} \equiv \left(\xi_{k0}^{v_0} \xi_{k1}^{v_1} \dots \xi_{k,n(k-1)}^{v_{n(k-1)}}\right)^{(p^n-1)y_0} \pmod{f^k}.$$

Очевидно, что натуральное число y такое, что $y \equiv y_0 \pmod{p}$, $y \equiv y_1 \pmod{p^n - 1}$, будет решением уравнения $h_1 = h_2^y$.

Второе утверждение теоремы доказывается аналогично теореме 5.37. □

Рассмотрим теперь задачу дискретного логарифмирования

$$P \equiv Q^y \pmod{F}, \quad (P, F) = (Q, F) = 1,$$

где P, Q, $F \in \mathbb{Z}/p\mathbb{Z}[x]$, $\deg F = N \geqslant 2$, F — унитарный многочлен. Пусть

$$F = F(x) = f_1(x) \dots f_s(x), \quad s \geqslant 2,$$

где $f_j(x)$ — различные унитарные неприводимые многочлены из $\mathbb{Z}/p\mathbb{Z}[x]$, $\deg f_j = n_j$.

Очевидно, что $P \equiv Q^y \pmod{F}$ тогда и только тогда, когда $P \equiv Q^y \pmod{f_j(x)}$, $j = 1, \dots, s$. Обозначим через $\varkappa_j$ порядок $Q \pmod{f_j(x)}$. Уравнение $P \equiv Q^y \pmod{f_j(x)}$ разрешимо тогда и только тогда, когда порядок $P \pmod{f_j(x)}$ делит $\varkappa_j$. В этом случае найдется $y_j \pmod{\varkappa_j}$, такое, что $P \equiv Q^{y_j} \pmod{f_j(x)}$. Для проверки разрешимости $P \equiv Q^y \pmod{F}$ теперь надо проверить разрешимость системы уравнений $y \equiv y_j \pmod{\varkappa_j}$, $j = 1, \dots, s$. Для разрешимости этой системы необходимо и достаточно, чтобы $y_i \equiv y_j \pmod{\varkappa_{ij}}$ при всех $i \neq j$, где $\varkappa_{ij} = (\varkappa_i, \varkappa_j)$. Если $\varkappa_{ij}$ невелики, то $y_i \pmod{\varkappa_{ij}}$ можно определить перебором из сравнений

$$P^{\varkappa_i/\varkappa_{ij}} \equiv Q^{(\varkappa_i/\varkappa_{ij})y_j} \pmod{f_i(x)},$$

и затем убедиться в том, что $y_i \equiv y_j \pmod{\varkappa_{ij}}$. Таким образом, мы описали алгоритм проверки разрешимости уравнения $P \equiv Q^y \pmod{F}$, который в некоторых случаях будет быстрым. Например, если p невелико и числа n_i попарно взаимно просты, то по лемме 5.34 все $\varkappa_{ij}$ не превосходят $p - 1$, и перебор будет коротким. Заметим также, что для определения порядков элементов нам надо знать разложение чисел $p^{n_i} - 1$ на множители.

Замечание 5.40. Для произвольного приводимого многочлена $F(x) \in \mathbb{Z}/p\mathbb{Z}[x]$ также можно описать аналогичный алгоритм проверки разрешимости в случае, когда все кратности неприводимых делителей $F(x)$ невелики.

§ 5.7. Заключение

Из более старых работ по дискретному логарифмированию читатель может обратить внимание на [130; 291; 131].

Из обзоров по дискретному логарифмированию мы можем рекомендовать [210; 176; 83; 209].

В работе [48] содержится великолепный обзор по сложности теоретико-числовых алгоритмов, включая задачу дискретного логарифмирования.

Вопросам получения оценок сложности в задаче дискретного логарифмирования посвящены работы [35; 34; 223; 174; 254; 255; 147]. В частности, в работе [147] оценивается снизу порядок линейной рекуррентной последовательности u_j, обладающей тем свойством, что $\log_a j \equiv u_j \pmod p$ для всех номеров j из некоторого промежутка.

В работе [268] содержится обзор алгоритмов дискретного логарифмирования в произвольной группе G со сложностью $O(|G|^{1/2})$ групповых операций. См. также [201].

Работа [128] посвящена параллельному вычислению дискретных логарифмов.

В работе [82] рассматриваются вопросы дискретного логарифмирования в произвольной конечной абелевой группе G, а также алгоритмы нахождения порядка элемента и определения структуры G. Доказана, в частности, следующая теорема.

Теорема 5.41. *Имеется алгоритм, который для заданных g, $d \in G$, $d \neq 1$, определяет, принадлежит ли d подгруппе $\langle g \rangle$, и если да, то находит $\log_g d$. Если $x = |\langle g \rangle|$ при $d \notin \langle g \rangle$ и $x = \log_g d$ при $d \in \langle g \rangle$, то алгоритм выполняет не более $6\lceil\sqrt{x}\,\rceil + \lceil\log_2 \sqrt{x}\,\rceil$ умножений в G. Он использует таблицу из не более чем $2\lceil\sqrt{x}\,\rceil$ пар $(h, r) \in G \times \{1, \ldots, 2\sqrt{x}\}$. Общее число обращений к таблице не превосходит $4\lceil\sqrt{x}\,\rceil$.*

В работе [251] получена оценка для наименьшего первообразного корня a по простому модулю p. В предположении расширенной гипотезы Римана

$$a = O(r^4(\log r + 1)^4(\log p)^2),$$

где r — число различных простых делителей $p-1$.

В различных криптографических схемах используются пары простых чисел q и p, где $p=2q+1$. В частности, для тестирования алгоритмов дискретного логарифмирования по простому модулю p обычно используют такие пары. В этом случае первообразный корень по модулю p можно определить с помощью следующей теоремы, см. [42, с. 168—170].

Теорема 5.42. *Пусть p, q — простые числа.*

1) *Если $q=4n+1$, $p=2q+1$, то 2 является первообразным корнем по модулю p.*

2) *Если $q=4n+3$, $p=2q+1$, то -2 является первообразным корнем по модулю p.*

Доказательство. Для того, чтобы число a являлось первообразным корнем по модулю p, необходима и достаточна неразрешимость уравнений

$$x^2 \equiv a \pmod{p}, \qquad x^q \equiv a \pmod{p}.$$

Пусть $q=4n+1$, тогда $p=8n+3$. В этом случае уравнение $x^2 \equiv 2 \pmod{p}$ неразрешимо, так как $\left(\frac{2}{p}\right) = -1$. Предположим, что уравнение $x^{4n+1} \equiv 2 \pmod{p}$ разрешимо. Тогда $x^{8n+2} \equiv 4 \pmod{8n+3}$. Отсюда по малой теореме Ферма получим, что $4 \equiv 1 \pmod{p}$. Так как $p > 3$, то это невозможно.

Пусть $q=4n+3$, $p=8n+7$. Уравнение $x^2 \equiv -2 \pmod{p}$ неразрешимо, поскольку

$$\left(\frac{-2}{p}\right) = \left(\frac{-1}{p}\right)\left(\frac{2}{p}\right) = (-1)^{\frac{p-1}{2}} = -1.$$

Предположим, что уравнение $x^{4n+3} \equiv -2 \pmod{p}$ разрешимо. Тогда $x^{8n+6} \equiv 4 \pmod{p}$, и мы получаем противоречие аналогично предыдущему случаю. □

Глава 6. Факторизация многочленов над конечными полями

§ 6.1. Введение. Вероятностный алгоритм решения алгебраических уравнений в конечных полях

В данной главе мы рассматриваем алгоритмы разложения на множители многочленов над конечными полями, а также некоторые методы решения алгебраических уравнений в конечных полях.

Пусть p — простое число, $p > 2$, $f(x) \in GF(p)[x]$, $\deg f(x) = n \geqslant 2$. Рассмотрим вероятностный алгоритм решения уравнения $f(x) = 0$ в поле $GF(p)$.

Алгоритм решения $f(x) = 0$ в $GF(p)$.

1 шаг. Вычислить

$$g(x) = \text{НОД}(f(x),\ x^p - x) \in GF(p)[x].$$

Заметим, что все корни $f(x)$ в $GF(p)$ являются корнями многочлена $g(x)$ кратности 1, и других корней у $g(x)$ нет. Если $\deg g(x) = 0$, то корней у $f(x)$ в $GF(p)$ нет. Если $\deg g(x) = 1$, $g(x) = x - a$, то a — единственный корень $f(x)$ в поле $GF(p)$ (без учета кратности). Если $\deg g(x) = p$, то все элементы $GF(p)$ являются корнями $f(x)$. Далее мы предполагаем, что $2 \leqslant \deg g(x) < p$, и ищем корни $g(x)$ в $GF(p)$.

2 шаг. Случайным образом выбрать элемент $\delta \in GF(p)$ и вычислить

$$d(x) = \text{НОД}\Big((x + \delta)^{\frac{p-1}{2}} - 1,\ g(x)\Big).$$

3 шаг. Если $d(x) = 1$ или $d(x) = g(x)$, то вернуться на шаг 2. Если $\deg d(x) = 1$, $d(x) = x - b$, то b — корень многочлена $f(x)$; мы заносим его в список найденных корней, заменяем $g(x)$ на $g(x)/(x - b)$ и возвращаемся на шаг 2. Аналогично, при $\deg d(x) = \deg g(x) - 1$ мы находим $x - b = \dfrac{g(x)}{d(x)}$, заносим b в список корней, заменяем $g(x)$ на $d(x)$ и возвращаемся на 2-й шаг. Если $2 \leqslant \deg d(x) < \deg g(x) - 1$, то мы рассматриваем вместо $g(x)$ два его делителя — многочлены $d(x)$ и $g(x)/d(x)$,

и к каждому из них мы применяем 2 и 3 шаги алгоритма, чтобы найти их корни.

Конец алгоритма.

Теорема 6.1. *Если* $g(x) \in GF(p)[x]$, $2 \leqslant \deg g(x) < p$ *и* $g(x) \mid x^p - x$, *то при случайном выборе* δ *на* 2*-м шаге с вероятностью, не меньшей, чем* $\frac{1}{2} - \frac{1}{2p}$, *будет выполнено неравенство* $1 \leqslant \deg d(x) < \deg g(x)$.

Доказательство. Пусть $a_1, a_2 \in GF(p)$, $a_1 \neq a_2$, $g(a_1) = g(a_2) = 0$. Пусть $\delta \neq -a_1$, $\delta \neq -a_2$. Тогда

$$(a_i + \delta)^{\frac{p-1}{2}} - 1 \quad \text{равно } 0 \text{ или } -2 \quad \text{при} \quad i = 1, 2.$$

Если

$$(a_1 + \delta)^{\frac{p-1}{2}} - 1 \neq (a_2 + \delta)^{\frac{p-1}{2}} - 1,$$

то ровно одно из чисел a_1, a_2 является корнем $d(x)$ и поэтому $1 \leqslant \deg d(x) < \deg g(x)$. Теперь предположим, что $\delta \neq -a_1, -a_2$, и

$$(a_1 + \delta)^{\frac{p-1}{2}} - 1 = (a_2 + \delta)^{\frac{p-1}{2}} - 1$$

Такие δ являются корнями многочлена $(x + a_1)^{\frac{p-1}{2}} - (x + a_2)^{\frac{p-1}{2}}$, который имеет степень $\frac{p-1}{2} - 1$. Поэтому, если мы рассмотрим все $\delta \in GF(p)$, то не более чем для $2 + \frac{p-1}{2} - 1 = \frac{p+1}{2}$ элементов δ многочлен $d(x)$ может не удовлетворять неравенству $1 \leqslant \deg d(x) < \deg g(x)$. Отсюда следует утверждение теоремы. □

Теорема 6.2. *В условиях теоремы* 6.1 *обозначим через* k *степень* $\deg g(x)$; *тогда вероятность будет равна* $1 - \frac{1}{2^{k-1}} + O\left(\frac{1}{\sqrt{p}}\right)$; *постоянная в* $O(\cdot)$ *зависит от* k.

Лемма 6.3. *Пусть* $g(x) = \prod_{i=1}^{k} (x - a_i)$, *где* $a_i \in GF(p)$, a_i *различны*; *пусть* $\varepsilon_1, \ldots, \varepsilon_k$ — *какой-либо фиксированный набор из* ± 1. *Рассмотрим* δ, *удовлетворяющие условиям*

$$0 \leqslant \delta \leqslant p - 1, \quad \left(\frac{a_i + \delta}{p}\right) = \varepsilon_i, \quad i = 1, \ldots, k.$$

Тогда количество таких δ *равно* $N = \frac{p}{2^k} + O(\sqrt{p})$, *где постоянная в* $O(\cdot)$ *зависит от* k.

Выведем теорему 6.2 из леммы. Пусть $0 \leqslant \delta \leqslant p-1$, $\delta \neq -a_i$, $i = 1, \ldots, k$. Предположим, что $\deg d(x) = 0$. Тогда

$$(a_i + \delta)^{\frac{p-1}{2}} - 1 \neq 0, \quad i = 1, \ldots, k.$$

Следовательно,

$$(a_i + \delta)^{\frac{p-1}{2}} = -1, \quad i = 1, \ldots, k,$$

т. е. $\varepsilon_1 = \ldots = \varepsilon_k = -1$. Теперь рассмотрим случай $\deg d(x) = k$. Тогда аналогично получаем, что

$$(a_i + \delta)^{\frac{p-1}{2}} = 1, \quad i = 1, \ldots, k,$$

откуда $\varepsilon_1 = \ldots = \varepsilon_k = 1$. Таким образом, среди всех значений δ количество тех δ, для которых $1 \leqslant \deg d(x) < k$, будет по лемме равно $p - k - 2\left(\frac{p}{2^k} + O(\sqrt{p})\right)$. Отсюда следует утверждение теоремы 6.2.

Доказательство. Пусть a — фиксированный квадратичный невычет по модулю p. Положим $d_i = a$, если $\varepsilon_i = -1$, и $d_i = 1$, если $\varepsilon_i = 1$. Рассмотрим систему уравнений

$$\begin{cases} y_1^2 \equiv d_1(x + a_1) \pmod{p}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ y_k^2 \equiv d_k(x + a_k) \pmod{p}, \end{cases}$$

относительно неизвестных $x, y_1, \ldots, y_k \in \mathbb{Z}/p\mathbb{Z}$. Обозначим для краткости $\chi(d) = \left(\frac{d}{p}\right)$. Уравнение $y^2 \equiv d \pmod{p}$ имеет $1 + \chi(d)$ решений. Поэтому число решений системы равно

$$M = \sum_{x=0}^{p-1} \prod_{i=1}^{k} (1 + \chi(d_i(x + a_i))) =$$

$$= \sum_{x=0}^{p-1} \sum_{l_1, \ldots, l_k \in \{0,1\}} \chi(d_1(x + a_1))^{l_1} \ldots \chi(d_k(x + a_k))^{l_k} =$$

$$= p + \sum_{\substack{\bar{l} = (l_1, \ldots, l_k); \\ \bar{l} \neq 0}} \sum_{x=0}^{p-1} \chi(g_{\bar{l}}(x)),$$

где $g_{\bar{l}}(x) = (d_1(x+a_1))^{l_1} \dots (d_k(x+a_k))^{l_k}$. Справедливо неравенство (см. [31, теорема 5.41])

$$\left| \sum_{x=0}^{p-1} \chi(g_{\bar{l}}(x)) \right| \leqslant (\deg g_{\bar{l}}(x) - 1)\sqrt{p},$$

которое следует из оценок тригонометрических сумм Вейля. Поэтому

$$|M - p| \leqslant \sum_{r=2}^{k} \binom{k}{r} (r-1)\sqrt{p},$$

так как если $\deg g_{\bar{l}}(x) = 1$, то $\sum\limits_{x=0}^{p-1} \chi(g_{\bar{l}}(x)) = 0$. Поскольку

$$(r-1)\binom{k}{r} = \frac{k!\,(r-1)}{r!\,(k-r)!} < \frac{k!}{(r-1)!\,(k-r)!} = k\binom{k-1}{r-1},$$

то $|M - p| \leqslant k\sqrt{p}(2^{k-1} - 1)$.

Рассмотрим δ, удовлетворяющие условию леммы. Тогда

$$\chi(d_i(\delta + a_i)) = \left(\frac{d_i(\delta + a_i)}{p}\right) = \left(\frac{d_i}{p}\right)\varepsilon_i = 1$$

по определению d_i. Поэтому каждое уравнение $y_i^2 \equiv d_i(\delta + a_i) \pmod p$ имеет два решения, и система при $x = \delta$ имеет 2^k решений. Следовательно,

$$M = 2^k N + N_1,$$

где N_1 — некоторые решения системы, возникающие при δ, не удовлетворяющих условию леммы. Если $\delta \neq -a_i$, $i = 1, \dots, k$, но $\left(\frac{\delta + a_i}{p}\right) = -\varepsilon_i$ при некотором i, то $\left(\frac{d_i(a_i + \delta)}{p}\right) = \left(\frac{d_i}{p}\right)(-\varepsilon_i) = -1$, и система при $x = \delta$ неразрешима. Если же $\delta = -a_i$ при некотором i, то при $x = \delta$ система имеет не более чем 2^k решений. Поэтому $N_1 \leqslant k \cdot 2^k$. Следовательно,

$$|2^k \cdot N - p| \leqslant |M - p| + |2^k N - M| \leqslant k\sqrt{p}(2^{k-1} - 1) + k \cdot 2^k.$$

Отсюда следует утверждение леммы 6.3. □

Скажем еще несколько слов о других методах нахождения корней многочленов над конечными полями. Эти методы описаны в [31, гл. 4, § 3]. Если конечное поле $GF(q)$ велико, но его характеристика p мала, то при $q = p^m$ следует вычислить многочлен $S(x) = \sum\limits_{j=0}^{m-1} x^{p^j}$, и пытаться

находить корни $f(x)$, вычисляя либо

$$\text{НОД}(f(x),\ S(x)-c), \qquad c\in GF(p),$$

либо

$$\text{НОД}(f(x),\ S(\beta^j x)-c), \qquad c\in GF(p),$$

где $\beta\in GF(q)$ такое, что $1,\ \beta,\ \ldots,\ \beta^{m-1}$ образуют базис $GF(q)$ над $GF(p)$ как линейного пространства. Если же и поле $GF(q)$ велико, и его характеристика p велика, то при $f(x)=\sum\limits_{i=0}^{n}\alpha_j x^j$ мы сначала строим многочлены

$$f_k(x)=\sum_{j=0}^{n}\alpha_j^{p^k}x^j, \qquad k=0,\ \ldots,\ m-1.$$

Затем для многочлена $F(x)=\prod\limits_{k=0}^{m-1}f_k(x)\in GF(p)[x]$ находим разложение $F(x)=G_1(x)\ldots G_r(x)$ в кольце $GF(p)[x]$ на степени различных неприводимых многочленов и пытаемся найти корни $f(x)$, вычисляя $\text{НОД}(f(x),\ G_t(x))$, $t=1,\ \ldots,\ r$.

На этом мы закончим описание методов нахождения корней произвольных многочленов над $GF(q)$. В следующем параграфе мы будем решать квадратные уравнения в конечных полях.

§ 6.2. Решение квадратных уравнений

Рассмотрим уравнение

$$x^2\equiv a \pmod p, \tag{6.1}$$

где $p>2$, p — простое число. С помощью очевидной замены переменной к уравнению (6.1) сводится произвольное уравнение $AX^2+BX+C\equiv 0 \pmod p$. Если $a\not\equiv 0 \pmod p$, то уравнение (6.1) разрешимо тогда и только тогда, когда $\left(\dfrac{a}{p}\right)=1$, т. е.

$$a^{(p-1)/2}\equiv 1 \pmod p \tag{6.2}$$

Предположим, что $p\equiv 3 \pmod 4$, т. е. $p=4k+3$. Тогда $\dfrac{p-1}{2}=2k+1$, и из (6.2) следует, что $a^{2k+1}\equiv 1 \pmod p$. Отсюда $a^{2k+2}\equiv a \pmod p$, и мы находим решение (6.1):

$$x\equiv\pm a^{k+1} \pmod p.$$

Предположим, что $p \equiv 1 \pmod 4$, $p = 4k+1$. Тогда $\dfrac{p-1}{2} = 2k$, и (6.2) принимает вид

$$a^{2^s t} \equiv 1 \pmod p, \tag{6.3}$$

где $2k = 2^s t$, $s \geqslant 1$, t нечетно. Для решения (6.1) нам нужно знать какой-либо квадратичный невычет N по модулю p. Для N выполнено соотношение

$$-1 = \left(\frac{N}{p}\right) \equiv N^{2k} \equiv N^{2^s t} \pmod p.$$

Теперь будем извлекать квадратные корни. Из (6.3) следует, что

$$a^{2^{s-1} t} \equiv \pm 1 \pmod p.$$

Если $a^{2^{s-1}t} \equiv +1 \pmod p$, то мы снова извлекаем квадратный корень, если же $a^{2^{s-1}t} \equiv -1 \pmod p$, то $a^{2^{s-1}t} \cdot N^{2^s t} \equiv 1 \pmod p$, и мы извлекаем корень из левой части этого уравнения. Продолжая этот процесс извлечения квадратных корней и домножения на $N^{2^s t} \equiv -1 \pmod p$, если при извлечении корня появляется $-1 \pmod p$, мы придем к соотношению

$$a^t \cdot N^{2l} \equiv 1 \pmod p$$

при некотором $l \in \mathbb{Z}_{\geqslant 0}$. Отсюда $(a^{(t+1)/2} N^l)^2 \equiv a \pmod p$, и мы находим $x \equiv \pm a^{(t+1)/2} N^l \pmod p$ — решение (6.1).

Замечание 6.4. С помощью случайного выбора N из множества $\{1, 2, \ldots, p-1\}$ мы с вероятностью $\dfrac{1}{2}$ найдем невычет. Оценка на величину наименьшего квадратичного вычета по модулю p была приведена в § 1.6 (при условии выполнения расширенной гипотезы Римана). Для некоторых значений $p \equiv 1 \pmod 4$ невычет N известен заранее; например, $N = 2$ при $p \equiv 5 \pmod 8$.

Несколько более эффективным для решения уравнения (6.1) в случае $p \equiv 1 \pmod 4$ является алгоритм Тонелли—Шэнкса. В нем мы сохраняем обозначение $p - 1 = 4k = 2^{s+1} \cdot t = 2^e \cdot t$; N — какой-либо известный нам квадратичный невычет по модулю p. Мы считаем, что $\left(\dfrac{a}{p}\right) = +1$.

Алгоритм Тонелли—Шэнкса.

1 шаг. Вычисляем следующие значения:

$$y := N^t \pmod p, \quad r := e, \quad x := a^{(t-1)/2} \pmod p,$$
$$b := ax^2 \pmod p, \quad x := ax \pmod p.$$

2 шаг. Если $b \equiv 1 \pmod p$, то x является искомым решением (6.1), и алгоритм останавливается.

3 шаг. Находим наименьшее $m \in \mathbb{N}$ такое, что $b^{2^m} \equiv 1 \pmod p$. Оно удовлетворяет неравенству $1 \leqslant m \leqslant r-1$.

4 шаг. Вычисляем значения

$$l := y^{2^{r-m-1}} \pmod p, \quad y := l^2, \quad r := m$$
$$x := xl \pmod p, \quad b := by \pmod p$$

и возвращаемся на 2-й шаг.

Конец алгоритма.

Покажем, что алгоритм работает корректно. Мы хотим найти числа $A_1, \ldots, A_{e-1}$, равные 0 или 1, такие, что

$$a^t \cdot N^{t(A_1 \cdot 2 + A_2 \cdot 2^2 + \ldots + A_{e-1} \cdot 2^{e-1})} \equiv 1 \pmod p.$$

Числа $A_1, \ldots, A_{e-1}$ мы определяем последовательно. При первом проходе алгоритма на 1-м шаге

$$b \equiv a^t \pmod p, \quad x \equiv a^{(t+1)/2} \pmod p, \quad r = e.$$

На 3 шаге мы находим m такое, что $b^{2^m} \equiv a^{t \cdot 2^m} \equiv 1 \pmod p$, $b^{2^{m-1}} = a^{t \cdot 2^{m-1}} \equiv -1 \pmod p$, и в силу (6.2), $m \leqslant r-1 = e-1$. Тогда мы полагаем $A_1 = \ldots = A_{e-m-1} = 0$, $A_{e-m} = 1$, и получаем соотношение

$$a^{2^{e-j}t} \cdot N^{t(A_1 \cdot 2^{e-j+1} + \ldots + A_{j-1} \cdot 2^{e-1})} \equiv 1 \pmod p,$$

где $j = e - m + 1$. Далее на 4 шаге мы вычисляем

$$l = N^{t \cdot 2^{r-m-1}} = N^{t \cdot 2^{j-2}}, \quad y = N^{t \cdot 2^{j-1}}, \quad r = m = e - j + 1,$$
$$x = a^{\frac{t+1}{2}} \cdot N^{t(A_1 + \ldots + A_{e-m} 2^{e-m-1})} \pmod p,$$
$$b = a^t \cdot N^{t(2A_1 + \ldots + A_{e-m} 2^{e-m})} \pmod p.$$

Если $b \equiv 1 \pmod p$, то очевидно, что x является решением (6.1). Если $b \not\equiv 1 \pmod p$, то алгоритм продолжает работу.

Предположим, что после нескольких проходов шагов 2—4 мы нашли числа $j \geqslant 2$ и $A_1, \ldots, A_{j-1} \in \{0; 1\}$, причем $A_{j-1} = 1$ и выполнены соотношения, являющиеся предположением индукции:

$$a^{2^{e-j} \cdot t} N^{t(A_1 2^{e-j+1} + \ldots + A_{j-1} 2^{e-1})} \equiv 1 \pmod p,$$
$$x \equiv a^{\frac{t+1}{2}} N^{t(A_1 + \ldots + A_{j-1} 2^{j-2})} \pmod p,$$
$$b \equiv a^t N^{t(A_1 \cdot 2 + \ldots + A_{j-1} 2^{j-1})} \pmod p,$$
$$l = N^{t \cdot 2^{j-2}}, \quad y = N^{t 2^{j-1}}, \quad r = e - j + 1.$$

Если $j = e$, то мы определили все $A_1, \ldots, A_{e-1}$, и x будет ответом. Если $j < e$, но $b \equiv 1 \pmod p$, то x также будет ответом, и алгоритм закончит работу. Если же $j < e$ и $b \not\equiv 1 \pmod p$, то при следующем проходе шагов 2—4 алгоритма мы находим очередное значение j' и числа $A_j = A_{j+1} = \ldots = A_{j'-1} = 0$, $A_{j'} = 1$, для которых

$$a^{2^{e-j'} \cdot t} N^{t(A_1 2^{e-j'+1} + \ldots + A_{j'-1} 2^{e-1})} \equiv 1 \pmod p.$$

При этом формулы, выражающие x, b, l, y, r через j', $A_1, \ldots, A_{j'-1}$, сохраняются. Таким образом мы по индукции доказали корректность алгоритма Тонелли—Шэнкса.

Замечание 6.5. Алгоритм Тонелли—Шэнкса описан в [89, гл. 1], см. также [60, гл. 7]. Он имеет полиномиальную сложность при условии, что невычет N нам известен.

Замечание 6.6. В работе [244] предложен алгоритм решения (6.1) со сложностью $O(|a|^{1/2+\varepsilon}(\log p)^9)$ битовых операций. Знание невычета здесь не предполагается; при фиксированном $a \in \mathbb{Z}$ алгоритм имеет полиномиальную сложность по переменной величине p.

Замечание 6.7. В книге [60, гл. 7] описан вероятностный алгоритм Чипполы для решения уравнения $x^2 = a$ в конечном поле $GF(q)$ при нечетном q, имеющий среднее время работы $O(\log^3 q)$ битовых операций.

Рассмотрим теперь уравнение

$$x^N \equiv a \pmod p, \tag{6.4}$$

где p — простое число, $N \in \mathbb{N}$, $N > 2$. Если $(N, p-1) = 1$, то решение этого уравнения имеет вид $x \equiv a^M \pmod p$, где $NM \equiv 1 \pmod{p-1}$. В работах [282; 49] предложены некоторые методы решения (6.4) при условии $(N, p-1) > 1$; см. также [60, гл. 7].

Если мы хотим решить уравнение $x^2 \equiv a \pmod n$, где $n = pq$, p, q — различные простые числа, то решение этого уравнения будет трудной задачей, при условии, что разложение n на множители нам неизвестно. На сложности решения этого уравнения основан ряд криптосистем, см. [4, гл. 3, 4].

Если $n \in \mathbb{N}$, то для нахождения $m = [\sqrt{n}]$ можно использовать следующий алгоритм, описанный в [89, гл. 1].

Алгоритм.

1 шаг. $x := n$.

2 шаг. Используя целочисленные деления и сдвиги (деление на 2), вычислить

$$y = \left[\frac{x + \left[\frac{n}{x}\right]}{2}\right].$$

3 шаг. Если $y < x$, то $x := y$ и вернуться на 2-й шаг. Иначе x будет искомым значением $[\sqrt{n}]$.

Конец алгоритма.

Покажем, что алгоритм выдает верный ответ. Если $t > 0$, то выполнено неравенство

$$\frac{t + \frac{n}{t}}{2} \geqslant \sqrt{n}.$$

Поэтому в алгоритме всегда $x \geqslant [\sqrt{n}]$, так как из неравенства $x + \frac{n}{x} \geqslant$ $\geqslant 2\sqrt{n}$ следует, что $x + \left[\frac{n}{x}\right] \geqslant [2\sqrt{n}] \geqslant 2[\sqrt{n}]$. Пусть на 3 шаге выполнено неравенство $y \geqslant x$. Мы хотим доказать, что $x = [\sqrt{n}]$. Предположим, что $x \geqslant [\sqrt{n}] + 1$, и придем к противоречию. Действительно,

$$0 \leqslant y - x = \left[\frac{x + \left[\frac{n}{x}\right]}{2}\right] - x = \left[\frac{\left[\frac{n}{x}\right] - x}{2}\right] < 0,$$

поскольку $x > \sqrt{n}$ и $\left[\frac{n}{x}\right] \leqslant \frac{n}{x} < \sqrt{n}$, т. е. $\left[\frac{n}{x}\right] - x \leqslant [\sqrt{n}] - [\sqrt{n}] - 1 = -1$.

С помощью данного алгоритма можно решать уравнения $x^2 = n$, $n \in \mathbb{N}$, в целых числах; прежде чем извлекать корень из n, следует проверить, является ли n квадратичным вычетом по нескольким небольшим модулям, см. [89, гл. 1].

§ 6.3. Алгоритм Берлекэмпа

Пусть p — простое число, $q = p^m$. В данном параграфе мы опишем алгоритм Берлекэмпа (см. [7; 31, гл. 4; 63]) для разложения унитарного многочлена $f(x) \in GF(q)[x]$ на множители. Пусть $\deg f(x) = n \geqslant 2$. Обозначим

$$f(x) = f_1(x)^{e_1} \dots f_k(x)^{e_k} \tag{6.5}$$

разложение $f(x)$ на различные неприводимые унитарные многочлены $f_1(x), \dots, f_k(x)$. Сначала мы покажем, как нахождение разложения (6.5) сводится к нахождению разложения многочлена

$$\tilde{f}(x) = f_{i_1}(x) \dots f_{i_l}(x), \qquad 1 \leqslant i_1 < i_2 < \dots < i_l \leqslant z, \tag{6.6}$$

не имеющего кратных неприводимых множителей. Для этого рассмотрим

$$d(x) = \text{НОД}(f(x),\, f'(x)).$$

Если $f'(x)$ — нулевой многочлен, то $f(x)=g_0(x^p)$, где $g_0(y)\in GF(q)[y]$. Поскольку возведение в степень p является автоморфизмом поля $GF(q)$ и так как $(A+B)^p=A^p+B^p$ для всех $A, B\in GF(q)[x]$, то $f(x)=g_0(x^p)=(g(x))^p$, и задача разложения $f(x)$ сводится к нахождению разложения $g(x)$, имеющего меньшую степень. Коэффициенты $g(x)$ быстро вычисляются по коэффициентам $f(x)$, если $GF(q)$ не слишком велико и $m>1$. Если же $q=p$ (т. е. $m=1$), то $g(x)=g_0(x)$, поскольку $a^p=a$ для всех $a\in GF(p)$.

Предположим теперь, что $f'(x)$ — ненулевой многочлен. Тогда очевидно, что

$$d(x)=f_1(x)^{v_1}\dots f_k(x)^{v_k},$$

где $v_i=e_i-1$ при $p\nmid e_i$, $v_i=e_i$ при $p\mid e_i$. При этом найдется $v_i\neq e_i$. Тогда многочлен $\tilde f(x)=f(x)/d(x)$ равен произведению тех из многочленов $f_1(x), \dots, f_k(x)$, для которых $p\nmid e_i$. Если мы разложим $\tilde f(x)$ на неприводимые множители, то затем пробными делениями найдем показатели e_i такие, что $p\nmid e_i$. Вычислим $f(x)/\prod\limits_{i:\ p\nmid e_i} f_i(x)^{e_i}$. У этого многочлена производная будет нулевой. Если он отличен от константы, то мы представим его в виде $\hat g(x)^p$, как это описано выше, и далее будем раскладывать $\hat g(x)$ аналогичным образом, и т. д.

Замечание 6.8. В [89, гл. 3, алгоритм 3.4.2] описан алгоритм, который находит представление унитарного многочлена $f(x)\in\mathbb{Z}/p\mathbb{Z}[x]$ в виде $f(x)=\prod\limits_{i\geqslant 1} A_i(x)^i$, где $A_i(x)\in\mathbb{Z}/p\mathbb{Z}[x]$, A_i не имеют кратных неприводимых множителей, и при $i\neq j (A_i(x), A_j(x)=1$.

Далее мы будем считать, что в разложении (6.5) все показатели e_i равны 1.

Теорема 6.9. *Пусть $h(x)\in GF(q)[x]$, $1\leqslant\deg h(x)<n$, и выполнено условие*

$$h(x)^q\equiv h(x)\pmod{f(x)}.$$

Тогда справедливо равенство

$$f(x)=\prod_{c\in GF(q)} \text{НОД}(f(x), h(x)-c), \tag{6.7}$$

причем правая часть этого равенства представляет собой нетривиальное разложение $f(x)$ на взаимно простые множители.

Доказательство. Поскольку при $c_1, c_2\in GF(q)$, $c_1\neq c_2$, многочлены $h(x)-c_1$ и $h(x)-c_2$ взаимно просты, то множители в правой части (6.7) взаимно просты. Поскольку $\deg(h(x)-c)<n$, то из выполнения равенства (6.7) следует, что это нетривиальное разложение

$f(x)$ (т. е. (6.7) есть разложение $f(x)$ на несколько многочленов меньшей степени). Само же равенство (6.7) следует теперь из того, что $h(x)^q - h(x) = \prod\limits_{c \in GF(q)} (h(x) - c) \equiv 0 \pmod{f(x)}$, и из взаимной простоты многочленов $h(x) - c$ при различных $c \in GF(q)$. $\square$

Определение 6.10. Многочлен $h(x) \in GF(q)[x]$ такой, что $1 \leqslant \leqslant \deg h(x) < n$, $h(x)^q \equiv h(x) \pmod{f(x)}$, называется f-разлагающим многочленом.

Из теоремы 6.9 следует, что если мы построим какой-нибудь f-разлагающий многочлен, то сможем с его помощью разложить $f(x)$ на не менее чем два множителя. Опишем процедуру построения таких многочленов.

Теорема 6.11. *Определим матрицу* $B = \|b_{ij}\|_{i,j=0,\ldots,n-1}$ *из сравнений*

$$x^{iq} \equiv \sum_{j=0}^{n-1} b_{ij} x^i \pmod{f(x)}, \quad i = 0, \ldots, n-1.$$

Многочлен $h(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1} \in GF(q)[x]$ *будет удовлетворять условию* $h(x)^q \equiv h(x) \pmod{f(x)}$ *тогда и только тогда, когда*

$$(a_0, \ldots, a_{n-1})B = (a_0, \ldots, a_{n-1}),$$

т. е. тогда и только тогда, когда вектор его коэффициентов является левым собственным вектором матрицы B *с собственным значением* 1.

Доказательство. Условие $h(x)^q \equiv h(x) \pmod{f(x)}$ равносильно условию

$$\sum_{i=0}^{n-1} a_i x^{iq} \equiv \sum_{i=0}^{n-1} a_i x^i \pmod{f(x)}.$$

Отсюда по определению матрицы B получим равенство

$$\sum_{i=0}^{n-1} a_i \sum_{j=0}^{n-1} b_{ij} x^j = \sum_{i=0}^{n-1} a_i x^i,$$

поэтому

$$\sum_{i=0}^{n-1} b_{ij} = a_j, \quad j = 0, \ldots, n-1,$$

что равносильно утверждению теоремы. $\square$

Алгоритм Берлекэмпа.

На входе алгоритма задан унитарный многочлен $f(x) \in GF(q)[x]$, $\deg f(x) = n \geqslant 2$, про который известно, что он не имеет кратных неприводимых множителей. На выходе — разложение $f(x)$ на неприводимые множители.

1 шаг. Вычислить матрицу B из теоремы 6.11.

2 шаг. Найти базис пространства решений системы линейных уравнений

$$B_1 \begin{pmatrix} x_0 \\ \dots \\ x_{n-1} \end{pmatrix} = 0, \tag{6.8}$$

где $B_1 = (B - I_n)^T$, I_n — единичная матрица, знак $(\cdot)^T$ означает транспонирование. Пусть $\overline{e}_1 = (1, 0, \dots, 0)$, $\overline{e}_2, \dots, \overline{e}_k$ — найденный базис.

Замечание 6.12. Поскольку $x^{iq} \equiv 1 \pmod{f(x)}$ при $i = 0$, то в матрице B первая строка всегда имеет вид $(1, 0, \dots, 0)$, и первый столбец матрицы B_1 будет нулевым. Поэтому $\overline{e}_1 = (1, 0, \dots, 0)$ будет присутствовать в базисе пространства решений.

3 шаг. При $k = 1$ многочлен $f(x)$ неприводим; вообще, найденное значение k равно количеству неприводимых делителей $f(x)$ в $GF(q)[x]$. При $k > 1$ надо взять $\overline{e}_2 = (h_{2,0}, \dots, h_{2,n-1})$ и построить f — разлагающий многочлен $h_2(x) = \sum\limits_{i=0}^{n-1} h_{2,i} x^i$. С его помощью, применяя теорему 6.9, т. е. вычисляя НОД$(f(x), h_2(x) - c)$ при $c \in GF(q)$, найти разложение

$$f(x) = g_1(x) \dots g_l(x),$$

где $g_i(x) \in GF(q)[x]$, $l \geqslant 2$. Если $l = k$, то алгоритм останавливается. Если $l < k$, то мы берем $\overline{e}_3 = (h_{30} \dots, h_{3,n-1})$, строим $h_3(x) = \sum\limits_{i=0}^{n-1} h_{3i} x^i$; вычисляя НОД$(g_i(x), h_3(x) - c)$ для найденных $g_i(x)$, мы получаем дальнейшее разложение $f(x)$, и т. д. Алгоритм закончит работу, когда мы переберем все базисные векторы $\overline{e}_2, \dots, \overline{e}_k$, и для соответствующих им многочленов $h_i(x)$ вычислим наибольшие общие делители найденных множителей $f(x)$ с $h_i(x) - c$, $c \in GF(q)$.

Замечание 6.13. Алгоритм останавливается, как только мы найдем разложение $f(x)$ на k множителей, где $k = n - \operatorname{rank} B_1$.

Конец алгоритма.

Теорема 6.14. *Алгоритм Берлекэмпа работает корректно и находит полное разложение $f(x)$ на множители.*

Доказательство. Поскольку $f(x) = f_1(x) \dots f_k(x)$ — произведение различных унитарных неприводимых многочленов, то по китайской теореме об остатках для любого набора $c_1, \dots, c_k \in GF(q)$ существует единственный многочлен $h(x) \in GF(q)[x]$ такой, что $h(x) \equiv c_i \pmod{f_i(x)}$, $i = 1, \dots, k$, $\deg h(x) < n$. Этот многочлен $h(x)$ удовлетворяет условию

$$h(x)^q \equiv c_i^q \equiv c_i \equiv h(x) \pmod{f_i(x)}, \quad i = 1, \dots, k,$$

и поэтому является f-разлагающим. Обратно, любой f-разлагающий многочлен $h(x)$ удовлетворяет соотношению

$$h^q(x) - h(x) \equiv \prod_{c \in GF(q)} (h(x) - c) \equiv 0 \pmod{f(x)},$$

откуда для каждого $f_i(x)$ найдется $c = c_i$ такое, что $h(x) \equiv c_i \pmod{f_i(x)}$. Таким образом, мы получили взаимно однозначное соответствие между f-разлагающими многочленами $h(x)$ и наборами $c_i, \dots, c_k \in GF(q)$. Поскольку по теореме 6.11 вектор коэффициентов $(h_0, \dots, h_{n-1})$ многочлена $h(x)$ удовлетворяет системе уравнений (6.8), то ранг матрицы B_1 равен $n - k$, и пространство решений k-мерно. Обозначим через $h_1(x), \dots, h_k(x)$ f-разлагающие многочлены, векторы коэффициентов которых образуют базис пространства решений системы (6.8); мы считаем, что $h_1(x) = 1$. Пусть $f_i(x)$ и $f_j(x)$ — какие-либо два неприводимых делителя $f(x)$, $i \neq j$. Мы хотим показать, что найдется номер u, $2 \leqslant u \leqslant k$, и постоянная $c \in GF(q)$ такая, что

$$h_u(x) \equiv c \pmod{f_i(x)}, \quad h_u(x) \not\equiv c \pmod{f_j(x)}.$$

Предположим противное, т. е. для всех $u = 2, \dots, k$ при некоторых $c_u \in GF(q)$ выполнены сравнения $h_u(x) \equiv c_u \pmod{f_i(x) \cdot f_j(x)}$. То же верно и для $u = 1$, так как $h_1(x) = 1 = c_1$. Но тогда, поскольку любой f-разлагающий многочлен $h(x)$ является линейной комбинацией над $GF(q)$ многочленов $h_1(x), \dots, h_k(x)$, то

$$h(x) \equiv c \pmod{f_i(x) \cdot f_j(x)},$$

где $c \in GF(q)$, c зависит от $h(x)$. Однако найдется f-разлагающий многочлен $h(x)$ такой, что

$$h(x) \equiv 0 \pmod{f_i(x)}, \quad h(x) \equiv 1 \pmod{f_j(x)}.$$

Полученное противоречие завершает доказательство теоремы и обоснование алгоритма Берлекэмпа. □

Замечание 6.15. Сложность алгоритма составляет $O(n^3 + qkn)$ арифметических операций (см. [22, с. 191]). Поэтому алгоритм будет

эффективен лишь для сравнительно небольших значений q (в частности, из-за перебора $c \in GF(q)$ на 3 шаге алгоритма); т. е. для небольших полей.

В последующих параграфах мы рассмотрим ряд усовершенствований алгоритма Берлекэмпа, а также методы факторизации в случае больших полей $GF(q)$.

Приведем еще два результата, связанных с факторизацией многочленов из $GF(q)[x]$. В работе [197] приведен следующий результат, позволяющий установить четность количества неприводимых делителей $f(x)$.

Теорема 6.16. *Пусть p — простое число, $f(x) \in \mathbb{Z}/p\mathbb{Z}$, $\deg f(x) = n < p$. Пусть D — дискриминант $f(x)$, w — число неприводимых множителей $f(x)$, и пусть $p \nmid D$. Тогда* $\left(\dfrac{D}{p}\right) = (-1)^{n-w}$.

Другой результат о f-разлагающих многочленах приведен в [31, теорема 4.3].

Теорема 6.17. *Пусть $f(x) \in GF(q)[x]$ — унитарный многочлен, $f(x) = f_1(x) \dots f_k(x)$ — разложение $f(x)$ на различные унитарные неприводимые многочлены, где $k \geqslant 2$. Пусть $\deg f_i(x) = n_i$, $i = 1, \dots, k$, $N = \mathrm{HOK}(n_1, \dots, n_k)$. Положим*

$$T(x) = x + x^q + x^{q^2} + \dots + x^{q^{N-1}},$$

и определим многочлены $T_i(x) \in GF(q)[x]$:

$$T_i(x) \equiv T(x^i) \pmod{f(x)}, \quad \deg T_i(x) < n, \quad i = 1, 2, \dots$$

Тогда по крайней мере один из многочленов $T_1(x), \dots, T_{n-1}(x)$ является f-разлагающим.

§ 6.4. Метод Кантора—Цассенхауза

Зафиксируем простое число p. Пусть $f(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $f(x)$ унитарен и не имеет кратных неприводимых множителей. Ниже мы опишем алгоритм, который для заданного $d \in \mathbb{N}$ находит произведение всех неприводимых делителей $f(x)$, имеющих степень d. Предварительно докажем лемму.

Лемма 6.18. *Пусть $g(x) \in \mathbb{Z}/p\mathbb{Z}[x]$ — унитарный неприводимый многочлен.*

1. *Если $\deg g(x) = d$, то $g(x)$ делит $x^{p^d} - x$.*

2. *Если $g(x)$ делит $x^{p^d} - x$ и $g(x)$ не делит $x^{p^e} - x$ для всех $e < d$, то $\deg g(x) = d$.*

Доказательство. Первое утверждение следует из того, что $\mathbb{K} = \mathbb{Z}/p\mathbb{Z}[x]/(g(x)) = GF(p^d)$, и элемент $\overline{x} \equiv x \pmod{g(x)} \in \mathbb{K}$ удовлетворяет соотношению $\overline{x}^{p^d} = \overline{x}$. Для доказательства второго утверждения обозначим $l = \deg g(x)$, $\mathbb{K}_1 = GF(p^l) = \mathbb{Z}/p\mathbb{Z}[x]/(g(x))$. Поскольку $x^{p^l} \equiv x \pmod{g(x)}$, то из условия утверждения следует, что $l \geqslant d$. С другой стороны, для произвольного многочлена $f(x) \in \mathbb{Z}/p\mathbb{Z}[x]$ выполнены соотношения

$$f(x)^{p^d} = f(x^{p^d}) \equiv f(x) \pmod{g(x)}. \tag{6.9}$$

Поскольку $\mathbb{K}_1^*$ является циклической группой порядка $p^l - 1$ и порождается элементом $f(x) \pmod{g(x)}$ для некоторого $f(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, то из (6.9) следует, что $p^l - 1 \mid p^d - 1$, т. е. $l \leqslant d$. Значит, $l = d$, что и требовалось доказать. □

Применим лемму 6.18. Рассмотрим

$$B_1(x) = \text{НОД}(f(x),\ x^p - x).$$

Очевидно, $B_1(x)$ состоит из всех неприводимых делителей $f(x)$ первой степени. Заменим $f(x)$ на $f_1(x) = f(x)/B_1(x)$. Тогда вычислим

$$B_2(x) = \text{НОД}(f_1(x),\ x^{p^2} - x).$$

$B_2(x)$ состоит из всех неприводимых делителей $f(x)$ второй степени. Положим $f_2(x) = f_1(x)/B_2(x)$. Продолжая этот процесс, мы получим, что

$$B_d(x) = \text{НОД}(f_{d-1}(x),\ x^{p^d} - x)$$

состоит из всех неприводимых делителей $f(x)$, имеющих степень d; далее $f_d(x) = f_{d-1}(x)/B_d(x)$, и т. д.

Замечание 6.19. Этот алгоритм нахождения произведений всех неприводимых делителей $f(x)$, имеющих фиксированную степень, можно применять до того, как мы применяем алгоритм Берлекэмпа для факторизации $f(x)$.

Лемма 6.20. *Пусть $g(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $g(x)$ унитарен, $\deg g(x) = d$, причем все неприводимые делители $g(x)$ имеют одинаковую степень и $g(x)$ не имеет кратных делителей. Многочлен $g(x)$ неприводим тогда и только тогда, когда $x^{p^d} \equiv x \pmod{g(x)}$ и для любого простого q, делящего d,*

$$\text{НОД}\left(x^{p^{d/q}} - x,\ g(x)\right) = 1.$$

Доказательство. Необходимость очевидна. Докажем достаточность. Если $g(x)$ приводим и имеет неприводимый делитель $g_1(x)$, $\deg g_1(x) = l < d$, то $g_1(x) \mid x^{p^l} - x$. Кроме того, по условию $l \mid d$, $l = d/k$.

Если k простое, то мы получили противоречие условию леммы. Если $k = qr$, где $r \geqslant 2$, и q — простое, то мы также получили противоречие, поскольку $g_1(x) \mid x^{p^l} - x \mid x^{p^{lr}} - x = x^{d/q} - x$. $\square$

Далее мы считаем, что $f(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $f(x)$ бесквадратен, и все его неприводимые делители имеют одинаковую степень d, известную нам. Сейчас мы опишем метод Кантора—Цассенхауза для факторизации $f(x)$ (см. [85]). В нем различаются случаи $p = 2$ и $p > 2$.

Теорема 6.21. *Если $p > 2$, то для любого многочлена $T = T(x) \in \mathbb{Z}/p\mathbb{Z}[x]$ справедливо равенство*

$$f(x) = \text{НОД}(f(x),\, T)\,\text{НОД}(f(x),\, T^{(p^d-1)/2} + 1)\,\text{НОД}(f(x),\, T^{(p^d-1)/2} - 1).$$

Доказательство. Все элементы поля $GF(p^d)$ представляют собой корни многочлена $x^{p^d} - x$. Пусть $T = T(x) \in \mathbb{Z}/p\mathbb{Z}[x]$. Тогда многочлен $T(x)^{p^d} - T(x) = T(x^{p^d}) - T(x)$ делится на $x^{p^d} - x$. Из леммы 6.18 следует, что $f(x)$ делит $T^{p^d} - T = T(T^{(p^d-1)/2} - 1)(T^{(p^d-1)/2} + 1)$, причем поскольку $p > 2$, то многочлены T, $T^{(p^d-1)/2} - 1$ и $T^{(p^d-1)/2} + 1$ взаимно просты. Теорема доказана. $\square$

Замечание 6.22. Можно доказать, что для случайно выбранного многочлена $T \in \mathbb{Z}/p\mathbb{Z}[x]$ такого, что $\deg T \leqslant 2d - 1$, наибольший общий делитель $D = \text{НОД}(f(x),\, T^{(p^d-1)/2} - 1)$ нетривиален с вероятностью, близкой к $1/2$, см. [89, гл. 3, § 3.4.4]. Факторизация $f(x)$ методом Кантора—Цассенхауза как раз состоит в случайном выборе T и вычислении D. Если $\deg D = 0$ или $\deg D = \deg f(x)$, то выбирается другой многочлен T; иначе мы факторизуем D и $f(x)/D$ тем же способом. При этом найденные делители $f(x)$, имеющие степень d, являются по условию неприводимыми.

Теперь рассмотрим случай $p = 2$.

Теорема 6.23. *Пусть $p = 2$, $U(x) = x + x^2 + x^4 + \ldots + x^{2^{d-1}} \in \mathbb{Z}/2\mathbb{Z}[x]$. Тогда для любого многочлена $T = T(x) \in \mathbb{Z}/2\mathbb{Z}[x]$ справедливо равенство*

$$f(x) = \text{НОД}(f(x),\, U(T))\,\text{НОД}(f(x),\, U(T) + 1).$$

Доказательство. Поскольку

$$U(T) = T + T^2 + \ldots + T^{2^{d-1}},$$

$$U(T)^2 = U(T^2) = T^2 + T^4 + \ldots + T^{2^d},$$

то

$$U(T) + U(T)^2 = T^{2^d} + T = T^{2^d} - T.$$

Дальнейшие рассуждения аналогичны доказательству теоремы 6.21. $\square$

Как и в случае $p > 2$, можно доказать, что для случайно выбранного многочлена $T = T(x)$, $\deg T \leqslant 2d-1$, наибольший общий делитель $f(x)$ и $U(T)$ будет нетривиален с вероятностью, примерно равной $\frac{1}{2}$. Также можно выбирать $T(x)$ среди степеней x, x^3, …, x^{2d-1}, т. е. при некотором нечетном j, $1 \leqslant j \leqslant 2d-1$, наибольший общий делитель $(U(x^j), f(x))$ будет нетривиален с вероятностью, примерно равной $\frac{1}{2}$. Докажем это. Предположим противное, т. е., что для любого нечетного j, $1 \leqslant j \leqslant 2d-1$, либо $U(x^j) \equiv 0 \pmod{f(x)}$, либо $U(x^j \equiv 1 \pmod{f(x)}$ (использовалась теорема 6.23). Также из теоремы 6.23 следует, что $U(T^2) = U(T)^2 \equiv U(T) \pmod{f(x)}$. Поэтому для всех j, $1 \leqslant j \leqslant 2d-1$ выполнено сравнение $U(x^j) \equiv 0 \pmod{f(x)}$ или сравнение $U(x^j) \equiv 1 \pmod{f(x)}$. Так как $U(T_1 + T_2) = U(T_1) + U(T_2)$, то для любого многочлена $H = H(x) \in \mathbb{Z}/2\mathbb{Z}[x]$, $\deg H(x) \leqslant 2d-1$, выполнено сравнение $U(H) \equiv 0 \pmod{f(x)}$ или $U(H) \equiv 1 \pmod{f(x)}$. Поскольку мы считаем, что $\deg f(x) > d$ (если $\deg f(x) = d$, то $f(x)$ неприводим), то у $f(x)$ есть два различных неприводимых делителя $f_1(x)$ и $f_2(x)$, $\deg f_1(x) = \deg f_2(x) = d$. Пусть α — корень $f_2(x)$ в $GF(2^d)$. Поскольку $f_2(x)$ неприводим, то $GF(2^d) = \mathbb{Z}/2\mathbb{Z}[\alpha]$. Так как $\deg U(x) = 2^{d-1}$, то существует $\beta \in GF(2^d)$, $U(\beta) \neq 0$. Для некоторого $P(x) \in \mathbb{Z}/2\mathbb{Z}[x]$, $\deg P(x) \leqslant d-1$, справедливо равенство $\beta = P(\alpha)$. По китайской теореме об остатках построим многочлен $T = T(x) \in \mathbb{Z}/2\mathbb{Z}[x]$ такой, что $\deg T(x) \leqslant 2d-1$, $T(x) \equiv 0 \pmod{f_1(x)}$, $T(x) \equiv P(x) \pmod{f_2(x)}$. Тогда $U(T) \equiv U(0) \equiv 0 \pmod{f_1(x)}$, $U(T) \equiv U(P(x)) \not\equiv 0 \pmod{f_2(x)}$, так как $U(\beta) = U(P(\alpha)) \neq 0$. Однако это противоречит тому, что $U(T) \equiv 0 \pmod{f(x)}$ или $U(T) \equiv 1 \pmod{f(x)}$. Полученное противоречие доказывает, что при некотором нечетном j, $1 \leqslant j \leqslant 2d-1$, наибольший общий делитель $(U(x^j), f(x))$ будет нетривиален.

§ 6.5. Некоторые другие усовершенствования алгоритма Берлекэмпа

Вернемся вновь к алгоритму Берлекэмпа, описанному в § 6.3. Ограничимся случаем простого поля $\mathbb{Z}/p\mathbb{Z}$.

Пусть p велико. Тогда перебор значений $c \in \mathbb{Z}/p\mathbb{Z}$ и вычисление НОД$(f(x), h(x) - c)$ займут много времени. Мы хотим определить множество $M \subset \mathbb{Z}/p\mathbb{Z}$, состоящее из тех c, для которых НОД$(f(x), h(x) - c)$ нетривиален.

Первый подход к нахождению множества M заключается в вычислении результанта $\operatorname{Res}(f(x), h(x)-c)$. Определение и свойства результанта можно найти в [9, гл. 5]. Если $f(x)=\sum\limits_{i=0}^{n} A_i x^i \in \mathbb{Z}/p\mathbb{Z}[x]$, и $h(x)=\sum\limits_{i=0}^{m} B_i x^i \in \mathbb{Z}/p\mathbb{Z}[x]$, то $\operatorname{Res}(f(x), h(x)-c)$ равен определителю матрицы R размера $(n+m)\times(n+m)$, где

$$R=\begin{pmatrix} A_n & A_{n-1} & \dots & A_0 & 0 & \dots & 0 \\ 0 & A_n & \dots & A_1 & A_0 & 0 \ \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & A_n & A_{n-1} & \dots & A_0 \\ B_m & B_{m-1} & \dots & B_0-c & 0 & \dots & 0 \\ 0 & B_m & \dots & B_1 & B_0-c & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & B_m & \dots & B_1 & B_0-c \end{pmatrix}$$

Мы видим, что $\operatorname{Res}(f(x), h(x)-c)$ является многочленом степени n от переменной c с коэффициентами из $\mathbb{Z}/p\mathbb{Z}$. Его корни являются в точности теми значениями c, для которых $\operatorname{НОД}(f(x), h(x)-c)$ нетривиален. Поэтому для определения множества M мы находим корни $\operatorname{Res}(f(x), h(x)-c)$, как многочлена от c, лежащие в поле $\mathbb{Z}/p\mathbb{Z}$; мы можем находить их, например, с помощью вероятностного алгоритма, описанного в § 6.1.

Другой подход к отысканию множества M был предложен Цассенхаузом. Сохраняя введенные выше обозначения, мы предположим дополнительно, что $f(x)$ унитарен, $m<n$, и что $h(x)^p \equiv h(x) \pmod{f(x)}$. Тогда

$$f(x)=\prod_{c\in M} \operatorname{НОД}(f(x), h(x)-c). \tag{6.10}$$

Обозначим $G(y)=\prod\limits_{c\in M}(y-c)\in \mathbb{Z}/p\mathbb{Z}[x]$.

Теорема 6.24. *Многочлен $f(x)$ делит $G(h(x))$; при этом $G(y)$ — многочлен минимальной степени среди всех многочленов $g(y)\in \mathbb{Z}/p\mathbb{Z}[y]\setminus\{0\}$ таких, что $f(x)$ делит $g(h(x))$.*

Доказательство. Обозначим через I идеал кольца $\mathbb{Z}/p\mathbb{Z}[y]$, состоящий из многочленов $g(y)$ таких, что $f(x)$ делит $g(h(x))$. По определению $G(y)\in I$. Кроме того, идеал I является главным, $I=(g_0(y))$ для некоторого $g_0(y)\in \mathbb{Z}/p\mathbb{Z}[y]\setminus\{0\}$. Из определения множества M

и равенства (6.10) следует, что $M' = M$. Поскольку $g_0(y) \mid G(y)$, то $g_0(y) = \prod_{c \in M'} (y - c)$ для некоторого подмножества $M' \subset M$. $\square$

Применим теорему 6.24 для нахождения M. Пусть $|M| = k$ — неизвестное нам число. Тогда $G(y) = \sum_{j=0}^{k} b_j y^j$, где $b_k = 1$, причем число m — минимальное натуральное число, обладающее тем свойством, что многочлены $1, h(x), \ldots, h(x)^k$ — линейно зависимы над $\mathbb{Z}/p\mathbb{Z}$ по модулю многочлена $f(x)$. Тогда мы находим векторы коэффициентов многочленов $h(x)^0 \pmod{f(x)} = 1 = \sum_{j=0}^{n-1} h_{0j} x^j$, $h(x) \pmod{f(x)} = \sum_{j=0}^{n-1} h_{1j} x^j$, $\ldots$ $\ldots, h(x)^i \pmod{f(x)} = \sum_{j=0}^{n-1} h_{ij} x^j$, $\ldots$, и берем первый номер k, такой, что вектор $(h_{k,0}, \ldots, h_{k,n-1})$ линейно выражается над $\mathbb{Z}/p\mathbb{Z}$ через $(h_{0,0}, \ldots, h_{0,n-1}), \ldots, (h_{k-1,0}, \ldots, h_{k-1,n-1})$. Это дает нам соотношение

$$\sum_{j=0}^{k} b_j h(x)^j \equiv 0 \pmod{f(x)}, \qquad b_k = 1,$$

с минимальным значением k. Тогда мы полагаем $G(y) = \sum_{j=0}^{k} b_j y^j$. Множество M будет состоять из всех корней многочлена $G(y)$; мы находим эти корни, например, вероятностными методами из § 6.1.

Еще один метод разложения унитарного многочлена $f(x)$ из кольца $GF(q)[x]$, не имеющего кратных неприводимых множителей, описан в [31, гл. 4, § 2]. Он основан на приведении некоторой эффективно вычислимой с помощью алгоритма Берлекэмпа матрицы A к диагональному виду. Однако сам процесс диагонализации довольно сложен.

В работе [139] предложена вероятностная версия алгоритма Берлекэмпа. С ее помощью можно разложить на множители многочлен из $GF(q)[x]$ степени n в среднем за $O((n \log n + \log q) n (\log n)^2 \log\log n)$ арифметических операций в поле $GF(q)$. В алгоритме был использован метод Видемана для решения систем линейных уравнений над конечными полями (см. об этом гл. 11 далее). Алгоритм Калтофена—Лобо был реализован на компьютере. Многочлены небольшой степени (скажем, до 250) быстрее раскладывать на множители с помощью обычного алгоритма Берлекэмпа, если рассматриваемые поля не слишком велики. Для многочленов высокой степени алгоритм Калтофена—Лобо в ряде случаев оказался эффективным; например, для многочленов степени 10 001 над полем $\mathbb{Z}/127\mathbb{Z}$ он работает около 102,5 часов на компьютере Sun 4.

§ 6.6. Вероятностный алгоритм проверки неприводимости многочленов над конечными полями

Пусть p — простое число, $p>2$, $r\in\mathbb{N}$, $r\geqslant 1$, $q=p^r$, $\mathbb{K}=GF(q)$. Пусть $f=f(x)$ — фиксированный унитарный многочлен из $\mathbb{K}[x]$, $n=\deg f(x)\geqslant 2$. В данном параграфе мы опишем вероятностный алгоритм проверки неприводимости $f(x)$, аналогичный алгоритму Миллера—Рабина для проверки простоты чисел (см. § 1.7).

Обозначим $q^n-1=2^k\cdot t$, где t нечетно, $R=\mathbb{K}[x]/(f)$ — кольцо из q^n элементов. Если $f(x)$ неприводим, то R является полем, $|R^*|=q^n-1$, и для любого $a\in R\setminus\{0\}$ выполнено равенство $a^{q^n-1}=a^{2^k t}=1$. Извлекая квадратный корень, мы получим, что

$$\text{либо } a^t=1,\quad \text{либо } a^{t\cdot 2^j}=-1 \text{ при некотором } j,\quad 0\leqslant j<k. \tag{6.11}$$

Для многочлена $f(x)$ мы обозначим через S множество элементов $a\in R$, для которых выполнено условие (6.11).

Теорема 6.25. *Если $f(x)$ приводим, то*

$$|S|\leqslant\frac{1}{2}|R^*|\leqslant\frac{q^n-1}{2}.$$

Следствие 6.26. *Проверяя условие (6.11) для случайно выбранного $a\in R$, мы с высокой вероятностью обнаружим, что $f(x)$ приводим. Если же условие (6.11) выполнено для l случайно выбранных элементов $a\in R$, то можно считать, что $f(x)$ неприводим с вероятностью не меньшей, чем $1-\dfrac{1}{2^l}$.*

Доказательство теоремы разбивается на два случая.

1 случай. Пусть существует неприводимый многочлен $g=g(x)\in\mathbb{K}[x]$ такой, что $g^2\mid f$. Зафиксируем его и обозначим через G множество

$$G=\left\{1+h(x)\frac{f(x)}{g(x)}\pmod{f(x)}\;\middle|\;h(x)\in\mathbb{K}[x],\ \deg h(x)<\deg g\right\}.$$

Лемма 6.27. *Справедливы следующие утверждения*:

1. *G — подгруппа в R^*;*
2. *для любого $h\in G\setminus 1$ порядок h равен p;*
3. *$|G|=q^m$, где $m=\deg g(x)$.*

Доказательство. Поскольку

$$\left(1+h_1\frac{f}{g}\right)\left(1+h_2\frac{f}{g}\right)\equiv 1+(h_1+h_2)\frac{f}{g}\pmod{f},$$

то первое утверждение очевидно. Если $w \in G \setminus 1$, $w = 1 + \dfrac{f}{g}$, то $w^p = \left(1 + h\dfrac{f}{g}\right)^p = 1 + h^p f \dfrac{f^{p-2}}{g^{p-2}} \dfrac{f}{g^2} \equiv 1 \pmod f$. Третье утверждение леммы также очевидно. □

Лемма 6.28. *Если $w \in G \setminus 1$, то $wS \cap S = \varnothing$.*

Доказательство. По лемме 6.27 порядок w равен p, поэтому $w^{q^n-1} \neq 1$. Если $a \in S$, то $a^{q^n-1} = 1$, откуда $(aw)^{q^n-1} \neq 1$. Значит, $aw \notin S$, что и требовалось доказать. □

Лемма 6.29. *Пусть a, b, $\in G$, $a \neq b$. Тогда $aS \cap bS = \varnothing$.*

Доказательство. $aS \cap bS = \varnothing$ тогда и только тогда, когда $S \cap a^{-1}bS = \varnothing$, а последнее равенство верно по лемме 6.28. □

Лемма 6.30. *Справедливо неравенство*

$$|S| \leqslant |R^*|/q^m,$$

где $m = \deg g$.

Доказательство. Обозначим через $w_1 = 1, w_2, \ldots, w_{q^m}$ все элементы $G \subseteq R^*$. Тогда множества $w_1 S, \ldots, w_{q^m} S$ содержатся в R^* и не пересекаются по лемме 6.29. Поэтому $|R^*| \geqslant q^m |S|$, что и требовалось доказать. □

Из леммы 6.30 следует утверждение теоремы 6.25 в рассматриваемом нами случае.

2 случай. Пусть многочлен f бесквадратен, т. е. $f = f_1 \ldots f_l$, где f_i — различные неприводимые многочлены из $\mathbb{K}[x]$, и $l \geqslant 2$.

Обозначим $d_i = \deg f_i$, $R_i = \mathbb{K}[x]/(f_i) = GF(q^{d_i})$, $i = 1, \ldots, l$. По китайской теореме об остатках, R изоморфно $R_1 \times \ldots \times R_l$; для элемента $a \in R$ мы будем обозначать $a_i \equiv a \pmod{f_i} \in R_i$, $i = 1, \ldots, l$. Если $m \in \mathbb{N}$, то $a^m \equiv 1$ в R тогда и только тогда, когда $a_i^m = 1$ в R_i, $i = 1, \ldots, l$. Поэтому

$$\#\{a \in R \mid a^m = 1\} = \prod_{i=1}^{l} \#\{a \in R_i \mid a^m = 1\} = \prod_{i=1}^{l} \text{НОД}(m, q^{d_i} - 1),$$

так как R_i^* — циклические группы порядка $q^{d_i} - 1$. Также

$$\#\{a \in R \mid a^m = -1\} = \prod_{i=1}^{l} \#\{a \in R_i \mid a^m = -1\} =$$

$$= \prod_{i=1}^{l} (\#\{a \in R_i \mid a^{2m} = 1\} - \#\{a \in R_i \mid a^m = 1\}) =$$

$$= \prod_{i=1}^{l} (\text{НОД}(2m, q^{d_i} - 1) - \text{НОД}(m, q^{d_i} - 1)).$$

Поэтому $S = S_1 \cup S_2$, где

$$S_1 = \{a \in R \mid a^t = 1\}, \quad |S_1| = \prod_{i=1}^{l} \text{НОД}(t,\, q^{d_i} - 1),$$

$$S_2 = \{a \in R \mid a^{t\cdot 2^{k-j}} = -1 \text{ при некотором } j,\ 1 \leqslant j \leqslant k\},$$

$$|S_2| = \sum_{j=1}^{k} \prod_{i=1}^{l} (\text{НОД}(t2^{k-j+1},\, q^{d_i} - 1) - \text{НОД}(t2^{k-j},\, q^{d_i} - 1)).$$

Пусть

$$q^{d_i} - 1 = 2^{c_i} b_i, \quad i = 1, \ldots, l,$$

где b_i — нечетные числа. Очевидно, что при $e \in \mathbb{Z}_{\geqslant 0}$

$$\text{НОД}(2^e t,\, 2^{c_i} b_i) = 2^{\min(e, c_i)}\, \text{НОД}(t,\, b_i),$$

$$\text{НОД}(2^{e+1} t,\, 2^{c_i} b_i) - \text{НОД}(2^e t,\, 2^{c_i} b_i) = \begin{cases} 2^e\, \text{НОД}(t,\, b_i) & \text{при } e < c_i, \\ 0 & \text{при } e \geqslant c_i. \end{cases}$$

Поэтому при $c = \min(c_1, \ldots, c_l)$ и при $v = \min(c,\, k)$ выполнено равенство

$$|S_2| = \sum_{j=0}^{v-1} \prod_{i=1}^{l} 2^j\, \text{НОД}(t,\, b_i).$$

Следовательно,

$$|S| = |S_1| + |S_2| = \prod_{i=1}^{l} \text{НОД}(t, b_i) \left(1 + \sum_{j=0}^{v-1} 2^{lj}\right) = \prod_{i=1}^{l} \text{НОД}(t, b_i) \left(1 + \frac{2^{vl} - 1}{2^l - 1}\right).$$

Далее,

$$\text{НОД}(q^n - 1,\, q^{d_i} - 1) = 2^{\min(k, c_i)}\, \text{НОД}(t,\, b_i),$$

откуда

$$\prod_{i=1}^{l} \text{НОД}(t,\, b_i) = \prod_{i=1}^{l} \frac{\text{НОД}(q^n - 1,\, q^{d_i} - 1)}{2^{\min(k, c_i)}} \leqslant$$

$$\leqslant \prod_{i=1}^{l} (q^{d_i} - 1) \Big/ \prod_{i=1}^{l} 2^{\min(k, c_i)} = \frac{|R^*|}{\prod_{i=1}^{l} 2^{\min(k, c_i)}}.$$

Следовательно,

$$|S| \leqslant \left(1 + \frac{2^{vl} - 1}{2^l - 1}\right) \frac{|R^*|}{\prod_{i=1}^{l} 2^{\min(k, c_i)}} \leqslant \left(1 + \frac{2^{vl} - 1}{2^l - 1}\right) \frac{|R^*|}{2^{vl}}.$$

Нетрудно видеть, что справедливо неравенство

$$\left(1+\frac{2^{vl}-1}{2^l-1}\right)\Big/ 2^{vl} \leqslant \frac{1}{2^{l-1}},$$

поскольку $v \geqslant 1$. Так как по условию теоремы $l \geqslant 2$, то $|S| \leqslant |R^*|/2$. Теорема 6.25 доказана. □

Замечание 6.31. В ряде случаев для проверки неприводимости многочленов над конечным простым полем эффективен следующий метод (см. [101]). Пусть p — простое число, $f(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, $\deg f(x) = n$. Многочлен $f(x)$ неприводим тогда и только тогда, когда для любого k, $1 \leqslant k \leqslant n/2$, выполнено равенство

$$\text{НОД}(f(x),\ x^{p^k} - x) = 1. \tag{6.12}$$

Действительно, если $f(x)$ приводим, то у него найдется неприводимый делитель $g(x)$, $\deg g(x) = k \leqslant n/2$. Все корни $g(x)$ лежат в $GF(p^k)$, следовательно, являются корнями $x^{p^k} - x$. Поэтому для $k = \deg g(x)$ условие (6.12) не будет выполнено. Если же $f(x)$ неприводим, то его корни не будут являться корнями $x^{p^k} - x$ при $k < n$. Алгоритм проверки неприводимости $f(x)$ заключается в проверке условия (6.12) для всех $k \leqslant n/2$.

§ 6.7. Заключение

В книге [31, гл. 4, заключение] содержится превосходный обзор методов факторизации многочленов над конечными полями. Там же в виде обзора описаны различные методы реализации арифметики в конечных полях, арифметики многочленов над конечными полями и методы решения уравнений в конечных полях.

В гл. 3 книги [89] описан ряд алгоритмов для реализации полиномиальной арифметики над конечными полями и над кольцом целых чисел, а также алгоритмы факторизации многочленов.

Ряд полезных сведений о факторизации многочленов над конечными полями и решении алгебраических уравнений в $GF(q)$ можно найти в книге [60, гл. 7]. Там же описан детерминированный метод нахождения неприводимого многочлена над конечным полем, имеющий в предположении расширенной гипотезы Римана полиномиальную сложность, и метод решения кубических уравнений в $GF(q)$ за $O(\log^4 q)$ битовых операций (также в предположении расширенной гипотезы Римана). В этой книге содержится и отличный обзор результатов о факторизации многочленов и решении алгебраических уравнений в конечных полях.

В работе [272] была разработана новая алгоритмическая техника применительно к алгоритму Кантора—Цассенхауза. С ее помощью многочлен из $GF(q)[x]$, равный произведению различных неприводимых многочленов одинаковой степени, может быть разложен на неприводимые множители в среднем за

$$O(n^{(\omega+1)/2+o(1)} + n^{1+o(1)} \log q)$$

операций в $GF(q)$; здесь ω — показатель в оценке сложности умножения квадратных матриц. В работе [142] был получен аналогичный результат: для любого β, $0 \leqslant \beta \leqslant 1$ существует вероятностный алгоритм факторизации многочлена степени n из $GF(q)[x]$, делающий в среднем

$$O(n^{(\omega+1)/2+(1-\beta)(\omega-1)/2} + n^{1+\beta+o(1)} \log\ q)$$

арифметических операций в $GF(q)$.

Из вероятностных алгоритмов факторизации многочленов отметим также алгоритмы Бен-Ора, см. [62].

Новый подход к факторизации многочленов над конечными полями был разработан Нидеррайтером, см. [206; 122; 207]. С точки зрения сложности этот метод близок к оригинальному алгоритму Берлекэмпа.

В работе [253] предложен эффективный алгоритм факторизации многочленов в $GF(q)[x]$ со сложностью $O(n^{2,5} + n^{1+o(1)} \log q)$ операций в $GF(q)$, требующий памяти для $O(n^{3/2})$ элементов $GF(q)$. Оказалось, что на практике этот алгоритм при большом q является в ряде случаев более эффективным, чем все предыдущие методы.

В работах [250; 252] предложен ряд алгоритмов для нахождения неприводимых многочленов над конечными полями. Работа [249] посвящена вопросам сложности алгоритмов факторизации многочленов над конечными полями.

Работа [141] посвящена разложению многочленов над большими алгебраическими расширениями конечных полей.

В работе [137] приведен обзор алгоритмов факторизации многочленов.

Для решения систем алгебраических уравнений применяется либо алгоритм Лазара, либо методы, использующие базисы Грёбнера; см. об этом [157; 172; 100; 156; 155; 81; 125].

Отметим здесь также работу [168]. Она посвящена эффективному нахождению изоморфизма между двумя представлениями конечного поля и имеет важные приложения как к задаче дискретного логарифмирования, так и к вопросам факторизации многочленов над конечными полями.

Глава 7. Приведенные базисы решеток и их приложения

§ 7.1. Введение. Решетки и базисы

Пусть $\mathbb{R}^n$ — n-мерное евклидово пространство со скалярным произведением $(\cdot, \cdot)$. Пусть $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^n$ — линейно независимые вектора. *Решеткой* в $\mathbb{R}^n$ называется множество

$$\Lambda = \Lambda(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) = \mathbb{Z}\boldsymbol{b}_1 \oplus \ldots \oplus \mathbb{Z}\boldsymbol{b}_n = \{x_1\boldsymbol{b}_1 + \ldots + x_n\boldsymbol{b}_n \mid x_1, \ldots, x_n \in \mathbb{Z}\};$$

векторы $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ называются *базисом решетки*. *Определителем решетки* называется величина $d(\Lambda)$,

$$d(\Lambda) = (\det \|(\boldsymbol{b}_i, \boldsymbol{b}_j)\|_{i,j=1,\ldots,n})^{1/2}.$$

Базис решетки можно выбирать разными способами; однако поскольку один базис выражается через другой с помощью целочисленной матрицы с определителем ± 1, $d(\Lambda)$ является инвариантом решетки. Имеет место *неравенство Адамара*:

$$d(\Lambda) \leqslant \prod_{i=1}^{n} |\boldsymbol{b}_i|,$$

где $|\boldsymbol{b}|$ — евклидова норма вектора $\boldsymbol{b} \in \mathbb{R}^n$. Если $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ — ортонормированный базис $\mathbb{R}^n$, и $\boldsymbol{b}_i = \sum\limits_{j=1}^{n} b_{ij}\boldsymbol{e}_j$, $i = 1, \ldots, n$, то

$$d(\Lambda) = |\det \|b_{ij}\|_{i,j=1,\ldots,n}|.$$

Замечание 7.1. Превосходным введением в геометрию чисел является книга Касселса [24]. В ней можно найти описание различных свойств решеток и их приложений.

Решетка является *дискретным множеством*, т. е. в любом ограниченном множестве пространства $\mathbb{R}^n$ содержится лишь конечное число точек решетки.

Напомним *процесс ортогонализации Грама—Шмидта*. Пусть $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ — линейно независимые векторы в $\mathbb{R}^n$. Пусть векторы

$\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*$ определяются соотношениями

$$\boldsymbol{b}_1^* = \boldsymbol{b}_1, \quad \boldsymbol{b}_i^* = \boldsymbol{b}_i - \sum_{j=1}^{i-1} \mu_{ij} \boldsymbol{b}_j^*, \quad i = 2, \ldots, n,$$

где

$$\mu_{ij} = (\boldsymbol{b}_i, \boldsymbol{b}_j^*)/(\boldsymbol{b}_j^*, \boldsymbol{b}_j^*), \quad 1 \leqslant j < i.$$

Тогда векторы $\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*$ попарно ортогональны.

При применении решеток в различных теоретико-числовых алгоритмах важную роль играют базисы, состоящие из достаточно коротких векторов. Ниже мы рассмотрим различные виды таких базисов.

Фиксируем решетку Λ в $\mathbb{R}^n$ и рассмотрим множество G_Λ, состоящее из всех ее базисов. Рассмотрим на G_Λ следующее *отношение упорядоченности*: при $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\}, \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\} \in G_\Lambda$

$$\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\} < \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\},$$

если существует номер j такой, что при всех $i < j$ справедливо равенство $|\boldsymbol{a}_i| = |\boldsymbol{b}_i|$, но $|\boldsymbol{a}_j| < |\boldsymbol{b}_j|$.

Определение 7.2. Минимальный элемент G_Λ по отношению $<$ называется *приведенным по Минковскому базисом решетки*.

Поскольку норма на решетке принимает дискретное множество значений, то приведенный по Минковскому базис существует. Однако он может быть не единственен. Например, в решетке $\mathbb{Z}^n$ любая перестановка единичных векторов $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ образует приведенный по Минковскому базис.

Рассмотрим теперь другое отношение упорядоченности $<_T$. Пусть $\boldsymbol{a} = (x_1, \ldots, x_n)$, $\boldsymbol{b} = (y_1, \ldots, y_n,)$, $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$. Скажем, что $\boldsymbol{a} <_T \boldsymbol{b}$, если $|\boldsymbol{a}| < |\boldsymbol{b}|$, либо $|\boldsymbol{a}| = |\boldsymbol{b}|$ и существует номер j такой, что $x_i = y_i$ при $i < j$, но $x_j > y_j$ (т. е. у меньшего вектора первая отличная координата больше). Очевидно, что любые два вектора из $\mathbb{R}^n$ сравнимы по отношению $<_T$. Теперь продолжим это отношение $<_T$ на множество G_Λ: если $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\}, \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\} \in G_\Lambda$, то

$$\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\} <_T \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\},$$

если существует номер j такой, что $\boldsymbol{a}_i = \boldsymbol{b}_i$ при $i < j$, но $\boldsymbol{a}_j <_T \boldsymbol{b}_j$.

Определение 7.3. Минимальный элемент множества G_Λ по отношению $<_T$ называется *вполне приведенным базисом*.

Замечание 7.4. Легко видеть, что вполне приведенный базис существует и единственен. В [217, гл. 3] приведен алгоритм нахождения вполне приведенного базиса решетки.

Приведенные по Минковскому и вполне приведенные базисы решеток имеют важные приложения в вычислительной алгебраической теории чисел, см. [217, гл. 3, 5]. В теоретико-числовых алгоритмах часто используются базисы, приведенные по Ленстре—Ленстре—Ловасу (LLL-приведенные базисы), о которых мы расскажем в следующих параграфах.

§ 7.2. LLL-приведенный базис и его свойства

Определение 7.5. Базис $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ решетки $\Lambda \subseteq \mathbb{R}^n$ называется *LLL-приведенным*, если для векторов $\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*$ и коэффициентов μ_{ij}, полученных с помощью процесса ортогонализации Грама—Шмидта (см. § 7.1), выполнены неравенства:

$$|\mu_{ij}| \leqslant \frac{1}{2}, \quad 1 \leqslant j < i \leqslant n, \tag{7.1}$$

$$|\boldsymbol{b}_i^* + \mu_{ii-1}\boldsymbol{b}_{i-1}^*|^2 \geqslant \frac{3}{4}|\boldsymbol{b}_{i-1}^*|^2. \tag{7.2}$$

Замечание 7.6. LLL-приведенные базисы впервые были введены в работе [160]. Условие (7.2) означает, что вектор $\boldsymbol{b}_i^* + \mu_{ii-1}\boldsymbol{b}_{i-1}^*$, равный проекции вектора $\boldsymbol{b}_i$ на $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-2})^\perp$, не является очень маленьким по сравнению с проекцией вектора $\boldsymbol{b}_i$ на $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-2})^\perp$.

Опишем ряд важных свойств LLL-приведенных базисов.

Теорема 7.7. *Пусть $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ — LLL-приведенный базис решетки $\Lambda \subseteq \mathbb{R}^n$. Тогда*:

1. $|\boldsymbol{b}_j|^2 \leqslant 2^{i-1}|\boldsymbol{b}_i^*|^2$ *при всех* i, j, $1 \leqslant j < i \leqslant n$;
2. $d(\Lambda) \leqslant \prod\limits_{i=1}^{n} |\boldsymbol{b}_i| \leqslant 2^{n(n-1)/4} d(\Lambda)$;
3. $|\boldsymbol{b}_1| \leqslant 2^{(n-1)/4} d(\Lambda)^{1/n}$.

Доказательство. Поскольку векторы $\boldsymbol{b}_i^*$ ортогональны, то

$$|\boldsymbol{b}_i^* + \mu_{ii-1}\boldsymbol{b}_{i-1}^*|^2 = |\boldsymbol{b}_i^*|^2 + \mu_{ii-1}^2|\boldsymbol{b}_{i-1}^*|^2.$$

Так как $\mu_{ii-1}^2 \leqslant 1/4$, то из неравенства (7.2) следует, что

$$|\boldsymbol{b}_i^*|^2 \geqslant \frac{1}{2}|\boldsymbol{b}_{i-1}^*|^2.$$

Отсюда следует неравенство

$$|\boldsymbol{b}_j^*| \leqslant 2^{i-j}|\boldsymbol{b}_i^*|^2, \tag{7.3}$$

выполняющееся при всех $i \geqslant j$. В силу ортогональности $\boldsymbol{b}_i^*$, получим

$$|\boldsymbol{b}_i|^2 = |\boldsymbol{b}_i^*|^2 + \sum_{i<j} \mu_{ij}^2 |\boldsymbol{b}_j^*|^2 \leqslant |\boldsymbol{b}_i^*|^2 \left(1 + \sum_{1 \leqslant j < i} \frac{1}{4} 2^{i-j}\right) =$$

$$= |\boldsymbol{b}_i^*|^2 \left(1 + \frac{1}{2}(2^{i-1} - 1)\right) \leqslant 2^{i-1} |\boldsymbol{b}_i^*|^2.$$

Поэтому $|\boldsymbol{b}_j|^2 \leqslant 2^{i-1} |\boldsymbol{b}_j^*|^2 \leqslant 2^{j-1} \cdot 2^{i-j} |\boldsymbol{b}_i^*|^2$, что доказывает первое утверждение теоремы.

Первое неравенство во втором утверждении теоремы есть неравенство Адамара. Второе неравенство выполняется, поскольку

$$\prod_{i=1}^{n} |\boldsymbol{b}_i| \leqslant 2^{\sum_{i=1}^{n} \frac{i-1}{2}} \prod_{i=1}^{n} |\boldsymbol{b}_i^*| = 2^{\frac{n(n-1)}{2}} \cdot d(\Lambda).$$

Здесь мы воспользовались тем (обозначая через $[\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n]$ матрицу, в i-й строке которой стоят координаты вектора $\boldsymbol{b}_i$), что выполнены равенства

$$d(\Lambda) = \left|\det[\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n]\right| = \left|\det[\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*]\right| =$$

$$= \left|\det[\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*] \cdot [\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*]^T\right|^{1/2} = \left|\det \|(\boldsymbol{b}_i^*, \boldsymbol{b}_j^*)\|\right|^{1/2} = \prod_{i=1}^{n} |\boldsymbol{b}_i^*|.$$

Наконец, из неравенства

$$|\boldsymbol{b}_1|^2 \leqslant 2^{i-1} |\boldsymbol{b}_i^*|^2, \quad i = 1, \ldots, n,$$

следует, что

$$|\boldsymbol{b}_1|^{2n} \leqslant 2^{n(n-1)/2} \prod_{i=1}^{n} |\boldsymbol{b}_i^*|^2 = 2^{n(n-1)/2} d(\Lambda)^2,$$

что доказывает последнее утверждение теоремы. □

Теорема 7.8. *Пусть $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ — LLL-приведенный базис решетки Λ. Тогда для любого вектора $\boldsymbol{x} \in \Lambda \setminus \boldsymbol{0}$ справедливо неравенство*

$$|\boldsymbol{b}_1|^2 \leqslant 2^{n-1} |\boldsymbol{x}|^2.$$

Замечание 7.9. Утверждение теоремы 7.8 означает, что вектор $\boldsymbol{b}_1$ является одним из самых коротких векторов решетки.

Доказательство. Пусть $\boldsymbol{x} = \sum_{i=1}^{n} r_i \boldsymbol{b}_i \in \Lambda \setminus 0$, где $r_i \in \mathbb{Z}$. Представим $\boldsymbol{x}$ в виде $\boldsymbol{x} = \sum_{i=1}^{n} r_i^* \boldsymbol{b}_i^*$, где $r_i^* \in \mathbb{R}$. Если i_0 — максимальный номер,

для которого $r_i \neq 0$, то очевидно, что $r^*_{i_0} = r_{i_0}$. Поэтому (учитывая ортогональность $\boldsymbol{b}^*_i$) получим неравенства

$$|\boldsymbol{x}|^2 = \sum_{i=1}^{n} (r^*_i)^2 |\boldsymbol{b}^*_i|^2 \geqslant r^2_{i_0} |\boldsymbol{b}^*_{i_0}|^2 \geqslant |\boldsymbol{b}^*_{i_0}|^2 \geqslant \frac{1}{2^{i_0-1}} |\boldsymbol{b}_1|^2 \geqslant \frac{1}{2^{n-1}} |\boldsymbol{b}_1|^2,$$

откуда следует утверждение теоремы. □

Теорема 7.10. *Пусть $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ — LLL-приведенный базис решетки Λ, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$ — линейно независимые векторы решетки Λ. Тогда при всех $j \leqslant t$ выполнены неравенства*

$$|\boldsymbol{b}_j|^2 \leqslant 2^{n-1} \max(|\boldsymbol{x}_1|^2, \ldots, |\boldsymbol{x}_t|^2).$$

Доказательство. Разложим $\boldsymbol{x}_i$ по базису:

$$\boldsymbol{x}_i = \sum_{j=1}^{n} r_{ij} \boldsymbol{b}_j, \quad r_{ij} \in \mathbb{Z}, \quad i = 1, \ldots, t.$$

Обозначим для каждого i, $1 \leqslant i \leqslant t$, через $j(i)$ наибольший номер j такой, что $r_{ij} \neq 0$. Тогда, аналогично доказательству теоремы 7.8, выполнены неравенства

$$|\boldsymbol{x}_i|^2 \geqslant |\boldsymbol{b}^*_{j(i)}|^2, \quad i = 1, \ldots, t.$$

Не ограничивая общности рассуждений, можно считать, что $j(1) \leqslant \leqslant j(2) \leqslant \ldots \leqslant j(t)$. Тогда $j(i) \geqslant i$, $i = 1, \ldots, t$, так как если найдется номер i такой, что $j(i) < i$, то $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i \in L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{j(i)})$, что противоречит линейной независимости $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i$. По теореме 7.7 имеем

$$|\boldsymbol{b}_i|^2 \leqslant 2^{j(i)-1} |\boldsymbol{b}^*_{j(i)}|^2 \leqslant 2^{n-1} |\boldsymbol{x}_i|^2;$$

последнее неравенство следует из определения $j(i)$. □

§ 7.3. Алгоритм построения LLL-приведенного базиса решетки

В этом параграфе мы опишем построение LLL-приведенного базиса решетки $\Lambda \subseteq \mathbb{R}^n$. Мы сохраняем обозначения § 7.2.

Алгоритм построения LLL-приведенного базиса.

В начале работы алгоритма задан $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ — какой-то базис решетки Λ. В конце работы $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ — LLL-приведенный базис.

Проводится индукция по $k \in \{1, 2, \ldots, n+1\}$. В начале $k = 2$; когда $k = n + 1$, алгоритм заканчивает работу и выдает LLL-приведенный базис.

Для каждого k символом $(*)_k$ мы будем обозначать совокупность неравенств

$$\begin{cases} |\mu_{ij}| \leqslant \dfrac{1}{2}, & 1 \leqslant j \leqslant i < k, \\ |\boldsymbol{b}_i^* + \mu_{ii-1}\boldsymbol{b}_{i-1}^*|^2 \geqslant \dfrac{3}{4}|\boldsymbol{b}_{i-1}^*|^2, & 1 < i < k. \end{cases} \qquad (*)_k$$

Если $k = 2$, то условия $(*)_2$ выполнены, поскольку для i получится пустое множество значений $1 < i < 2$. Если $k = n + 1$, то $(*)_{n+1}$ означает, что базис приведенный (по определению).

Предположим, что для некоторого k, $1 < k < n + 1$, выполнены неравенства $(*)_k$. Мы хотим обеспечить выполнение $(*)_{k+1}$. Для начала обеспечим выполнение неравенства

$$|\mu_{k,k-1}| \leqslant \frac{1}{2}. \qquad (7.4)$$

Если (7.4) уже выполнено, то двигаемся дальше. В противном случае находим r — ближайшее целое к $\mu_{k,k-1}$, и заменяем $\boldsymbol{b}_k$ на

$$\boldsymbol{b}_k - r\boldsymbol{b}_{k-1} = \boldsymbol{b}_k^* + \sum_{j=1}^{k-2}(\mu_{kj} - r\mu_{k-1,j})\boldsymbol{b}_j^* + (\mu_{k,k-1} - r)\boldsymbol{b}_{k-1}^*.$$

Тогда коэффициент $\mu_{k,k-1}$ заменится на $\mu_{k,k-1} - r$, что обеспечит выполнение (7.4). Коэффициенты $\mu_{k,j}$ заменятся на $\mu_{kj} - r\mu_{k-1,j}$, $j = 1, \ldots, k-2$. Все остальные коэффициенты μ_{ij} и векторы $\boldsymbol{b}_i^*$ при $i < k$ и при $i > k$, а также вектор $\boldsymbol{b}_k^*$ не изменятся. Действительно, $\boldsymbol{b}_i^*$ есть проекция $\boldsymbol{b}_i$ на $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1})^\perp$; при замене $\boldsymbol{b}_k$ на $\boldsymbol{b}_k - r\boldsymbol{b}_{k-1}$ данные линейные оболочки не изменяются, значит, не изменяются и все $\boldsymbol{b}_i^*$. Далее, поскольку $\mu_{ij} = (\boldsymbol{b}_i, \boldsymbol{b}_j^*)/(\boldsymbol{b}_j^*, \boldsymbol{b}_i^*)$, то, при $i \neq k$, μ_{ij} не меняются в силу того, что $\boldsymbol{b}_i$ не изменился и все $\boldsymbol{b}_j^*$ также не изменились.

Продолжим обеспечение условий $(*)_{k+1}$ в предположении, что (7.4) выполнено.

1 случай. Пусть $k \geqslant 2$ и выполнено неравенство

$$|\boldsymbol{b}_k^* + \mu_{k,k-1}\boldsymbol{b}_{k-1}^*|^2 < \frac{3}{4}|\boldsymbol{b}_{k-1}^*|^2. \qquad (7.5)$$

Тогда мы меняем местами вектора $\boldsymbol{b}_k$ и $\boldsymbol{b}_{k-1}$. При этом изменятся векторы $\boldsymbol{b}_{k-1}^*$ и $\boldsymbol{b}_k^*$ и коэффициенты $\mu_{k,k-1}$, $\mu_{k-1,j}$, $\mu_{k,j}$, $\mu_{i,k-1}$, $\mu_{i,k}$, где $j < k-1$ или $i > k$. Остальные $\boldsymbol{b}_i$, $\boldsymbol{b}_i^*$ и μ_{ij} не изменятся по тем же причинам, что и ранее, т. е. потому, что $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_i)^\perp$ при $i \neq k$, $k-1$ остается той же самой, и векторы $\boldsymbol{b}_i^*$ при $i \neq k$, $k-1$ не изменятся.

Что произошло при замене местами $\boldsymbol{b}_k$ и $\boldsymbol{b}_{k-1}$? Вектор $\boldsymbol{b}_k^* + \mu_{k,k-1}\boldsymbol{b}_{k-1}^*$ ранее был равен проекции $\boldsymbol{b}_k$ на $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{k-2})^{\perp}$, а теперь он равен проекции нового вектора $\boldsymbol{b}_{k-1}$ на $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{k-2})^{\perp}$. Далее, $\boldsymbol{b}_{k-1}^*$ был равен проекции $\boldsymbol{b}_{k-1}$ на $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{k-2})^{\perp}$, а теперь это есть проекция нового вектора $\boldsymbol{b}_k$ на $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_i)^{\perp}$. Из неравенства (7.5) следует, что при нашей замене значение $|\boldsymbol{b}_{k-1}^*|^2$ уменьшилось более чем в $3/4$ раза.

Сделав эту замену местами $\boldsymbol{b}_k$ и $\boldsymbol{b}_{k-1}$, мы заменяем k на $k-1$ и оказываемся в условиях $(*)_{k-1}$.

2 случай. Пусть либо $k = 1$, либо

$$|\boldsymbol{b}_k^* + \mu_{k,k-1}\boldsymbol{b}_{k-1}^*| \geqslant \frac{3}{4}|\boldsymbol{b}_{k-1}^*|^2. \tag{7.6}$$

Если $k = 1$, то присваиваем k значение 2 и продолжаем процесс, т. е. обеспечиваем выполнение $(*)_2$.

Если выполнено условие (7.6) и $k > 1$, то обеспечиваем выполнение неравенств

$$|\mu_{kj}| \leqslant \frac{1}{2}, \quad 1 \leqslant j < k-1 \tag{7.7}$$

(для $j = k-1$ (7.7) уже выполнено в силу (7.4)). Пусть l — наибольший номер, для которого $|\mu_{kl}| > \frac{1}{2}$. Тогда $l \leqslant k-2$. Возьмем r — ближайшее целое к μ_{kl} — и заменим $\boldsymbol{b}_k$ на $\boldsymbol{b}_k - r\boldsymbol{b}_l$. При этом μ_{kj} заменятся на $\mu_{kl} - r$. Все остальные μ_{ij} и все векторы $\boldsymbol{b}_i^*$ останутся при этом неизменными. Мы продолжаем этот процесс, уменьшая l, пока не достигнем значения $l = 1$. В этом случае мы обеспечим выполнение условий $(*)_{k+1}$.

Если мы достигли выполнения условий $(*)_{n+1}$, то алгоритм останавливается, поскольку $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ образуют приведенный базис; в противном случае мы продолжаем процесс, т. е. увеличиваем значение k.

Конец алгоритма.

Докажем, что алгоритм заканчивает работу. Обозначим через

$$d_i = \det \|(\boldsymbol{b}_j, \boldsymbol{b}_l)\|_{1 \leqslant j,l \leqslant i}, \quad i = 1, \ldots, n.$$

Очевидно, что

$$d_i = \det \|(\boldsymbol{b}_j\| \cdot \|\boldsymbol{b}_l\|^T) = (\det \|\boldsymbol{b}_j\|_{1 \leqslant j \leqslant i})^2 =$$

$$= (\det \|\boldsymbol{b}_j^*\|_{1 \leqslant j \leqslant i})^2 = \det(\|\boldsymbol{b}_j^*\| \cdot \|\boldsymbol{b}_l^*\|^T) = \prod_{j=1}^{i} |\boldsymbol{b}_j^*|^2.$$

Поскольку 2 случай срабатывает за конечное число операций и увеличивает значение k на единицу, то нам нужно доказать, что 1 случай встретится лишь конечное число раз. Заметим, что при прохождении 1 случая при некотором значении k величина d_{k-1} уменьшается более чем в $3/4$ раза, поскольку так уменьшается значение $|\boldsymbol{b}^*_{k-1}|^2$.

Однако величины d_i ограничены снизу некоторой положительной постоянной, зависящей только от решетки Λ. Точнее, если обозначить через $m(\Lambda)$ квадрат длины кратчайшего ненулевого вектора Λ, то выполнены неравенства

$$m(\Lambda) \leqslant \left(\frac{4}{3}\right)^{(i-1)/2} d_i^{1/i}, \quad i = 1, \ldots, n$$

(см. [24, гл. 2]). Поэтому для каждого значения k 1 случай может встретиться лишь конечное число раз. Это означает, что алгоритм закончит работу.

Теорема 7.11 (см. [160]). *Если Λ — решетка в $\mathbb{Z}^n$ с базисом $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, причем*

$$|\boldsymbol{b}_i| \leqslant B, \quad i = 1, \ldots, n,$$

где $B \in \mathbb{R}$, $B \geqslant 2$, то алгоритм построения LLL-приведенного базиса делает $O(n^4 \log B)$ арифметических операций. При этом целые числа, встречающиеся в ходе работы алгоритма, имеют двоичную длину $O(n \log B)$ битов.

Замечание 7.12. В книге [89, гл. 2] приведена блок-схема алгоритма построения LLL-приведенного базиса, удобная для практической реализации.

Замечание 7.13. В работе [149] рассмотрена возможность изменения значения постоянной $c = 3/4$ в условии Ловаса

$$|\boldsymbol{b}^*_i + \mu_{i,i-1}\boldsymbol{b}^*_{i-1}|^2 \geqslant \frac{3}{4}|\boldsymbol{b}^*_{i-1}|^2.$$

При этом значение c меняется в ходе работы алгоритма приведения от $\frac{1}{4} + \varepsilon$ до 0,99.

Замечание 7.14. Многочисленные усовершенствования алгоритма построения LLL-приведенного базиса содержатся в работе [239]. В следующем параграфе мы опишем некоторые из них.

§ 7.4. Алгоритм Шнорра—Ойхнера и целочисленный LLL-алгоритм

В этом параграфе мы вкратце изложим некоторые модификации алгоритма построения LLL-приведенного базиса. Эти модификации описаны в работах [239; 102; 240; 216] и в книге [89].

Мы будем обозначать через B_i следующие величины:

$$B_i = (\boldsymbol{b}_i^*,\, \boldsymbol{b}_i^*), \quad i = 1, \ldots, n.$$

Одна из модификаций LLL-алгоритма была предложена Шнорром и Ойхнером. Она называется *LLL-алгоритмом с глубокой вставкой*. Для этой модификации теоретическая оценка сложности хуже, чем для исходного LLL-алгоритма, однако на практике с ее помощью зачастую можно получать более короткие вектора решетки, что немаловажно для практических приложений.

Идея LLL-алгоритма с глубокой вставкой заключается в следующем. Пусть $k > i$; вставим вектор $\boldsymbol{b}_k$ между $\boldsymbol{b}_{i-1}$ и $\boldsymbol{b}_i$ в процессе построения приведенного базиса. Поскольку

$$\boldsymbol{b}_k = \boldsymbol{b}_k^* + \sum_{j=1}^{k-1} \mu_{kj}\boldsymbol{b}_j^* = \boldsymbol{b}_k^* + \boldsymbol{v} + \sum_{j=1}^{i-1} \mu_{kj}\boldsymbol{b}_j^*,$$

где $\boldsymbol{v} = \sum\limits_{j=i}^{k-1} \mu_{kj}\boldsymbol{b}_j^*$, то после вставки $\boldsymbol{b}_k$ между $\boldsymbol{b}_i$ и $\boldsymbol{b}_{i-1}$ новое значение вектора $\boldsymbol{b}_i^*$ будет равно $\boldsymbol{b}_{i,\mathrm{new}}^*$:

$$\boldsymbol{b}_{i,\mathrm{new}}^* = \boldsymbol{b}_k^* + \boldsymbol{v} = \boldsymbol{b}_k^* + \sum_{j=i}^{k-1} \mu_{kj}\boldsymbol{b}_j^*.$$

Новое значение квадрата длины вектора $\boldsymbol{b}_i^*$ равно $B_{i,\mathrm{new}}$:

$$B_{i,\mathrm{new}} = \left(\boldsymbol{b}_k^* + \sum_{j=i}^{k-1} \mu_{kj}\boldsymbol{b}_j^*,\, \boldsymbol{b}_k^* + \sum_{j=i}^{k-1} \mu_{kj}\boldsymbol{b}_j^* 4\right) = B_k + \sum_{j=i}^{k-1} \mu_{kj}^2 B_j.$$

Если новое значение $B_{i,\mathrm{new}}$ меньше старого $\left(\text{например, } B_{i,\mathrm{new}} \leqslant \frac{3}{4}B_i\right)$, то такая замена местами уместна; если $i = k-1$, то такая замена равносильна одному из шагов LLL-алгоритма. Однако если $i \neq k-1$, то приходится перевычислять больше значений коэффициентов μ_{ij}, чем в исходном LLL-алгоритме; это увеличивает время работы алгоритма

Шнорра—Ойхнера. Блок-схему LLL-алгоритма с глубокой вставкой можно найти в [89, гл. 2, алгоритм 2.6.4].

Другая модификация LLL-алгоритма называется *целочисленным LLL-алгоритмом*. Она была предложена Де Вегером. Эта модификация применима в случае, когда матрица Грама $G = \|(\boldsymbol{b}_i, \boldsymbol{b}_j)\|_{i,j=1,\ldots,n}$ базиса $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ решетки Λ является целочисленной (это имеет место, например, в случае, когда сама решетка Λ содержится в $\mathbb{Z}^n$). Тогда справедлива следующая теорема, описывающая некоторые арифметические свойства решетки.

Теорема 7.15. *Пусть* (*как и в* § 7.3) $d_0 = 1$, $d_i = \det \|(\boldsymbol{b}_j, \boldsymbol{b}_l)\|_{j,l=1,\ldots,i}$, $i = 1, \ldots, n$. *Если матрица Грама решетки* Λ *является целочисленной, то для всех номеров* i, $1 \leqslant i \leqslant n$, *и всех* $j < i$ *выполнены следующие утверждения*:

1. $d_{i-1} B_i \in \mathbb{Z}$, $d_i \mu_{ij} \in \mathbb{Z}$;
2. *для всех* m, $j < m \leqslant i$, $d_i \cdot \sum\limits_{1 \leqslant k \leqslant j} \mu_{ik} \mu_{mk} B_k \in \mathbb{Z}$.

Доказательство. В § 7.3 мы видели, что

$$d_i = \prod_{j=1}^{i} |\boldsymbol{b}_j^*|^2 = \prod_{j=1}^{i} B_j.$$

Поэтому $d_i = d_{i-1} B_i \in \mathbb{Z}$. Пусть $j < i$. Рассмотрим вектор

$$\boldsymbol{v} = \boldsymbol{b}_i - \sum_{k=1}^{j} \mu_{ik} \boldsymbol{b}_k^* = \boldsymbol{b}_i^* + \sum_{k=j+1}^{i-1} \mu_{ik} \boldsymbol{b}_k^*. \tag{7.8}$$

Очевидно, что $(\boldsymbol{v}, \boldsymbol{b}_k^*) = 0$ при $k = 1, \ldots, j$. Поскольку $L_{\text{об}}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k) = L_{\text{об}}(\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_k^*)$, то

$$(\boldsymbol{v}, \boldsymbol{b}_k) = 0, \quad k = 1, \ldots, j \tag{7.9}$$

и

$$\boldsymbol{v} = \boldsymbol{b}_i - \sum_{k=1}^{j} x_k \boldsymbol{b}_k \tag{7.10}$$

при некоторых $x_1, \ldots, x_k \in \mathbb{R}$. Из (7.8) и (7.9) получаем систему линейных уравнений

$$\begin{pmatrix} (\boldsymbol{b}_i, \boldsymbol{b}_1) \\ \ldots \\ (\boldsymbol{b}_i, \boldsymbol{b}_j) \end{pmatrix} = \begin{pmatrix} (\boldsymbol{b}_1, \boldsymbol{b}_1) & \ldots & (\boldsymbol{b}_1, \boldsymbol{b}_j) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ (\boldsymbol{b}_j, \boldsymbol{b}_1) & \ldots & (\boldsymbol{b}_j, \boldsymbol{b}_j) \end{pmatrix} \begin{pmatrix} x_1 \\ \ldots \\ x_j \end{pmatrix}.$$

Решая эту систему по правилу Крамера, получим, что $x_k = \dfrac{m_k}{d_j}$, $k = 1, \ldots, j$, где $m_k \in \mathbb{Z}$. Поскольку

$$\sum_{k=1}^{j} x_k \boldsymbol{b}_k = \sum_{k=1}^{j} \mu_{ik} \boldsymbol{b}_k^*,$$

то $x_j = \mu_{ij}$, откуда следует первое утверждение теоремы.

Докажем второе утверждение. Для вектора $\boldsymbol{v}$, определенного в формуле (7.8), справедливо равенство

$$d_j \boldsymbol{v} = d_j \boldsymbol{b}_i - \sum_{k=1}^{j} d_j x_k \boldsymbol{b}_k,$$

причем по доказанному выше $d_i x_k \in \mathbb{Z}$. Это означает, что $d_j \boldsymbol{v} \in \Lambda$, откуда $(d_j \boldsymbol{v},\, \boldsymbol{b}_m) \in \mathbb{Z}$ для всех $m = 1, \ldots, n$. Тогда

$$d_j \left(\boldsymbol{b}_i - \sum_{k=1}^{j} \mu_{ik} \boldsymbol{b}_k^*,\, \boldsymbol{b}_m \right) \in \mathbb{Z}, \quad m = 1, \ldots, n.$$

Поэтому

$$d_j \left(\sum_{k=1}^{j} \mu_{ik} \boldsymbol{b}_k^*,\, \boldsymbol{b}_m \right) \in \mathbb{Z}, \quad m = 1, \ldots, n.$$

Пусть $j < m \leqslant i$. Тогда $\boldsymbol{b}_m = \boldsymbol{b}_m^* + \sum\limits_{l=1}^{m-1} \mu_{ml} \boldsymbol{b}_l^*$ и

$$d_j \left(\sum_{k=1}^{j} \mu_{ik} \boldsymbol{b}_k^*,\, \boldsymbol{b}_m \right) = d_j \left(\sum_{k=1}^{j} \mu_{ik} \boldsymbol{b}_k^*,\, \boldsymbol{b}_m^* + \sum_{l=1}^{m-1} \mu_{ml} \boldsymbol{b}_l^* \right) =$$

$$= d_j \cdot \sum_{k=1}^{j} \mu_{ik} \mu_{mk} B_k \in \mathbb{Z},$$

что и требовалось доказать. □

Следствие 7.16. *В условиях теоремы положим дополнительно*

$$\lambda_{ij} = d_j \mu_{ij} \in \mathbb{Z} \quad \textit{при } i < j, \qquad \lambda_{ii} = d_i.$$

При фиксированных i, j, таких, что $j \leqslant i$, рассмотрим последовательность

$$u_0 = (\boldsymbol{b}_i,\, \boldsymbol{b}_j), \qquad u_k = \frac{d_k u_{k-1} - \lambda_{ik} \lambda_{jk}}{d_{k-1}}, \qquad k = 1, \ldots, j-1.$$

Тогда $u_k \in \mathbb{Z}$, $u_{j-1} = \lambda_{ij}$.

Доказательство. Покажем, что

$$u_k = d_k\left((\boldsymbol{b}_i, \boldsymbol{b}_j) - \sum_{l=1}^{k} \frac{\lambda_{il}\lambda_{jl}}{d_l d_{l-1}}\right). \tag{7.11}$$

При $k = 0$ это очевидно. Пусть для $m < k$ формула (7.11) верна. Тогда

$$u_k = \frac{d_k u_{k-1}}{d_{k-1}} - \frac{d_k}{d_{k-1}} \cdot \frac{\lambda_{ik}\lambda_{jk}}{d_k} = d_k\left((\boldsymbol{b}_i, \boldsymbol{b}_j) - \sum_{l=1}^{k-1} \frac{\lambda_{il}\lambda_{jl}}{d_l d_{l-1}} - \frac{1}{d_k d_{k-1}}\lambda_{ik}\lambda_{jk}\right),$$

что доказывает (7.11) для k. Из (7.11) следует, что

$$u_k = d_k\left((\boldsymbol{b}_i, \boldsymbol{b}_j) - \sum_{l=1}^{k} \mu_{il}\mu_{jl}B_l\right), \tag{7.12}$$

поскольку

$$\frac{\lambda_{il}\lambda_{jl}}{d_l d_{l-1}} = \frac{d_l}{d_{l-1}}\mu_{il}\mu_{jl} = B_l\mu_{il}\mu_{jl}.$$

Из теоремы и (7.12) следует, что $u_k \in \mathbb{Z}$. Равенство $u_{j-1} = \lambda_{ij}$ также следует из (7.12), поскольку

$$\begin{aligned} u_{j-1} &= d_{j-1}\left((\boldsymbol{b}_i, \boldsymbol{b}_j) - \sum_{l=1}^{j-1} \mu_{il}\mu_{jl}B_l\right) = \\ &= d_{j-1}\left(\left(\boldsymbol{b}_i^* + \sum_{t=1}^{i-1} \mu_{it}\boldsymbol{b}_t^*,\ \boldsymbol{b}_j^* + \sum_{s=1}^{j-1} \mu_{js}\boldsymbol{b}_s^*\right) - \sum_{l=1}^{j-1} \mu_{il}\mu_{jl}B_l\right) = \\ &= d_{j-1}\left(\mu_{ij}B_j + \sum_{s=1}^{j-1} \mu_{is}\mu_{js}B_s - \sum_{l=1}^{j-1} \mu_{il}\mu_{jl}B_l\right) = d_{j-1}\mu_{ij}\ B_j = d_j\mu_{ij} = \lambda_{ij}. \end{aligned}$$

Следствие доказано. □

Целочисленный LLL-алгоритм устроен следующим образом. Он работает не с векторами $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, а с целочисленной матрицей Грама G. Выход алгоритма — целочисленная матрица H, выражающая координаты LLL-приведенного базиса через координаты исходного базиса решетки Λ. При этом с помощью теоремы и следствия из нее все вычисления проводятся только с целыми числами. Блок-схему алгоритма можно найти в [89, гл. 2, алгоритм 2.6.7].

Последней модификацией LLL-алгоритма, о которой мы упомянем в данном параграфе, является MLLL-алгоритм Поста. Этот

алгоритм работает с векторами $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^n$, образующими множество $\Lambda = \mathbb{Z}\boldsymbol{b}_1 + \ldots + \mathbb{Z}\boldsymbol{b}_n \subseteq \mathbb{R}^n$, причем линейная независимость $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ не предполагается. MLLL-алгоритм вычисляет ранг $\mathbb{Z}$-модуля Λ и находит его LLL-приведенный базис. Блок-схему алгоритма также можно найти в [89, гл. 2, алгоритм 2.6.8].

§ 7.5. Некоторые приложения LLL-алгоритма

Одним из возможных приложений LLL-алгоритма является нахождение ядра и образа линейного отображения евклидова пространства, задаваемого целочисленной матрицей. Соответствующие алгоритмы описаны в [89, § 2.7]. В данном параграфе мы опишем другие приложения LLL-алгоритма: нахождение целочисленной линейной зависимости для заданных действительных чисел и нахождение коротких векторов в решетках.

Пусть $z_1, \ldots, z_n$ — фиксированные действительные числа, $z_1 \neq 0$. Мы хотим найти целочисленную линейную зависимость

$$x_1 z_1 + \ldots + x_n z_n = 0, \quad x_1, \ldots, x_n \in \mathbb{Z}, \tag{7.13}$$

где не все x_i равны 0. Выберем достаточно большое натуральное число N и рассмотрим квадратичную форму

$$Q(a_1, \ldots, a_n) = Q(\boldsymbol{a}) = a_2^2 + \ldots + a_n^2 + N(z_1 a_1 + \ldots + z_n a_n)^2. \tag{7.14}$$

Поскольку $Q(\boldsymbol{a})$ — положительно определенная квадратичная форма, она задает скалярное произведение на $\mathbb{R}^n$. Идея нахождения $x_1, \ldots, x_n$, удовлетворяющих (7.13), заключается в следующем: если вектор $(a_1, \ldots, a_n) \in \mathbb{Z}^n$ является коротким относительно нормы, индуцированной квадратичной формой $Q(\boldsymbol{a})$, то величина $z_1 a_1 + \ldots + z_n a_n$ будет небольшой, небольшими будут и коэффициенты a_j при $j > 1$. Поэтому если $z_1, \ldots, z_n$ линейно зависимы, то мы скорее всего найдем решение (7.13).

Мы начинаем со *стандартного* базиса решетки $\Lambda \subseteq \mathbb{R}^n$, состоящего из векторов $\boldsymbol{b}_i = (\delta_{i1}, \ldots, \delta_{in})$, $i = 1, \ldots, n$, где δ_{ij} — символ Кронекера. С помощью LLL-алгоритма из § 7.3 (или его модификаций из § 7.4) мы находим LLL-приведенный базис Λ (скалярное произведение в $\mathbb{R}^n$ задается квадратичной формой $Q(a)$ из (7.14)). Когда LLL-приведенный базис будет построен, один из его коротких векторов, возможно, даст решение (7.13), как это было объяснено выше.

Лемма 7.17. *Если* $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ *— стандартный базис* $\mathbb{Z}^n$, *то в процессе ортогонализации Грама—Шмидта выполняются равенства*

$$\begin{aligned} &\mu_{i1} = z_i/z_1, && i = 2, \ldots, n;\\ &\mu_{ij} = 0, && 2 \leqslant j < i \leqslant n;\\ &\boldsymbol{b}_i^* = \boldsymbol{b}_i - \frac{z_i}{z_1}\boldsymbol{b}_1, && i = 2, \ldots, n. \end{aligned}$$

Далее, если $B_i = |\boldsymbol{b}_i^*|^2$ *— длины векторов* $\boldsymbol{b}_i^*$ *в нашей метрике, задаваемой* $Q(\boldsymbol{a})$, *то*

$$B_1 = Nz_1^2, \quad B_i = 1 \quad \textit{при} \quad i = 2, \ldots, n.$$

Доказательство. Пусть $[\boldsymbol{u}, \boldsymbol{v}]$ — скалярное произведение на $\mathbb{R}^n$, индуцируемое $Q(\boldsymbol{a})$. Тогда по определению

$$[\boldsymbol{u}, \boldsymbol{v}] = \frac{1}{2}(Q(\boldsymbol{u} + \boldsymbol{v}) - Q(\boldsymbol{u}) - Q(\boldsymbol{v})).$$

Теперь легко видеть, что векторы

$$\begin{aligned} &\boldsymbol{b}_1^* = \boldsymbol{b}_1 = (1, 0, \ldots, 0),\\ &\boldsymbol{b}_i^* = \boldsymbol{b}_i - \frac{z_i}{z_1}\boldsymbol{b}_1 = \left(-\frac{z_i}{z_1}, 0, \ldots, 1, \ldots, 0\right), \qquad i = 2, \ldots, n, \end{aligned}$$

являются ортогональными. В самом деле, при $i, j \geqslant 2$, $i \neq j$,

$$\begin{aligned} Q(\boldsymbol{b}_i^* + \boldsymbol{b}_j^*) - Q(\boldsymbol{b}_i^*) - Q(\boldsymbol{b}_j^*) = \\ = 1 + 1 + N((-z_i - z_j) + z_i + z_j)^2 - \\ - (1 + N(-z_i + z_i)^2) - (1 + N(-z_j + z_j)^2) = 0 \end{aligned}$$

и, при $j \geqslant 2$,

$$\begin{aligned} Q(\boldsymbol{b}_1^* + \boldsymbol{b}_j) - Q(\boldsymbol{b}_1^*) - Q(\boldsymbol{b}_j^*) = \\ = 1 + N((z_1 - z_j) + z_j)^2 - Nz_1^2 - (1 + N(-z_i + z_i)^2) = 0. \end{aligned}$$

Утверждение леммы о значениях B_i также очевидно. □

Прежде чем описывать алгоритм нахождения линейной зависимости, скажем несколько слов о выборе натурального параметра N в формуле (7.14). Число N выбирается эвристически; можно попробовать несколько различных значений N. В книге [89, гл. 2] рекомендуется выбирать N из промежутка $\left(\frac{1}{\varepsilon}; \frac{1}{\varepsilon^2}\right)$, если все значения z_i не слишком удалены от 1 (скажем, лежат в интервале $(10^{-6}; 10^6)$) и известны нам с точностью ε. Число ε также должно быть достаточно малым: если мы

предполагаем, что решения x_i в (7.13) не превосходят X, то ε можно взять равным $X^{-3n/2}$. Число всегда следует выбирать меньшим X^{-n} (здесь, однако, значение X нам неизвестно и также определяется эвристически).

Алгоритм нахождения линейной зависимости (эвристический).
На входе алгоритма заданы $z_1, \ldots, z_n \in \mathbb{R}$, не все равные нулю, и параметр $N \in \mathbb{N}$. На выходе малая по абсолютной величине линейная комбинация $x_1 z_1 + \ldots + x_n z_n$ с небольшими коэффициентами $x_i \in \mathbb{Z}$, не всеми равными нулю.

1 шаг. Для стандартного базиса $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ решетки $\mathbb{Z}^n$ по лемме 7.17 находим ортогональный базис $\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*$ и коэффициенты μ_{ij}.

2 шаг. Теперь к базису $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ (и соответствующему ему ортогональному базису $\boldsymbol{b}_1^*, \ldots, \boldsymbol{b}_n^*$ и коэффициентам μ_{ij}) применяем LLL-алгоритм из §7.3; скалярное произведение в $\mathbb{R}^n$ задается квадратичной формой (7.14). В результате будет построен LLL-приведенный базис Λ решетки $\mathbb{Z}^n$.

3 шаг. Берем короткие векторы из найденного LLL-приведенного базиса; их координаты $x-1, \ldots, x_n$, возможно, являются решениями (7.13), что устанавливается проверкой.

Конец алгоритма.

Замечание 7.18. Вместо стандартного LLL-алгоритма можно использовать LLL-алгоритм с глубокой вставкой, описанный в §7.4, поскольку он зачастую выдает более короткие векторы.

Теперь опишем алгоритм для нахождения всех достаточно коротких векторов решетки. Точнее, пусть $\Lambda = \mathbb{Z}\boldsymbol{b}_1 + \ldots + \mathbb{Z}\boldsymbol{b}_n$ — решетка в $\mathbb{R}^n$, C — положительная постоянная; требуется найти все векторы $\boldsymbol{b} \in \Lambda$ такие, что

$$|\boldsymbol{b}|^2 \leqslant C. \tag{7.15}$$

Эта задача является трудной; описываемый ниже алгоритм в некоторых случаях может иметь сложность, экспоненциально зависящую от размерности пространства n.

Если $\boldsymbol{b} \in \Lambda$, то $\boldsymbol{b} = x_1\boldsymbol{b}_1 + \ldots + x_n\boldsymbol{b}_n$ при некоторых $x_1, \ldots, x_n \in \mathbb{Z}$. Тогда неравенство (7.15) примет вид

$$|Q(\boldsymbol{x})| = |Q(x_1, \ldots, x_n)| \leqslant C,$$

где

$$Q(x_1, \ldots, x_n) = (x_1\boldsymbol{b}_1 + \ldots + x_n\boldsymbol{b}_n,\ x_1\boldsymbol{b}_1 + \ldots + x_n\boldsymbol{b}_n) = \sum_{i,j=1}^{n} (\boldsymbol{b}_i, \boldsymbol{b}_j) x_i x_j$$

— положительно определенная квадратичная форма. Ее мы приводим к более удобному виду

$$Q(\boldsymbol{x}) = \sum_{i=1}^{n} q_{ii}\left(x_i + \sum_{j=i+1}^{n} q_{ij}x_j\right)^2, \quad q_{ii} > 0. \tag{7.16}$$

Для этого запишем $Q(\boldsymbol{x})$ в виде

$$Q(\boldsymbol{x}) = \sum_{i,j=1}^{n} a_{ij}x_ix_j, \quad \text{где} \quad a_{ij} = (\boldsymbol{b}_i, \boldsymbol{b}_j), \ \ a_{ij} = a_{ji}.$$

Число $a_{11} = (\boldsymbol{b}_1, \boldsymbol{b}_1)$ положительно. Тогда

$$\begin{aligned} Q(\boldsymbol{x}) = a_{11}x_1^2 + 2\sum_{j=2}^{n} a_{1j}x_1x_j + \sum_{i=2}^{n}\sum_{j=2}^{n} a_{ij}x_ix_j = \\ = a_{11}\left(x_1^2 + 2\sum_{j=2}^{n}\frac{a_{1j}}{a_{11}}x_1x_j + \left(\sum_{j=2}^{n}\frac{a_{1j}}{a_{11}}x_j\right)^2\right) + \\ + \sum_{i=2}^{n}\sum_{j=2}^{n} a_{ij}x_ix_j - \left(\sum_{j=2}^{n}\frac{a_{1j}}{a_{11}}x_j\right)^2 = \\ = a_{11}\left(x_1 + \sum_{j=2}^{n}\frac{a_{1j}}{a_{11}}x_j\right)^2 + Q_1(x_2, \ldots, x_n), \end{aligned}$$

где

$$Q_1(x_2, \ldots, x_n) = \sum_{j=2}^{n}\sum_{j=2}^{n} a_{ij}x_ix_j - \left(\sum_{j=2}^{n}\frac{a_{ij}}{a_{11}}x_j\right)^2$$

также является положительно определенной квадратичной формой, поскольку при очевидной замене переменных справедливо равенство

$$Q(x_1, \ldots, x_n) = a_{11}y_1^2 + Q_1(y_2, \ldots, y_n).$$

Далее аналогично поступаем с $Q_1(x_2, \ldots, x_n)$ и в конечном итоге приводим $Q(\boldsymbol{x})$ к виду (7.16). Представление (7.16) называется *разложением Холецкого* для квадратичной формы $Q(\boldsymbol{x})$.

Алгоритм нахождения коротких векторов решетки.

На входе алгоритма заданы положительно определенная квадратичная форма $Q(\boldsymbol{x}) = Q(x_1, \ldots, x_n)$ вида (7.16) и положительная постоянная C. На выходе получаются все векторы $\boldsymbol{x} \in \mathbb{Z}^n$ такие, что $Q(\boldsymbol{x}) \leqslant C$ (из каждой пары $\pm\boldsymbol{x}$ выдается только один вектор).

1 шаг. $i := n$, $T_i := C$, $V_i := 0$.

2 шаг. $z := \sqrt{T_i/q_{ii}}$, $L_i := [z - V_i]$, $x_i := \lceil -z - V_i \rceil - 1$.

3 шаг. $x_i := x_i + 1$. Если $x_i > L_i$, то $i := i + 1$ и вернуться на 3-й шаг. Иначе при $i > 1$ присвоить

$$T_{i-1} := T_i - q_{ii}(x_i + V_i)^2, \quad i := i - 1, \quad V_i := \sum_{j=i+1}^{i} q_{ij} x_j$$

и вернуться на 2-й шаг.

4 шаг. Если $(x_1, \ldots, x_n) = (0, \ldots, 0)$, то алгоритм заканчивает работу. Иначе выдается очередной вектор $\boldsymbol{x} = (x_1, \ldots, x_n)$, значение $Q(\boldsymbol{x}) = C - T_1 + q_{11}(x_1 + V_1)^2$ и мы возвращаемся на 3 шаг.

Конец алгоритма.

Покажем, что алгоритм работает корректно. По сути, в этом алгоритме мы описали перебор значений $x_1, \ldots, x_n$. В начале $i = n$, $T_n = C$, $V_n = 0$, $z = \sqrt{C/q_{nn}}$, $L_n = \sqrt{C/q_{nn}}$, $x_n = \lceil -\sqrt{C/q_{nn}} \rceil - 1$. Это минимальное возможное значение $x_n \in \mathbb{Z}$, для которого вектор $\boldsymbol{x} = (x_1, \ldots, x_n)$ может удовлетворять неравенству $Q(\boldsymbol{x}) \leqslant C$. Действительно, для вектора $\boldsymbol{x} = (x_1, \ldots, x_n)$ из неравенства $Q(\boldsymbol{x}) \leqslant C$ в силу (7.16) следует, что $q_{nn}x_n^2 \leqslant C$, $|x_n| \leqslant \sqrt{+C/q_{nn}}$, откуда $x_n \leqslant L_n$ (это проверка окончания перебора) и $x_n \geqslant \lceil -\sqrt{C/q_{nn}} \rceil$. Затем на 3 шаге мы присвоим величине T_{n-1} значение $c - q_{n,n-1}x_n^2$, V_{n-1} — значение $q_{n-1,n}x_n$ и вернемся на 2 шаг. Здесь мы аналогично выберем минимально возможное (при уже имеющемся значении x_n) значение x_{n-1}. Оно будет удовлетворять неравенству

$$q_{n-1,n-1}(x_{n-1} + V_{n-1})^2 \leqslant T_{n-1},$$

откуда

$$|x_{n-1} + V_{n-1}| \leqslant \sqrt{\frac{T_{n-1}}{q_{n-1,n-1}}} = z.$$

Поэтому

$$x_{n-1} \geqslant \lceil -z - V_{n-1} \rceil,$$

и также $x_{n-1} \leqslant z - V_{n-1}$, откуда $x_{n-1} \leqslant L_{n-1}$. Из этого следует, что если на 3 шаге $x_{n-1} > L_{n-1}$, то мы вышли за пределы диапазона изменения x_{n-1} (при данном x_n), и тогда мы перейдем к следующему значению x_n. Далее, аналогично, при заданных x_n и x_{n-1}, мы перебираем x_{n-2} и так далее. Перебор закончится, когда (возрастающая) переменная x_n достигнет значения 0 и все остальные (возрастающие)

переменные при $x_n = 0$ также достигнут значения 0. Тогда, действительно, из каждой пары $\pm\boldsymbol{x}$, удовлетворяющей неравенству $Q(\boldsymbol{x}) \leqslant C$, мы найдем ровно один вектор. Заметим также, что на шаге 3 очередное значение $T_{i-1} := T_i - q_{ii}(x_i + V_i)^2$ удовлетворяет неравенству $T_{i-1} \geqslant 0$, поскольку из неравенства $x_i \leqslant L_i$ и неравенства $x_i \geqslant \lceil -z - V_i \rceil$ следует, что $|x_i + V_i| \leqslant \sqrt{T_i/q_{ii}} = z$.

Дальнейшее усовершенствование данного алгоритма было предложено Финке и Постом. В нем после нахождения разложения Холецкого для квадратичной формы $Q(\boldsymbol{x}) = \boldsymbol{x}^T A\boldsymbol{x}$, что равносильно представлению A в виде $A = R^T R$ для некоторой верхнетреугольной матрицы R, применяется LLL-алгоритм к строкам матрицы R^{-1}. В результате мы сможем существенно уменьшить перебор значений x_i в алгоритме; на практике это значительно ускоряет его работу. Дальнейшие детали см. в [121] и [89, гл. 2].

Замечание 7.19. Неизвестно, является ли задача нахождения кратчайшего вектора решетки NP-полной. NP-полнота этой задачи доказана для нормирований $|\cdot|_\infty$ и $|\cdot|_1$ пространства $\mathbb{R}^n$, см. [48].

§ 7.6. Алгоритм Фергюсона—Форкейда

В данном параграфе мы опишем алгоритм Фергюсона—Форкейда, см. [117], а также см. [116; 119; 120; 118; 76]. Этот алгоритм предназначен для нахождения целочисленной линейной зависимости заданного конечного набора действительных чисел. Один алгоритм для решения этой задачи мы уже описали выше в § 7.5.

Введем некоторые обозначения. Фиксируем $n \in \mathbb{N}$, $n \geqslant 2$. Для вещественной матрицы $A = \|a_{ij}\|$ размера $n \times n$ положим $|A| = \sqrt{\sum\limits_{i,j=1}^{n} a_{ij}^2}$. Очевидно, что $|A + B| \leqslant |A| + |B|$, $|AB| \leqslant |A| \cdot |B|$.

Фиксируем вектор $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$. Вектор $\boldsymbol{m} \in \mathbb{Z}^n$, $\boldsymbol{m} \neq \boldsymbol{0}$, мы будем называть *соотношением*, если

$$\boldsymbol{x}\boldsymbol{m}^{\mathrm{T}} = 0. \tag{7.17}$$

Мы будем считать, что все координаты вектора $\boldsymbol{x}$ отличны от нуля, поскольку в противном случае поиск соотношения (7.17) тривиален. Также обозначим через P матрицу размера $n \times n$:

$$P = \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} \cdot I_n - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}, \tag{7.18}$$

где I_l обозначает единичную матрицу размера $l \times l$, $l = 1, 2, 3, \ldots$ Тогда

$\boldsymbol{x}P = \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}(\boldsymbol{x}I_n - \boldsymbol{x}) = \boldsymbol{0}$. Далее, если $\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{y}P = \boldsymbol{0}$, то

$$\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{x}) - (\boldsymbol{y}, \boldsymbol{x})\boldsymbol{x} = \boldsymbol{0},$$

откуда $y_i = x_i \cdot \dfrac{(\boldsymbol{y}, \boldsymbol{x})}{(\boldsymbol{x}, \boldsymbol{x})}$. Следовательно, rank $P = n - 1$.

Рассмотрим матрицу H размера $n \times (n-1)$, столбцы которой составляют базис $(L_{\text{об}}(\boldsymbol{x}))^{\perp}$. Очевидно, rank $H = n - 1$, $\boldsymbol{x}H = \boldsymbol{0}$. Далее мы покажем, как вычислить матрицу H. Обозначим через $\boldsymbol{v}_i$ i-ю строку матрицы H, $i = 1, \ldots, n$; $\boldsymbol{v}_i \in \mathbb{R}^{n-1}$. Положим $H_0 = \boldsymbol{0} \in \mathbb{R}^{n-1}$,

$$H_i = \begin{pmatrix} \boldsymbol{v}_1 \\ \ldots \\ \boldsymbol{v}_i \end{pmatrix}, \quad i = 1, \ldots, n. \tag{7.19}$$

При $i \geqslant 1$ H_i — матрицы размера $i \times (n-1)$, $H_n = H$.

Также положим $G_0 = 1$ и рассмотрим $i \times i$ матрицы

$$G_i = \begin{pmatrix} (\boldsymbol{v}_1, \boldsymbol{v}_1) & \ldots & (\boldsymbol{v}_1, \boldsymbol{v}_i) \\ \ldots\ldots\ldots\ldots\ldots\ldots & & \\ (\boldsymbol{v}_i, \boldsymbol{v}_1) & \ldots & (\boldsymbol{v}_i, \boldsymbol{v}_i) \end{pmatrix} = H_i H_i^{\mathrm{T}}, \tag{7.20}$$

$i = 1, \ldots, n$. Так как все координаты вектора $\boldsymbol{x}$ отличны от нуля, то матрицы G_i обратимы при $1 \leqslant i < n$. Действительно, поскольку $\boldsymbol{x}H = \boldsymbol{0}$, то $x_1\boldsymbol{v}_1 + \ldots + x_n\boldsymbol{v}_n = \boldsymbol{0}$. Следовательно, $\boldsymbol{v}_n$ линейно выражается через $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{n-1}$, и так как rank $H = n - 1$, то $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{n-1}$ линейно независимы. Значит, G_i — матрицы Грама для $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_i$ и поэтому являются невырожденными при $i < n$. Положим

$$P_i = H_i^{\mathrm{T}} G_i^{-1} H_i, \quad i = 0, 1, \ldots, n, \tag{7.21}$$

P_i — матрицы размера $(n-1) \times (n-1)$. Также обозначим

$$Q_i = I_{n-1} - P_i, \tag{7.22}$$

и рассмотрим числа

$$C(k, j) = \boldsymbol{v}_k P_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}}, \quad 1 \leqslant j \leqslant k \leqslant n. \tag{7.23}$$

Для всех остальных номеров j, k положим $C(k, j) = 0$. Получаем массив значений $C(H)$, определяемый матрицей H.

Следующий алгоритм является вспомогательным. Он переводит пару $\boldsymbol{x}$, H в пару $\boldsymbol{x}A^{-1}$, AH для некоторой матрицы A, вычисляемой в ходе его работы.

Алгоритм 1.

Шаг 0. Если существует координата $x_i = 0$, то алгоритм заканчивает работу.

Шаг 1. Пусть $T = \| T_{kj} \|$ — нижняя треугольная матрица размера $n \times n$, на диагонали которой стоят единицы, а при $1 \leqslant j < k \leqslant n$ элемент T_{kj} равен ближайшему целому к числу

$$-\frac{C(k, j)}{C(j, j)} + \sum_{j<i<k} \frac{T_{ki} C(i, j)}{C(j, j)}.$$

(Ниже мы покажем, что $C(j, j) \neq 0$, т. е. числа T_{kj} определены.) Тогда набор $\boldsymbol{x}$, H, $C(H)$ переводится в набор $\boldsymbol{x}T^{-1}$, TH, $C(TH)$.

Шаг 2. Пусть номер i, $1 \leqslant i < n$, такой, что

$$2^i C(i, i) = \max_{1 \leqslant i < n} 2^j C(j, j).$$

Пусть E — $n \times n$-матрица перестановки i-й и $(i+1)$-й строк H. Тогда набор $\boldsymbol{x}$, H, $C(H)$ переводится в набор $\boldsymbol{x}E^{-1}$, EH, $C(EH)$.

Шаг 3. Положить $A = ET$.

Конец алгоритма.

Заметим, что матрица A, построенная на 3 шаге, является целочисленной и невырожденной.

Матрица H, определенная выше, может быть найдена следующим способом. Если мы рассмотрим матрицу

$$X = \begin{pmatrix} 1 & \dots & 0 & x_1 \\ \dots & \dots & \dots & \dots \\ 0 & \dots & 1 & x_{n-1} \\ 0 & \dots & 0 & x_n \end{pmatrix},$$

то $PX = (H\mathbf{0}^{\mathrm{T}})$. Действительно, последний столбец PX равен $P\boldsymbol{x}^{\mathrm{T}} = \mathbf{0}$, как показано выше, поскольку $P = P^{\mathrm{T}}$. Далее, в силу невырожденности матрицы X имеем $\operatorname{rank} H = \operatorname{rank} PX = \operatorname{rank} P = n - 1$. Так как $\boldsymbol{x}P = \mathbf{0}$, то $\boldsymbol{x}PX = \mathbf{0}$, следовательно, матрица H — искомая.

Положим $A_0 = I_n$, $\boldsymbol{x}^{(0)} = \boldsymbol{x}$, $H^{(0)} = H$ и определим последовательность

$$\boldsymbol{x}^{(k)},\ H^{(k)},\ A_k, \qquad k = 1, 2, \dots \tag{7.24}$$

Если для некоторого $k \geqslant 1$ мы уже построили $\boldsymbol{x}^{(k-1)}$, $H^{(k-1)}$ и вектор $\boldsymbol{x}^{(k-1)}$ не имеет нулевых координат, то применим к паре $\boldsymbol{x}^{(k-1)}$, $H^{(k-1)}$ алгоритм 1. На 3-м шаге им будет построена матрица $A = A_k$. Тогда мы полагаем $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} A_k^{-1}$, $H^{(k)} = A_k H^{(k-1)}$ (фактически, это есть выход алгоритма 1). Если обозначить

$$R_k = A_1^{-1} \dots A_k^{-1}, \tag{7.25}$$

то

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}R_k, \quad H^{(k)} = R_k^{-1}H. \tag{7.26}$$

Предположим дополнительно, что вектор $\boldsymbol{x}$ удовлетворяет условиям

$$\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} = 1, \quad 0 < x_1 < \ldots < x_n. \tag{7.27}$$

Такой вектор мы будем называть *нормализованным*.

Теорема 7.20. *Пусть вектор $\boldsymbol{x}$ нормализован и имеет соотношение $\boldsymbol{m}$ нормы M. Тогда при некотором k, удовлетворяющем неравенству*

$$k < 2^{n+1} n \log(3n^3M^2), \tag{7.28}$$

один из столбцов матрицы R_k является соотношением.

Прежде, чем приступить к доказательству теоремы, докажем ряд вспомогательных утверждений. Пусть по-прежнему H_j, G_j, P_j, Q_j задаются формулами (7.19), (7.20), (7.21), (7.22), $\operatorname{rank} H_j = j$ при $1 \leqslant j \leqslant n-1$, $\operatorname{rank} H = n-1$, $\boldsymbol{v}_j$ — строки H, матрицы G_j обратимы при $j \leqslant n-1$. Если

$$V = L_{\text{об}}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n), \tag{7.29}$$

то $\dim V = n-1$.

Лемма 7.21. *Для всех j, k, $1 \leqslant j, k \leqslant n$, справедливы следующие утверждения*:

1) $P_j^{\mathrm{T}} = P_j$, $Q_j^{\mathrm{T}} = Q_j$, $P_jP_j = P_j$, $Q_jQ_j = Q_j$, $P_jQ_j = \mathbf{0}$, *и если* $\boldsymbol{v} \in \mathbb{R}^n$, $\boldsymbol{v}P_j \neq \mathbf{0}$ $(\boldsymbol{v}Q_j \neq \mathbf{0})$, *то* $\boldsymbol{v}P_j\boldsymbol{v}^{\mathrm{T}} > 0 \quad (\boldsymbol{v}Q_j\boldsymbol{v}^{\mathrm{T}} > 0)$;

2) $P_jP_k = P_{\min(j,k)}$, $Q_jQ_k = Q_{\max(j,k)}$, $P_jQ_k = P_j - P_{\min(j,k)} = Q_k - Q_{\max(j,k)}$;

3) *При* $j \leqslant k$ $VP_j \subseteq VP_k$, $VQ_j \supseteq VQ_k$, $\boldsymbol{v}_j = \boldsymbol{v}_jP_k$, $P_jQ_k = \mathbf{0}$;

4) $Q_kP_j = P_jQ_k$;

5) $I_{n-1} = \sum\limits_{1 \leqslant j \leqslant n-1} (Q_{j-1} - Q_j) = \sum\limits_{1 \leqslant j \leqslant n-1} P_jQ_{j-1}$ *есть ортогональные разложения единичной матрицы* I_{n-1};

6) $\operatorname{rank} P_jQ_{j-1} = 1$, $\boldsymbol{v}_jQ_{j-1} \neq \mathbf{0}$ *при* $1 \leqslant j \leqslant n-1$.

Доказательство. Так как $G_j^{\mathrm{T}} = G_j$, то очевидно, что $P_j^{\mathrm{T}} = P_j$, откуда $Q_j^{\mathrm{T}} = Q_j$. Далее, $P_jP_j = H_j^{\mathrm{T}}(H_jH_j^{\mathrm{T}})^{-1}H_jH_j^{\mathrm{T}}(H_jH_j^{\mathrm{T}})^{-1}H_j = P_j$; тогда из (7.22) следует, что $Q_jQ_j = Q_j$. Поэтому $P_jQ_j = P_j - P_j^2 = \mathbf{0}$. Наконец, из $\boldsymbol{v}P_j \neq \mathbf{0}$ следует, что $\boldsymbol{v}P_j\boldsymbol{v}^{\mathrm{T}} = \boldsymbol{v}P_j^2\boldsymbol{v}^{\mathrm{T}} = \boldsymbol{v}P_jP_j^{\mathrm{T}}\boldsymbol{v}^{\mathrm{T}} > 0$; аналогично, из $\boldsymbol{v}Q_j \neq \mathbf{0}$ следует $\boldsymbol{v}Q_j\boldsymbol{v}^{\mathrm{T}} > 0$. Таким образом, первое утверждение леммы доказано.

Далее, поскольку $V = \sum\limits_{j=1}^{n} \mathbb{R}\boldsymbol{v}_j$, то

$$VP_k = \sum_{j=1}^{k} \mathbb{R}\boldsymbol{v}_j. \tag{7.30}$$

Действительно,

$$P_k = (\boldsymbol{v}_1^{\mathrm{T}} \dots \boldsymbol{v}_k^{\mathrm{T}}) G_k^{-1} \begin{pmatrix} \boldsymbol{v}_1 \\ \dots \\ \boldsymbol{v}_k \end{pmatrix},$$

откуда при $l \leqslant k$ получаем

$$\boldsymbol{v}_l P_k = ((\boldsymbol{v}_l, \boldsymbol{v}_1) \dots (\boldsymbol{v}_l, \boldsymbol{v}_k)) G_k^{-1} \begin{pmatrix} \boldsymbol{v}_1 \\ \dots \\ \boldsymbol{v}_k \end{pmatrix} = \boldsymbol{v}_l.$$

А при $l > k$ имеем

$$\boldsymbol{v}_l P_k = ((\boldsymbol{v}_l, \boldsymbol{v}_1) \dots (\boldsymbol{v}_l, \boldsymbol{v}_k)) G_k^{-1} \begin{pmatrix} \boldsymbol{v}_1 \\ \dots \\ \boldsymbol{v}_k \end{pmatrix} \subseteq \sum_{j=1}^{k} \mathbb{R}\boldsymbol{v}_j.$$

Из формулы (7.30) следует, что при $j \leqslant k$

$$VP_j \subset VP_k, \quad \boldsymbol{v}_j = \boldsymbol{v}_j P_k. \tag{7.31}$$

Так как $I_{n-1} = P_j + Q_j$, то $V = VP_j + VQ_j$. Это разложение является ортогональным, поскольку $P_j Q_j = \mathbf{0}$. Действительно, $(\boldsymbol{u}P_j, \boldsymbol{w}Q_j) = (\boldsymbol{u}P_j Q_j^{\mathrm{T}}, \boldsymbol{w}) = 0$. Поэтому из включения $VP_j \subseteq VP_k$ при $j \leqslant k$ следует, что $VQ_j \supseteq VQ_k$. Из этого также следует, что при $j \leqslant k$ имеем $P_j Q_k = \mathbf{0}$, так как если $P_j Q_k \neq \mathbf{0}$, то найдется вектор $\boldsymbol{u}$ такой, что $\boldsymbol{w} = \boldsymbol{u}P_j Q_k \neq \mathbf{0}$. Отсюда $0 < (\boldsymbol{w}, \boldsymbol{w}) = (\boldsymbol{u}P_j Q_k, \boldsymbol{u}P_j Q_k) = (\boldsymbol{u}P_j Q_k, \boldsymbol{u}P_j)$. Но $\boldsymbol{u}P_j Q_k \in VQ_k$, $\boldsymbol{u}P_j \in VP_j$, и по доказанному VP_j ортогонально VQ_k. Таким образом, доказано третье утверждение леммы.

Из определения G_0 и H_0 следует, что $P_0 = \mathbf{0}$ — нулевая матрица, $Q_0 = I_{n-1}$. Также $P_{n-1} = H_{n-1}^{\mathrm{T}} G_{n-1}^{\mathrm{T}} H_{n-1} = I_{n-1}$, поскольку матрица H_{n-1} — квадратная, и $G_{n-1} = H_{n-1} H_{n-1}^{\mathrm{T}}$. Следовательно, $Q_{n-1} = \mathbf{0}$. Поэтому

$$I_{n-1} = \sum_{j=1}^{n-1} (Q_{j-1} - Q_j). \tag{7.32}$$

Докажем, что $P_j P_k = P_{\min(j,k)}$. Пусть $j \geqslant k$. Тогда $P_j P_k = (I_{n-1} - Q_j) P_k = = P_k - Q_j P_k = P_k$, так как $(Q_j P_k)^{\mathrm{T}} = Q_j P_k = \mathbf{0}$ по доказанному нами

третьему утверждению леммы. Аналогично доказываются остальные равенства второго пункта леммы.

Четвертое утверждение леммы следует теперь из равенств

$$Q_k P_j = Q_k^{\mathrm{T}} P_j^{\mathrm{T}} = (P_j Q_k)^{\mathrm{T}} = (P_j - P_{\min(j,k)})^{\mathrm{T}} = P_j - P_{\min(j,k)} = P_j Q_k.$$

Равенство $I_{n-1} = \sum\limits_{j=1}^{n-1} P_j Q_{j-1}$ следует из доказанной формулы (7.32) и равенства $Q_{j-1} - Q_j = P_j Q_{j-1}$.

Докажем ортогональность разложений в пятом утверждении леммы. При $i \neq j$ имеем

$$(Q_{j-1} - Q_j)(Q_{i-1} - Q_i) = Q_{j-1}Q_{i-1} - Q_j Q_{j-1} - Q_{j-1}Q_i + Q_j Q_i = M_{ij}.$$

Если $i < j$, то $M_{ij} = Q_{j-1} - Q_j - Q_{j-1} + Q_j = \mathbf{0}$, аналогично рассматривается случай $i > j$. Пятое утверждение леммы доказано.

Наконец, докажем шестое утверждение леммы. Выполнены неравенства $\operatorname{rank} P_j Q_{j-1} \leqslant 1$, поскольку для вектора $\boldsymbol{v} \in V$ имеем

$$\boldsymbol{v} P_j Q_{j-1} = \left(\sum_{l=1}^{j} \alpha_l \boldsymbol{v}_l\right) Q_{j-1} = \sum_{l=1}^{j} \alpha_l \boldsymbol{v}_l (I_{n-1} - P_{j-1}) = \alpha_j \boldsymbol{v}_j (I_{n-1} - P_{j-1}).$$

То есть $P_j Q_{j-1}$ переводит $(n-1)$-мерное пространство V в не более чем одномерное. Но так как матрица $I_{n-1} = \sum\limits_{j=1}^{n-1} P_j Q_{j-1}$ имеет ранг $n-1$, то ранг каждого слагаемого равен 1 и $\boldsymbol{v}_j Q_{j-1} \neq \mathbf{0}$ при $j = 1, \ldots, n-1$. $\square$

Следствие 7.22. $C(j, j) = \boldsymbol{v}_j P_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} > 0$.

Лемма 7.23. *Пусть B — вещественная матрица размера $n \times n$; обозначим B_j матрицы размера $j \times j$, образованные первыми j строками и столбцами матрицы B, $j = 1, \ldots, n$. Пусть* $\det B_j \neq 0$, $\overline{H}_j = B_j H_j$, $j = 1, \ldots, n$, *и матрицы $\overline{P}_j$, $\overline{Q}_j$ получены с помощью формул* (7.21) *и* (7.22) *с заменой H_j на $\overline{H}_j$. Тогда $\overline{P}_j = P_j$, $\overline{Q}_j = Q_j$.*

Доказательство. По определению

$$\overline{P}_j = \overline{H}_j^{\mathrm{T}} (\overline{H}_j \overline{H}_j^{\mathrm{T}})^{-1} \overline{H}_j = H_j^{\mathrm{T}} B_j^{\mathrm{T}} (B_j H_j H_j^{\mathrm{T}} B_j^{\mathrm{T}})^{-1} B_j H_j =$$
$$= H_j^{\mathrm{T}} B_j^{\mathrm{T}} (B_j^{\mathrm{T}})^{-1} (H_j H_j^{\mathrm{T}})^{-1} B_j^{-1} B_j H_j = P_j.$$

Равенство $\overline{Q}_j = Q_j$ теперь очевидно. $\square$

В этом параграфе для действительного числа α мы будем обозначать через $[[\alpha]]$ ближайшее целое к α число и $\{\{\alpha\}\} = \alpha - [[\alpha]]$; при $\alpha = \dfrac{1}{2} + n$, $n \in \mathbb{Z}$, будем для определенности считать, что $[[\alpha]] = n$.

Из доказательства леммы 7.21 следует, что справедливы равенства

$$\boldsymbol{v}_k = \boldsymbol{v}_k I_{n-1} = \sum_{i=1}^{k} a_{ki}\boldsymbol{v}_i P_i Q_{i-1} \tag{7.33}$$

при некоторых $a_{ki} \in \mathbb{R}$, $k = 1, \ldots, n$. Действительно, поскольку $\boldsymbol{v}_i P_i = \boldsymbol{v}_i$ и $\boldsymbol{v}_i Q_{i-1} = \boldsymbol{v}_i - \boldsymbol{v}_i P_{i-1} = \boldsymbol{v}_i + \sum\limits_{j=1}^{i-1} \alpha_{ij}\boldsymbol{v}_j$, то такие числа a_{ki} можно найти. При этом, очевидно, $a_{kk} = 1$, $k = 1, \ldots, n$. Рассмотрим числа $C(k, j) = \boldsymbol{v}_k P_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}}$ из (7.23). Тогда

$$a_{kj} = \frac{C(k, j)}{C(j, j)}, \quad 1 \leqslant j < k \leqslant n. \tag{7.34}$$

Действительно, по (7.23) и лемме 7.21 имеем

$$C(k, i) = \boldsymbol{v}_k Q_{i-1}^{\mathrm{T}} P_i^{\mathrm{T}} \boldsymbol{v}_i^{\mathrm{T}} = (\boldsymbol{v}_k, \boldsymbol{v}_i P_i Q_{i-1}) = \\ = a_{ki}(\boldsymbol{v}_i P_i Q_{i-1}, \boldsymbol{v}_i P_i Q_{i-1}) = a_{ki} C(i, i),$$

что доказывает (7.34).

Теперь определим последовательность $D_{k,j}$:

$$D_{k,k-1} = [[a_{k,k-1}]], \quad k = 2, \ldots, n, \tag{7.35}$$

и для очередной пары чисел k, j, где $k \leqslant n$, $k - 1 > j \geqslant 1$, положим

$$D_{k,j} = \left[\left[a_{kj} - \sum_{j<i<k} D_{k,i} a_{ij}\right]\right]. \tag{7.36}$$

Для всех остальных значений k и j, $n \geqslant k$, $j \geqslant 1$, положим

$$D_{kj} = 0. \tag{7.37}$$

Матрица $D = \| D_{k,j} \|_{k,j=1,\ldots,n}$, является нижней треугольной, а матрица

$$B = I_n - D \tag{7.38}$$

является обратимой целочисленной. При этом

$$B^{-1} = I_n + D + D^2 + \ldots + D^{n-1}.$$

Положим

$$\overline{H} = BH. \tag{7.39}$$

Строки матрицы $\overline{H}$ составляют вектора $\overline{\boldsymbol{v}}_k$,

$$\overline{\boldsymbol{v}}_k = \boldsymbol{v}_k - \sum_{1 \leqslant j < k} D_{k,j}\boldsymbol{v}_j. \tag{7.40}$$

По лемме 7.23 и формуле (7.33) получим (пользуясь тем, что $\overline{P}_j = P_j$ и $\overline{Q}_j = Q_j$) ортогональное разложение векторов $\overline{\boldsymbol{v}}_k$, $k = 1, \ldots, n$:

$$\overline{\boldsymbol{v}}_k = \sum_{j=1}^{k} \overline{a}_{kj} \overline{\boldsymbol{v}}_j P_j Q_{j-1}, \quad \overline{a}_{kj} \in \mathbb{R}. \tag{7.41}$$

Лемма 7.24. *При* $1 \leqslant j < k \leqslant n$ *справедливы неравенства* $|\overline{a}_{kj}| \leqslant 1/2$.

Доказательство. Пусть $j < k$. Тогда по (7.40) и лемме 7.21, п. 4 имеем

$$\begin{aligned}\overline{\boldsymbol{v}}_k \overline{P}_j \overline{Q}_{j-1} = \overline{\boldsymbol{v}}_k P_j Q_{j-1} = \boldsymbol{v}_k P_j Q_{j-1} - \sum_{1 \leqslant i < k} D_{k,i} \boldsymbol{v}_i P_j Q_{j-1} = \\ = a_{kj} \boldsymbol{v}_j P_j Q_{j-1} - \sum_{j \leqslant i < k} D_{k,i} \boldsymbol{v}_i P_j Q_{j-1} = \\ = \left(a_{kj} - \sum_{j \leqslant i < k} D_{k,i} a_{ij} \right) \boldsymbol{v}_j P_j Q_{j-1}, \end{aligned} \tag{7.42}$$

последние два равенства следуют из (7.33) и ортогональности разложения. Также из (7.40) и леммы 7.21 получаем

$$\overline{\boldsymbol{v}}_j P_j Q_{j-1} = \boldsymbol{v}_j P_j Q_{j-1}. \tag{7.43}$$

Тогда

$$\overline{\boldsymbol{v}}_j \overline{P}_j \overline{Q}_{j-1} \overline{\boldsymbol{v}}_j^{\mathrm{T}} = \boldsymbol{v}_j P_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}},$$

поскольку

$$\begin{aligned}(\boldsymbol{v}_j P_j Q_{j-1}, \overline{\boldsymbol{v}}_j) = (\boldsymbol{v}_j P_j, \overline{\boldsymbol{v}}_j Q_{j-1}) = \\ = (\boldsymbol{v}_j P_j, (\boldsymbol{v}_j - \sum_{l<j} D_{jl} \boldsymbol{v}_l) Q_{j-1}) = (\boldsymbol{v}_j P_j, \boldsymbol{v}_j Q_{j-1}).\end{aligned}$$

Далее, по (7.42) и (7.43) получаем

$$\begin{aligned}\overline{\boldsymbol{v}}_k \overline{P}_j \overline{Q}_{j-1} \overline{\boldsymbol{v}}_j^{\mathrm{T}} = \left(a_{kj} - \sum_{j \leqslant i < k} D_{ki} a_{ij} \right) \boldsymbol{v}_j P_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} = \\ = \left(a_{kj} - \sum_{j \leqslant i < k} D_{ki} a_{ij} \right) \boldsymbol{v}_j P_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} = \left(a_{kj} \sum_{j \leqslant i < k} D_{ki} a_{ij} \right) C(j, j).\end{aligned}$$

Поэтому

$$\overline{a}_{kj} = \frac{\overline{\boldsymbol{v}}_k \overline{P}_j \overline{Q}_{j-1} \overline{\boldsymbol{v}}_j^{\mathrm{T}}}{\overline{\boldsymbol{v}}_j \overline{P}_j \overline{Q}_{j-1} \overline{\boldsymbol{v}}_j^{\mathrm{T}}} = a_{kj} - \sum_{j \leqslant i < k} D_{ki} a_{ij} = a_{kj} - \sum_{j < i < k} D_{ki} a_{ij} - D_{kj}.$$

Утверждение леммы теперь вытекает из формулы (7.36) и равенства $a_{jj} = 1$. $\square$

Зафиксируем i, $1 \leqslant i < n$. Пусть T_{in} — матрица перестановки i и $i+1$ строк матрицы H (T_{in} — матрица размера $n \times n$). Положим

$$\widetilde{H} = T_{in} H = \begin{pmatrix} \boldsymbol{v}_1 \\ \dots \\ \boldsymbol{v}_{i+1} \\ \boldsymbol{v}_i \\ \dots \\ \boldsymbol{v}_n \end{pmatrix}$$

и обозначим $\widetilde{H}_i$, $\widetilde{P}_j$, $\widetilde{Q}_j$ — матрицы, получающиеся по формулам (7.19), (7.21) и (7.22), где H и H_i заменены на $\widetilde{H}$ и $\widetilde{H}_i$. Тогда матрица $\widetilde{H}_i$ удовлетворяет равенству

$$\widetilde{H}_i = \begin{pmatrix} H_{i-1} \\ \boldsymbol{v}_{i+1} \end{pmatrix}.$$

Строки $\widetilde{H}_n$ обозначим $\tilde{\boldsymbol{v}}_j$, $j = 1, \dots, n$. Пусть T_{ij} — матрица размера $j \times j$, образованная первыми j строками и столбцами матрицы T_{in}. Если $j \neq i$, то матрица T_{ij} обратима, и вычисление, проведенное в доказательстве леммы 7.23, верно для $B_j = T_{ij}$. Следовательно,

$$\widetilde{P}_j = P_j, \qquad \widetilde{Q}_j = Q_j, \qquad 1 \leqslant j \leqslant n-1,\ j \neq i. \tag{7.44}$$

Отсюда получим ортогональное разложение

$$I_{n-1} = \sum_{j=1}^{n-1} \widetilde{P}_j \widetilde{Q}_{j-1} = \sum_{j=1}^{i-1} P_j Q_{j-1} + \widetilde{P}_i Q_{i-1} + P_{i+1} \widetilde{Q}_i + \sum_{j=i+1}^{n-1} P_j Q_{j-1}.$$

Далее мы обозначаем символом $\langle \boldsymbol{u}, \boldsymbol{w} \rangle$ угол между векторами $\boldsymbol{u}$ и $\boldsymbol{w}$, отсчитываемый от первого вектора ко второму и по модулю не превосходящий π.

Лемма 7.25. *Пусть* $1 \leqslant i < n$, $\theta = \langle \boldsymbol{v}_{i+1} Q_{i-1},\ \boldsymbol{v}_i Q_{i-1} \rangle$. *Тогда*

$$\tilde{\boldsymbol{v}}_i \widetilde{Q}_{i-1} \tilde{\boldsymbol{v}}_i^{\mathrm{T}} = \boldsymbol{v}_{i+1} Q_i \boldsymbol{v}_{i+1}^{\mathrm{T}} \csc^2 \theta, \qquad \tilde{\boldsymbol{v}}_{i+1} \widetilde{Q}_i \tilde{\boldsymbol{v}}_{i+1}^{\mathrm{T}} = \boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} \sin^2 \theta.$$

Доказательство. Если $\boldsymbol{v} = \boldsymbol{u} + c\boldsymbol{w}$, где $\boldsymbol{u}\boldsymbol{w}^{\mathrm{T}} = 0$, то

$$(\boldsymbol{v}\boldsymbol{w}^{\mathrm{T}})^2 = c^2 (\boldsymbol{w}\boldsymbol{w}^{\mathrm{T}})^2, \qquad c^2 \boldsymbol{w}\boldsymbol{w}^{\mathrm{T}} = \boldsymbol{v}\boldsymbol{v}^{\mathrm{T}} \cdot \cos^2 \langle \boldsymbol{v}, \boldsymbol{w} \rangle, \qquad \boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} = \boldsymbol{v}\boldsymbol{v}^{\mathrm{T}} \cdot \sin^2 \langle \boldsymbol{v}, \boldsymbol{w} \rangle.$$

Кроме того, $\overline{\theta} = \langle \tilde{\boldsymbol{v}}_{i+1}\widetilde{Q}_{i-1},\, \tilde{\boldsymbol{v}}_i\widetilde{Q}_{i-1}\rangle = \langle \boldsymbol{v}_i Q_{i-1},\, \boldsymbol{v}_{i+1}Q_{i-1}\rangle = -\theta$. Положим

$$\boldsymbol{u} = \tilde{\boldsymbol{v}}_{i+1}\widetilde{Q}_i = \tilde{\boldsymbol{v}}_{i+1}\widetilde{P}_{i+1}\widetilde{Q}_i = \boldsymbol{v}_i Q_i, \quad \boldsymbol{v} = \tilde{\boldsymbol{v}}_{i+1}\widetilde{Q}_{i-1} = \boldsymbol{v}_i Q_{i-1},$$
$$\boldsymbol{w} = \tilde{\boldsymbol{v}}_i\widetilde{P}_i\widetilde{Q}_{i-1} = \tilde{\boldsymbol{v}}_i\widetilde{Q}_{i-1} = \boldsymbol{v}_{i+1}Q_{i-1}.$$

Из леммы 7.21 следует, что $\boldsymbol{u}\boldsymbol{w}^{\mathrm{T}} = 0$. Отсюда, поскольку $\langle \boldsymbol{w},\, \boldsymbol{v}\rangle = \theta$, следует второе утверждение леммы, при $\boldsymbol{v} = \boldsymbol{u} + c\boldsymbol{w}$ для некоторого c. Но по лемме 7.21 имеем

$$\widetilde{P}_i\widetilde{Q}_{i-1} = \widetilde{Q}_{i-1} - \widetilde{Q}_i,$$
$$\boldsymbol{v} - \boldsymbol{u} = \tilde{\boldsymbol{v}}_{i+1}(\widetilde{Q}_{i-1} - \widetilde{Q}_i) = \tilde{\boldsymbol{v}}_{i+1}\widetilde{P}_i\widetilde{Q}_{i-1} = c\tilde{\boldsymbol{v}}_i\widetilde{Q}_{i-1} = c\boldsymbol{w}$$

при некотором $c \in \mathbb{R}$.

Для доказательства первого утверждения леммы положим

$$\boldsymbol{u} = \boldsymbol{v}_{i+1}Q_i = \boldsymbol{v}_{i+1}P_{i+1}Q_i, \quad \boldsymbol{v} = \boldsymbol{v}_{i+1}Q_{i-1} = \tilde{\boldsymbol{v}}_i\widetilde{Q}_{i-1} = \tilde{\boldsymbol{v}}_i\widetilde{P}_i\widetilde{Q}_{i-1},$$
$$\boldsymbol{w} = \boldsymbol{v}_i Q_{i-1} = \boldsymbol{v}_i P_i Q_{i-1} \neq 0.$$

Снова $\boldsymbol{u}\boldsymbol{w}^{\mathrm{T}} = 0$. Так как $Q_{i-1} - Q_i = P_i Q_{i-1}$, то

$$\boldsymbol{v}_{i+1}(Q_{i-1} - Q_i) = \boldsymbol{v}_{i+1}P_i Q_{i-1} = c\boldsymbol{v}_i P_i Q_{i-1} = c\boldsymbol{w}$$

при некотором $c \in \mathbb{R}$. Поскольку $\langle \boldsymbol{v},\, \boldsymbol{w}\rangle = \theta$, из наших рассуждений следует первое утверждение леммы. $\square$

Лемма 7.26. *Если найдется номер i, $1 \leqslant i < n$ такой, что $a_{i+1,i} = \{\{a_{i+1,i}\}\}$ и при этом*

$$\boldsymbol{v}_i Q_{i-1}\boldsymbol{v}_i^{\mathrm{T}} > 2\boldsymbol{v}_{i+1}Q_i\boldsymbol{v}_{i+1}^{\mathrm{T}},$$

то

$$3\boldsymbol{v}_i Q_{i-1}\boldsymbol{v}_i^{\mathrm{T}} > 4\boldsymbol{v}_{i+1}Q_{i-1}\boldsymbol{v}_{i+1}^{\mathrm{T}}.$$

Доказательство. Поскольку $Q_{i-1} - Q_i = P_i Q_{i-1}$, то

$$\boldsymbol{v}_{i+1}Q_{i-1} = \boldsymbol{v}_{i+1}Q_i + a_{i+1,i}\boldsymbol{v}_i Q_{i-1},$$

причем слагаемые в правой части этого равенства являются ортогональными векторами. Отсюда

$$\boldsymbol{v}_{i+1}Q_{i-1}\boldsymbol{v}_{i+1}^{\mathrm{T}} = \boldsymbol{v}_{i+1}Q_i\boldsymbol{v}_{i+1}^{\mathrm{T}} + a_{i+1,i}^2\boldsymbol{v}_i Q_{i-1}\boldsymbol{v}_i^{\mathrm{T}}.$$

Тогда, пользуясь условием леммы, получим

$$4\boldsymbol{v}_{i+1}Q_{i-1}\boldsymbol{v}_{i+1}^{\mathrm{T}} < 2\boldsymbol{v}_i Q_{i-1}\boldsymbol{v}_i^{\mathrm{T}} + \boldsymbol{v}_i Q_{i-1}\boldsymbol{v}_i^{\mathrm{T}},$$

поскольку $4a_{i+1,i}^2 \leqslant 1$. $\square$

Теперь приведем некоторые оценки, связанные с величиной

$$L(H) = \sum_{j=1}^{n-1} (2n - 2j + 1) \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}}. \tag{7.45}$$

Мы будем обозначать символом $\Lambda(A)$ сумму диагональных элементов квадратной матрицы A.

Лемма 7.27. *Пусть матрица B и числа a_{kj} определены формулами* (7.33) *и* (7.38). *Тогда*

1) $\Lambda(HH^{\mathrm{T}}) = \sum\limits_{j=1}^{n-1} \Big(1 + \sum\limits_{k=j+1}^{n} a_{kj}^2\Big) \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}}$;

2) *если для всех k, j, $k \neq j$, выполнено равенство $a_{kj} = \{\{a_{kj}\}\}$, то $\Lambda(HH^{\mathrm{T}}) < L(H)$;*

3) *для матрицы $\overline{H} = BH$ выполнено равенство $L(\overline{H}) = L(H)$.*

Доказательство. Нетрудно видеть, что справедливо равенство $\boldsymbol{v}_k \boldsymbol{v}_k^{\mathrm{T}} = \sum\limits_{j=1}^{k} a_{kj}^2 \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}}$. Тогда

$$\Lambda(HH^{\mathrm{T}}) = \sum_{k=1}^{n} \boldsymbol{v}_k \boldsymbol{v}_k^{\mathrm{T}} = \sum_{k=1}^{n} \sum_{j=1}^{k} a_{kj}^2 \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} = \sum_{j=1}^{n-1} \Big(1 + \sum_{k=j+1}^{n} a_{kj}^2\Big) \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}}$$

(мы воспользовались тем, что $a_{kk} = 1$, $Q_{n-1} = \mathbf{0}$). Далее, если $a_{kj} = \{\{a_{kj}\}\}$ при $k \neq j$, то $1 + \sum\limits_{k=j+1}^{n} a_{kj}^2 \leqslant 1 + \dfrac{n-j}{4}$. Из этого следует второе утверждение леммы. Наконец, при $\overline{H} = BH$ из доказательства леммы 7.24 следует, что

$$\overline{\boldsymbol{v}}_j \overline{Q}_{j-1} \overline{\boldsymbol{v}}_j^{\mathrm{T}} = \overline{\boldsymbol{v}}_j \overline{P}_j \overline{Q}_{j-1} \overline{\boldsymbol{v}}_j^{\mathrm{T}} = \boldsymbol{v}_j P_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} = \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}},$$

откуда вытекает третье утверждение леммы. □

Лемма 7.28. *Пусть номер i, $1 \leqslant i \leqslant n-1$, такой, что*

$$2^i \boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} \geqslant 2^j \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}}$$

для всех $j = 1, \ldots, n-1$. Тогда

$$L(H) \leqslant 2^n n \boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}}.$$

Доказательство. Поскольку

$$\boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} \leqslant 2^{i-j} \boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}},$$

то

$$L(H) \leqslant 2^i \boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} \sum_{j=1}^{n-1} \frac{2n-2j+1}{2^j} \leqslant 2^{i+1} n \boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}},$$

так как $i \leqslant n-1$ и $n \geqslant 2$. $\square$

Лемма 7.29. *Пусть i — то же, что в лемме 7.28; рассмотрим матрицу перестановок T_{in} и матрицу $\widetilde{H} = T_{in} H$. Пусть $a_{i,i+1} = \{\{a_{i,i+1}\}\}$. Тогда*

$$L(\widetilde{H}) \leqslant \Bigl(1 - \frac{1}{2^{n+1} n}\Bigr) L(H).$$

Доказательство. В силу (7.44) и леммы 7.25 имеем

$$\begin{aligned} L(H) - L(\widetilde{H}) &= (2n-2i+1)(\boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} - \tilde{\boldsymbol{v}}_i \widetilde{Q}_{i-1} \tilde{\boldsymbol{v}}_i^{\mathrm{T}}) + \\ &\quad + (2n-2i-1)(\boldsymbol{v}_{i+1} Q_i \boldsymbol{v}_{i+1}^{\mathrm{T}} - \tilde{\boldsymbol{v}}_{i+1} \widetilde{Q}_i \tilde{\boldsymbol{v}}_{i+1}^{\mathrm{T}}) = \\ &= (2n-2i+1)(\boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} - \tilde{\boldsymbol{v}}_i \widetilde{Q}_{i-1} \tilde{\boldsymbol{v}}_i^{\mathrm{T}}) + \\ &\quad + (2n-2i-1)(\tilde{\boldsymbol{v}}_i \widetilde{Q}_{i-1} \tilde{\boldsymbol{v}}_i^{\mathrm{T}} \sin^2\theta - \boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} \sin^2\theta) = \\ &= (2n-2i+1-(2n-2i-1)\sin^2\theta)(\boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} - \boldsymbol{v}_{i+1} Q_i \boldsymbol{v}_{i+1}^{\mathrm{T}}). \end{aligned}$$

Тогда $L(H) - L(\widetilde{H}) \geqslant 2(\boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}} - \boldsymbol{v}_{i+1} Q_i \boldsymbol{v}_{i+1}^{\mathrm{T}})$. По выбору i выполнено условие леммы 7.26, откуда, используя лемму 7.28, получим

$$L(H) - L(\widetilde{H}) \geqslant 2(\boldsymbol{v}_i Q_{i-1} \boldsymbol{v}_i^{\mathrm{T}}/4) \geqslant \frac{L(H)}{2^{n+1} n}.$$

Лемма доказана. $\square$

Лемма 7.30. *Имеет место неравенство*

$$L(H) < (\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}})^2 n^2.$$

Доказательство. Поскольку $I_{n-1} = P_{i-1} + Q_{i-1}$ есть ортогональное разложение матрицы I_{n-1}, и так как $\boldsymbol{v}_j P_{j-1} \boldsymbol{v}_j^{\mathrm{T}} \geqslant 0$, то

$$\boldsymbol{v}_j \boldsymbol{v}_j^{\mathrm{T}} = \boldsymbol{v}_j P_{j-1} \boldsymbol{v}_j^{\mathrm{T}} + \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} \geqslant \boldsymbol{v}_j Q_{j-1} \boldsymbol{v}_j^{\mathrm{T}} \geqslant 0.$$

Из определения матрицы P следует, что

$$\boldsymbol{v}_j \boldsymbol{v}_j^{\mathrm{T}} = (\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}})^2 - x_j^2 (\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} + x_n^2) \leqslant (\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}})^2,$$

при $j = 1, \ldots, n-1$. Поэтому

$$L(H) \leqslant \sum_{j=1}^{n-1} (2n-2j+1) \boldsymbol{v}_j \boldsymbol{v}_j^{\mathrm{T}} \leqslant (\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}})^2 (n^2-1),$$

что и требовалось доказать. $\square$

Лемма 7.31. *Пусть вектор* $\boldsymbol{m} \in \mathbb{Z}^n \setminus \mathbf{0}$ *является соотношением. Тогда для любой невырожденной целочисленной матрицы* A *размера* $n \times n$ *выполнены неравенства*:

$$0 < (\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}})^2 \leqslant (\boldsymbol{m}\boldsymbol{m}^{\mathrm{T}})|AP|^2,$$

$$0 < \frac{(x_n \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}})^2}{(\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} + x_n^2(n-1))} \leqslant (\boldsymbol{m}\boldsymbol{m}^{\mathrm{T}})|AH|^2.$$

Доказательство. Поскольку $1 \leqslant |A\boldsymbol{m}^{\mathrm{T}}|$, то

$$0 \leqslant \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} \leqslant |A\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{m}^{\mathrm{T}}| = |AP\boldsymbol{m}^{\mathrm{T}}| \leqslant |AP| \cdot |\boldsymbol{m}|.$$

Далее, $P = (H\mathbf{0}^T)X^{-1}$, откуда $|AP|^2 \leqslant |AH|^2|X^{-1}|^2$, причем $|X^{-1}|^2 = n - 1 + \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}/x_n^2$. Из этого следует второе утверждение леммы. □

Докажем теорему 7.20. Поскольку $\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} = 1$, $0 < x_1 < \ldots < x_n$, то $\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}} \leqslant nx_n^2$, откуда $\dfrac{1}{x_n^2} \leqslant n$. Положим $A = R_k^{-1}$. Пусть $\boldsymbol{x}\boldsymbol{m}^{\mathrm{T}} = 0$; матрица $PX = [H\mathbf{0}^T]$. Справедливо неравенство $|X^{-1}| \leqslant 2n - 1$. Тогда по леммам 7.27 и 7.31 получим, что

$$|AH|^2 \leqslant L(AH),$$

$$1 < 3n(\boldsymbol{m}\boldsymbol{m}^{\mathrm{T}})L(AH). \tag{7.46}$$

Положим $\varepsilon_n = \dfrac{1}{2^{n+1}n}$. По лемме 7.29

$$L(R_k^{-1}H) \leqslant (1 - \varepsilon_n)^k L(H).$$

Тогда по лемме 7.30

$$L(AH) = L(R_k^{-1}H) < (1 - \varepsilon_n)^k (\boldsymbol{x}\boldsymbol{x}^{\mathrm{T}})^2 n^2 = (1 - \varepsilon_n)^k n^2.$$

Из (7.46) теперь следует, что

$$1 < 3n^3(\boldsymbol{m}\boldsymbol{m}^{\mathrm{T}})(1 - \varepsilon_n)^k.$$

Так как для любого t, $0 < t < 1$, выполнено неравенство $e^{-t} > 1 - t$, то

$$1 < 3n^3(\boldsymbol{m}\boldsymbol{m}^{\mathrm{T}})e^{-k\varepsilon_n}.$$

Поэтому $k < \dfrac{\log(3n^3(\boldsymbol{m}\boldsymbol{m}^{\mathrm{T}}))}{\varepsilon_n} = 2^{n+1}n\log(3n^3M^2)$. Теорема 7.20 доказана.

Алгоритм 2 (нахождение соотношения).

На входе алгоритма задан нормализованный вектор $\boldsymbol{x} \in \mathbb{R}^n$. Алгоритм 2 состоит в нахождении с помощью алгоритма 1 последовательности $\boldsymbol{x}^{(k)}$, $H^{(k)}$, A_k из (7.24) и $R_k = A_1^{-1} \ldots A_k^{-1}$, $k = 1, 2, \ldots$

Если существует ненулевой вектор $\boldsymbol{m} \in \mathbb{Z}^n$ нормы M, удовлетворяющий (7.17), то при некотором k, удовлетворяющем (7.28), один из столбцов R_k даст нам соотношение.

Конец алгоритма.

Замечание 7.32. Лагариас и Хастад показали, что алгоритм 2 делает $O(2n^2 \log M + 20n^3)$ арифметических операций с действительными числами.

§ 7.7. Заключение

В данной главе мы описали некоторые виды приведенных базисов решеток. Основное внимание было уделено LLL-приведенному базису, алгоритму его построения и различным его приложениям. Одно из основных приложений LLL-приведенного базиса будет описано в следующей главе — это алгоритм разложения на неприводимые множители многочленов с рациональными коэффициентами, имеющий полиномиальную сложность от длины входа.

Заметим, что приложения LLL-алгоритма в линейной алгебре, описанные в данной главе, не всегда являются достаточно эффективными. Это видно и на примере алгоритма нахождения целочисленной линейной зависимости действительных чисел из § 7.5, и на примере алгоритма нахождения коротких векторов решеток из того же параграфа.

Нахождение целочисленной линейной зависимости для заданного набора действительных чисел имеет важные приложения в криптографии. В § 7.5 и § 7.6 мы описали два алгоритма для решения этой задачи. Еще один алгоритм можно найти в [129].

Глава 8. Факторизация многочленов над полем рациональных чисел с полиномиальной сложностью

§ 8.1. Введение

В данной главе мы рассматриваем алгоритмы разложения на неприводимые множители многочленов из $\mathbb{Z}[x]$. Центральное место в ней займет описание LLL-алгоритма факторизации многочленов, предложенного А. Ленстрой, Х. Ленстрой и Л. Ловасом в работе [160]. Этот алгоритм имеет полиномиальную сложность от длины входа. Мы также приводим и другие алгоритмы факторизации, эффективные на практике.

Мы будем называть многочлен $f(x) \in \mathbb{Z}[x]$ *примитивным*, если наибольший общий делитель всех его коэффициентов равен 1.

Разложение на неприводимые множители многочленов в кольце $\mathbb{Q}[x]$ сводится к разложению на неприводимые многочлены в кольце $\mathbb{Z}[x]$ с помощью леммы Гаусса. Пусть мы хотим разложить $f_0(x) \in \mathbb{Q}[x]$ на неприводимые множители в $\mathbb{Q}[x]$. Домножив $f_0(x)$ на общий знаменатель коэффициентов и вынося наибольший общий делитель получившихся целочисленных коэффициентов, мы сводим нашу задачу факторизации к задаче факторизации примитивного многочлена $f(x) \in \mathbb{Q}[x]$ на неприводимые многочлены в $\mathbb{Q}[x]$.

Лемма Гаусса. *Если* $f(x), g(x), h(x) \in \mathbb{Z}[x]$, $\deg g(x) \geqslant 1$, $\deg h(x) \geqslant 1$, $g(x)$ *и* $h(x)$ *— примитивные*, $f(x) = g(x) \cdot h(x)$, *то* $f(x)$ *также примитивен.*

Доказательство. Пусть $g(x) = \sum\limits_{i=0}^{l} b_i x^i$, $h(x) = \sum\limits_{j=0}^{m} c_j x^j$, $f(x) = \sum\limits_{k=0}^{n} a_k x^k$. Тогда

$$a_k = \sum_{\substack{i+j=k,\\ 0 \leqslant i \leqslant l,\\ 0 \leqslant j \leqslant m}} b_i c_j, \qquad 0 \leqslant k \leqslant n. \tag{8.1}$$

Предположим, что $f(x)$ не примитивен. Тогда найдется простое число p, делящее все a_k, $k = 0, \ldots, n$. В силу примитивности $g(x)$ найдется номер i_0, такой, что $p \mid b_i$ при всех $i > i_0$, $p \nmid b_{i_0}$. Аналогично, найдется

номер j_0, такой, что $p \mid c_j$ при всех $j > j_0$, $p \nmid c_{j_0}$. Пусть $k_0 = i_0 + j_0$. Но тогда в силу (8.1) число p не делит

$$a_{k_0} = b_{i_0} c_{j_0} + \sum_{\substack{i+j=i_0+j_0,\\ \text{либо } i>i_0,\\ \text{либо } j>j_0}} b_i c_j,$$

поскольку первое слагаемое в правой части этого равенства не делится на p, а второе — делится. □

Лемма 8.2. *Пусть $f(x) \in \mathbb{Z}[x]$, $f(x)$ примитивен и неприводим в $\mathbb{Z}[x]$. Тогда $f(x)$ неприводим и в $\mathbb{Q}[x]$.*

Доказательство. Предположим, что $f(x) = g(x) \cdot h(x)$, где $g(x)$, $h(x) \in \mathbb{Q}[x]$, $\deg g(x) \geqslant 1$, $\deg h(x) \geqslant 1$. Тогда, вынося общие знаменатели коэффициентов $g(x)$ и $h(x)$ и затем вынося наибольшие общие делители получившихся целочисленных коэффициентов, мы можем представить наше разложение в виде

$$f(x) = \frac{A}{B} g_1(x) h_1(x),$$

где $A \in \mathbb{Z}$, $B \in \mathbb{N}$, $(A, B) = 1$, $g_1(x)$, $h_1(x) \in \mathbb{Z}[x]$, $g_1(x)$ и $h_1(x)$ — примитивные. Отсюда

$$Bf(x) = Ag_1(x)h_1(x). \tag{8.2}$$

Поскольку из этого равенства следует, что B делит все коэффициенты многочлена $Ag_1(x)h_1(x)$, в силу примитивности $g_1(x)h_1(x)$ (лемма Гаусса) и равенства $(A, B) = 1$ получаем, что $B = 1$. Но тогда из (8.2) следует, что $f(x)$ приводим в $\mathbb{Z}[x]$, что противоречит условию леммы. □

Итак, мы свели задачу факторизации многочлена $f_0 \in \mathbb{Q}[x]$ к задаче факторизации примитивного многочлена $f(x) \in \mathbb{Z}[x]$ на неприводимые множители в кольце $\mathbb{Z}[x]$. Обозначим через

$$f(x) = \sum_{i=0}^{n} \hat{a}_i x_i, \qquad n = \deg f(x) \geqslant 2, \tag{8.3}$$

где $\hat{a}_i \in \mathbb{Z}$, $i = 0, \ldots, n-1$, $\hat{a}_n \in \mathbb{N}$, числа $\hat{a}_0, \ldots, \hat{a}_n$ взаимно просты в совокупности. Эти обозначения и условия мы будем использовать в §§ 8.2—8.5.

Нормой многочлена $h(x) = \sum\limits_{i=0}^{m} h_i x^i \in \mathbb{Q}[x]$ мы будем называть величину $|h| = \sqrt{\sum\limits_{i=0}^{m} h_i^2}$ — евклидову длину вектора коэффициентов многочлена.

Мы также будем переходить от многочленов из кольца $\mathbb{Z}[x]$ к многочленам из кольца $\mathbb{Z}/l\mathbb{Z}[x]$, где l — какое-либо натуральное число; для многочлена $h(x)=\sum\limits_{i=0}^{m} h_i x^i \in \mathbb{Z}[x]$ мы будем обозначать через $h(x) \pmod l$ многочлен $\sum\limits_{i=0}^{m} h_i \pmod l x^i \in \mathbb{Z}/l\mathbb{Z}[x]$. Соответственно, запись $h(x) \pmod l \mid g(x) \pmod l$ означает делимость в $\mathbb{Z}/l\mathbb{Z}[x]$.

Также напомним, что кольцо многочленов $\mathbb{Z}[x]$ является факториальным (см., например, [27, гл. 9]).

§ 8.2. LLL-алгоритм факторизации: разложение по простому модулю

Пусть примитивный многочлен $f(x)$ тот же, что в формуле (8.3) из § 8.1. Предположим, что задано простое число p, натуральное число k и многочлен $h(x) \in \mathbb{Z}[x]$, обладающие следующими свойствами:

А) старший коэффициент $h(x)$ равен 1;

Б) $h(x) \pmod{p^k}$ делит $f(x) \pmod{p^k}$ в кольце $\mathbb{Z}/p^k\mathbb{Z}[x]$;

В) $h(x) \pmod p$ неприводим в $\mathbb{Z}/p\mathbb{Z}[x]$;

Г) $(h(x) \pmod p)^2 \nmid f(x) \pmod p$ в $\mathbb{Z}/p\mathbb{Z}[x]$.

Лемма 8.3. *Существует единственный примитивный неприводимый многочлен $h_0(x) \in \mathbb{Z}[x]$ такой, что $h_0(x)$ делит $f(x)$ в $\mathbb{Z}[x]$, $h(x) \pmod p \mid h_0(x) \pmod p$. При этом для многочленов $g(x) \in \mathbb{Z}[x]$, делящих $f(x)$ в $\mathbb{Z}[x]$, эквивалентны следующие условия*:

1) $h(x) \pmod p \mid g(x) \pmod p$;

2) $h(x) \pmod{p^k} \mid g(x) \pmod{p^k}$;

3) $h_0(x)$ *делит* $g(x)$ *в* $\mathbb{Z}[x]$.

Доказательство. Из свойства Б) следует, что $h(x) \pmod p$ делит $f(x) \pmod p$ в $\mathbb{Z}/p\mathbb{Z}[x]$. Если мы разложим $f(x)$ на неприводимые множители в $\mathbb{Z}[x]$ (они также будут примитивными в силу примитивности $f(x)$), то один из них, взятый по модулю p, делится на $h(x) \pmod p$. В силу свойства Г) такой неприводимый делитель $f(x)$ будет единственен; его и обозначим через $h_0(x)$.

Теперь докажем эквивалентность условий 1)—3). Очевидно, что из третьего условия следует первое, и что из второго условия также следует первое.

Покажем, что из первого условия следует второе.

Пусть $h(x) \pmod p \mid g(x) \pmod p$. Тогда в силу свойства Г)

$$h(x) \pmod p \nmid \left(\frac{f(x)}{g(x)}\right) \pmod p.$$

Поскольку $\mathbb{Z}/p\mathbb{Z}$ является полем и $h(x) \pmod p$ неприводим, $h(x) \pmod p$ и $\left(\frac{f(x)}{g(x)}\right) \pmod p$ взаимно просты в $\mathbb{Z}/p\mathbb{Z}[x]$. Следовательно, найдутся многочлены $\lambda(x)$, $\mu(x) \in \mathbb{Z}[x]$ такие, что

$$\lambda(x) \pmod p h(x) \pmod p + \mu(x) \pmod p \left(\frac{f(x)}{g(x)}\right) \pmod p = 1,$$

или, эквивалентно,

$$\lambda(x)h(x) + \mu(x)\frac{f(x)}{g(x)} = 1 - p\nu(x)$$

для некоторого $\nu(x) \in \mathbb{Z}[x]$. Умножая это равенство на $(1 + p\nu(x) + \ldots + p^{k-1}\nu(x)^{k-1})g(x)$, получим, что

$$\lambda_1(x)h(x) + \mu_1(x)f(x) = (1 - p^k\nu(x)^k)g(x),$$

где $\lambda_1(x)$, $\mu_1(x) \in \mathbb{Z}[x]$. Отсюда

$$\lambda_1(x) \pmod{p^k} h(x) \pmod{p^k} + \mu_1(x) \pmod{p^k} f(x) \pmod{p^k} = g(x) \pmod{p^k},$$

и в силу Б) получаем, что $h(x) \pmod{p^k} | g(x) \pmod{p^k}$.

Теперь докажем, что из первого условия следует третье. Из свойства Г) вытекает, что $h(x) \pmod p \Big| \frac{f(x)}{g(x)} \pmod p$. Поэтому $h_0(x) \Big| \frac{f(x)}{g(x)}$ в $\mathbb{Z}[x]$. Поскольку $h_0(x)$ неприводим, $h_0(x) \mid g(x)$. □

В следующих параграфах мы подходящим образом выберем p, k и $h(x)$ так, чтобы выполнялись свойства А)—Г). Это приведет нас в конечном счете к нахождению разложения $f(x)$ на множители в $\mathbb{Z}[x]$.

§ 8.3. LLL-алгоритм факторизации: использование решеток

Мы предполагаем, что $f(x)$, p, k и $h(x) \in \mathbb{Z}[x]$ — те же, что и в § 8.2, и выполняются свойства А)—Г). Обозначим $l = \deg h(x)$. Тогда $l \leqslant n$ в силе свойств А), Б) и примитивности $f(x)$; равенство $l = n$ будет по лемме 8.3 означать, что $f(x)$ неприводим.

Фиксируем натуральный параметр m, $m \geqslant l$.

Рассмотрим множество

$$L = \left\{P(x) \in \mathbb{Z}[x] \;\middle|\; \deg P \leqslant m,\ h(x) \pmod{p^k} \mid P(x) \pmod{p^k}\right\},$$

и отображение

$$\Phi_0 \colon L \to \mathbb{Z}^{m+1}, \quad \Phi_0\left(\sum_{i=0}^{m} z_i x^i\right) = (a_0, a_1, \ldots, a_m).$$

Очевидно, что $\widetilde{L} = \Phi_0(L) \subseteq \mathbb{Z}^{m+1}$ является аддитивной подгруппой $\mathbb{Z}^{m+1}$. Мы далее покажем, что $\widetilde{L}$ на самом деле является решеткой в $\mathbb{Z}^{m+1}$, т. е., состоит из всех целочисленных линейных комбинаций некоторых $m+1$ линейно независимых векторов.

Заметим, что $|P(x)| = |\Phi_0(P(x))|$, где слева стоит рассматриваемая нами норма многочленов (см. § 8.1), а справа — евклидова норма векторов в $\mathbb{R}^{m+1}$.

Образующими $\widetilde{L}$ над $\mathbb{Z}$ являются следующие векторы:

$$(0, \ldots, 0, p^k, 0, \ldots, 0) = \Phi_0(p^k \cdot x^i), \quad i = 0, 1, \ldots, l-1, \tag{8.4}$$

и

$$\Phi_0(h(x)x^{i-l}), \quad i = l, \ldots, m. \tag{8.5}$$

Действительно, если $P(x) \in L$, то $P(x) \pmod{p^k} = h(x) \pmod{p^k} \cdot Q(x) \pmod{p^k}$ для некоторого $Q(x) \in \mathbb{Z}[x]$. Отсюда $P(x) = h(x)Q(x) + p^k \cdot T(x)$ при некотором $T(x) \in \mathbb{Z}[x]$, и можно считать, что $\deg T(x) < \deg h(x) = l$, так как $h(x)$ унитарен. Но тогда, если $Q(x) \neq 0$, то $\deg Q(x) = \deg P(x) - \deg h(x) \leqslant m - l$. Кроме того, все многочлены указанного вида $P(x) = h(x)Q(x) + p^k T(x) \in \mathbb{Z}[x]$, имеющие степень не выше m, содержатся в L. Теперь очевидно, что векторы (8.4) и (8.5) образуют $\widetilde{L}$. Их линейная независимость над $\mathbb{Z}$ и над $\mathbb{Q}$ также очевидна: матрица, строки которой образуют векторы (8.4) и (8.5), является треугольной, и на ее диагонали стоят числа p^k (l раз) и единицы ($m-l+1$ раз). Поэтому $\widetilde{L}$ является решеткой, и ее определитель (см. § 7.1) равен

$$d(L) = p^{kl}. \tag{8.6}$$

Лемма 8.4. *Пусть многочлен $h_0(x)$ — из утверждения леммы 8.3. Пусть $b(x) \in L$, причем выполнено неравенство*

$$p^{kl} > |f(x)|^m \cdot |b(x)|^n.$$

Тогда $h_0(x)$ делит $b(x)$ в $\mathbb{Z}[x]$.

Доказательство. Пусть $b(x) \neq 0$ (иначе утверждение тривиально). Пусть $g(x) = \text{НОД}(f(x), b(x))$ — наибольший общий делитель в $\mathbb{Z}[x]$. По лемме 8.3 достаточно доказать, что $h(x) \pmod{p} | g(x) \pmod{p}$. Если мы предположим, что $h(x) \pmod{p} \nmid g(x) \pmod{p}$, то в силу свойства B) из § 8.2 будет выполнено равенство

$$\lambda_3(x)h(x) + \mu_3(x)g(x) = 1 - p\nu_3(x) \tag{8.7}$$

при некоторых $\lambda_3(x), \mu_3(x), \nu_3(x) \in \mathbb{Z}[x]$. Обозначим $m_1 = \deg b(x)$,

$m_2 = \deg g(x)$. Тогда $m \geqslant m_1 \geqslant m_2 \geqslant 0$. Рассмотрим множество

$$M = \{\lambda f + \mu b \mid \lambda, \mu \in \mathbb{Z}[x],\ \deg\lambda < m_1 - m,\ \deg\mu < n - m_2\},$$

где $f = f(x)$, $b = b(x)$. Степень многочленов из M не превосходит $n + m_1 - m_2 - 1$.

Докажем, что если $\lambda f + \mu b \in M$ и $\deg(\lambda f + \mu b) < m_2$, то $\lambda = \mu = 0$. Действительно, так как $g(x) = g|\lambda f + \mu b$, $\deg g = m_2$, то $\lambda f + \mu b = 0$, $\lambda\dfrac{f}{g} = -\mu\dfrac{b}{g}$. Но НОД$\left(\dfrac{f}{g}, \dfrac{b}{g}\right) = 1$; поэтому $\dfrac{f}{g} \,\Big|\, \mu$ в $\mathbb{Z}[x]$. Так как $\deg\mu < n - m_2 = \deg f - \deg g$, то $\mu = 0$; тогда и $\lambda = 0$.

Рассмотрим отображение

$$\Phi\colon M \to \mathbb{Z}^{n+m_1-2m_2},$$

$$\Phi\left(\sum_{i=0}^{n+m_1-m_2-1} a_i x^i\right) = (a_{m_2}, a_{m_2+1}, \ldots, a_{n+m_1-m_2-1}).$$

По доказанному выше Φ является вложением и гомоморфизмом аддитивных групп. Также по доказанному векторы

$$\Phi(x^i f(x)), \quad i = 0, 1, \ldots, m_1 - m_2 - 1, \tag{8.8}$$

$$\Phi(x^j b(x)), \quad j = 0, 1, \ldots, n - m_2 - 1, \tag{8.9}$$

являются линейно независимыми над $\mathbb{Z}$ и порождают $\Phi(M)$. Значит, $\Phi(M)$ является решеткой в $\mathbb{Z}^{n+m_1-2m_2}$. По неравенству Адамара и условию леммы определитель $d(\Phi(M))$ решетки $\Phi(M)$ удовлетворяет неравенству

$$d(\Phi(M)) \leqslant |f(x)|^{m_1-m_2}|b(x)|^{n-m_2} \leqslant |f|^m|b|^n < p^{kl}.$$

Покажем теперь, что выполняется и обратное неравенство; полученное противоречие будет доказывать утверждение нашей леммы.

Докажем включение множеств

$$\{\nu(x) \in M \mid \deg\nu(x) < m_2 + l\} \subseteq p^k \cdot \mathbb{Z}[x]. \tag{8.10}$$

Пусть $\nu(x) \in M$, $\deg\nu(x) < m_2 + l$. Тогда $\dfrac{\nu(x)}{g(x)} \in \mathbb{Z}[x]$. Умножим равенство (8.7) на $\dfrac{\nu(x)}{g(x)}$ и на $1 + p\nu_3(x) + \ldots + p^{k-1}\nu_3^{k-1}(x)$. Получим, что

$$\lambda_4(x)h(x) + \mu_4(x)\nu(x) - \frac{\nu(x)}{g(x)} \in\ p^k\mathbb{Z}[x]. \tag{8.11}$$

Поскольку $\nu(x) \in M$, $b(x) \in L$, то $h(x) \pmod{p^k}$ делит $\nu(x) \pmod{p^k}$. Тогда из (8.11) следует, что

$$h(x) \pmod{p^k} \mid \frac{\nu(x)}{g(x)} \pmod{p^k}.$$

Но $\deg \dfrac{\nu(x)}{g(x)} < m_2 + l - m_2 = l = \deg h(x) = \deg(h(x) \pmod{p^k})$. Значит, $\dfrac{\nu(x)}{g(x)} \in p^k\mathbb{Z}[x]$, откуда следует (8.10).

Хорошо известно, что в любой решетке, содержащейся в $\mathbb{Z}^N$, можно выбрать треугольный базис (см. [24, гл. 1]). Выберем такой базис в $\Phi(M) \subseteq \mathbb{Z}^{n+m_1-2m_2}$. Обозначим его элементы через $\boldsymbol{b}_i = (b_{i1}, \ldots, b_{ii}, 0, \ldots, 0)$, где $b_{ij} \in \mathbb{Z}$, $b_{ii} \neq 0$, $i = 1, \ldots, n + m_1 - 2m_2$. При этом $d(\Phi(M)) = \prod\limits_{i=1}^{n+m_1-2m_2} |b_{ii}|$. Прообразы $\Phi^{-1}(\boldsymbol{b}_i)$ векторов $\boldsymbol{b}_i$ являются многочленами из M степени $i + m_2 - 1$, $i = 1, \ldots, n + m_1 - 2m_2$. Из формулы (8.10) следует тогда, что $b_{ii} \equiv 0 \pmod{p^k}$ для $i = 1, \ldots, l$. Поэтому $d(\Phi(M)) \geqslant p^{kl}$. $\square$

Лемма 8.5. *Пусть $h_0(x)$ — неприводимый делитель $f(x)$ такой, что $h(x) \pmod p \mid h_0(x) \pmod p$. Пусть $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{m+1}$ — приведенный базис решетки $\widetilde{L} = \Phi_0(L)$, и пусть выполнено неравенство*

$$p^{kl} > 2^{mn/2}(n+1)^{n/2}e^{n^2}|f|^{n+m}. \tag{8.12}$$

Степень многочлена $h_0(x)$ не превосходит m тогда и только тогда, когда

$$|\boldsymbol{b}_1| < \left(\frac{p^{kl}}{|f|^m}\right)^{1/n}. \tag{8.13}$$

Доказательство. Из (8.13) следует, что $|\boldsymbol{b}_1|^n|f|^m < p^{kl}$, т. е. выполнены условия леммы 8.4 для многочлена $b(x) = \Phi_0(\boldsymbol{b}_1) \in L$. По определению L выполнено неравенство $\deg b(x) \leqslant m$. Из леммы 8.4 следует, что $h_0(x) \mid b(x)$, поэтому $\deg h_0(x) \leqslant m$.

Пусть $\deg h_0(x) \leqslant m$. Тогда $h_0(x) \in L$. Поскольку $h_0(x)$ делит $f(x)$, выполнено неравенство

$$|h_0(x)| \leqslant \sqrt{n+1}e^n|f(x)| \tag{8.14}$$

(доказательство этого неравенства см. в работе [183]; оно оценивает величину коэффициентов многочлена-делителя через коэффициенты делимого). Применим теорему 7.8 к вектору $\boldsymbol{\xi} = \Phi_0(h_0(x)) \in \widetilde{L} = \Phi_0(L)$. Поскольку $\boldsymbol{\xi} \neq 0$ и $\widetilde{L} \in \mathbb{R}^{m+1}$, то

$$\boldsymbol{b}_1 \leqslant 2^{m/2}|\boldsymbol{\xi}| = 2^{m/2}|h_0(x)| \leqslant 2^{m/2}\sqrt{n+1}e^n|f(x)|.$$

Отсюда

$$|\boldsymbol{b}_1|^n|f(x)|^m \leqslant 2^{mn/2}(n+1)^{n/2}\ e^{n^2}|f|^{m+n}.$$

Из (8.12) тогда получим, что

$$|\boldsymbol{b}_1|^n |f(x)|^m < p^{kl},$$

откуда следует (8.13). □

Лемма 8.6. *В обозначениях и условиях предыдущей леммы предположим дополнительно, что*

$$t = \max\left\{ j \;\middle|\; |\boldsymbol{b}_j| < \left(\frac{p^{kl}}{|f(x)|^m} \right)^{1/n} \right\} \geqslant 1. \tag{8.15}$$

Тогда $\deg h_0(x) = m + 1 - t$ *и*

$$h_0(x) = \text{НОД}\left(\frac{\Phi_0^{-1}(\boldsymbol{b}_1),\ \ldots,\ \Phi_0^{-1}(\boldsymbol{b}_t)}{p^\tau} \right), \tag{8.16}$$

где τ *— наибольшее целое неотрицательное число, для которого* p^τ *делит все коэффициенты* $\text{НОД}(\Phi_0^{-1}(\boldsymbol{b}_1),\ \ldots,\ \Phi_0^{-1}(\boldsymbol{b}_t)) \in \mathbb{Z}[x]$.

Доказательство. Рассмотрим множество $J = \left\{ j \mid 1 \leqslant j \leqslant m+1,\ |\boldsymbol{b}_j| < \left(\frac{p^{kl}}{|f|^m} \right) \right\} \neq \varnothing$. По лемме 8.4 для $j \in J$ $h_0(x) \mid \Phi_0^{-1}(\boldsymbol{b}_j)$. Пусть $h_1(x) = \text{НОД}_{j \in J}(\Phi_0^{-1}(\boldsymbol{b}_j))$; тогда $h_0(x) \mid h_1(x)$. Докажем, что $J = \{1, \ldots, t\}$ и что $h_0(x) = \pm h_1(x)/p^\tau$, где τ из (8.16).

Поскольку $\deg \Phi_0^{-1}(\boldsymbol{b}_j) \leqslant m$ для $j = 1, \ldots, m-1$, и многочлены $\Phi_0^{-1}(\boldsymbol{b}_j)$ линейно независимы над $\mathbb{Z}$, то

$$|J| \leqslant m + 1 - \deg h_1(x). \tag{8.17}$$

Далее, $|h_0(x) \cdot x^j| = |h_0(x)| \leqslant \sqrt{n+1}\, e^n |f(x)|$ в силу (8.14), и для $i = 0, 1, \ldots, m - \deg h_0(x)$ многочлены $h_0(x) \cdot x^i$ содержатся в L и линейно независимы. Тогда по теореме 7.10 при $j = 1, 2, \ldots, m + 1 - \deg h_0(x)$ выполнены неравенства

$$|\boldsymbol{b}_j| \leqslant 2^{m/2} \max\{|\Phi_0(h_0(x))|, |\Phi_0(h_0(x) \cdot x)|, \ldots, |\Phi_0(h_0(x) \cdot x^{m - \deg h_0(x)})|\} =$$
$$= 2^{m/2} |\Phi_0(h_0(x))| = 2^{m/2} |h_0(x)| \leqslant 2^{m/2} \sqrt{n+1}\, e^n |f(x)|.$$

Отсюда, аналогично доказательству предыдущей леммы, получим

$$|\boldsymbol{b}_j|^n |f(x)|^m \leqslant p^{kl} \tag{8.18}$$

для $j = 1, \ldots, m + 1 - \deg h_0(x)$. Поэтому $1, 2, \ldots, m + 1 - \deg h_0(x) \in J$. Следовательно, $m + 1 - \deg h_0(x) \leqslant |J| \leqslant m + 1 - \deg h_1(x)$; значит, $\deg h_1(x) \leqslant \deg h_0(x)$. Поскольку $h_0(x) | h_1(x)$ в $\mathbb{Z}[x]$, то

$$h_1(x) = d h_0(x), \quad \text{где} \quad d \in \mathbb{Z},\ d \neq 0. \tag{8.19}$$

Мы доказали, что $|J| = m + 1 - \deg h_0(x)$, $J = \{1, \ldots, m + 1 - \deg h_0(x)\}$, и что $t = |J|$, т. е. $\deg h_0(x) = m + 1 - t$.

Теперь докажем, что $h_1(x)$ в (8.19) является почти примитивным, т. е. наибольший общий делитель его коэффициентов может быть равен лишь некоторой степени числа p; из этого будет следовать (8.16). Предположим противное, т. е. что найдется простое число q, $q \neq p$, такое, что $h_1(x)/q \in \mathbb{Z}[x]$. По определению $h_1(x)$ получим, что вектор $\frac{1}{q}\boldsymbol{b}_1$ лежит в $\widetilde{L}$, так как $1 \in J$. Пусть $\boldsymbol{b}_1 = (b_{10}, \ldots, b_{1m}) \in \mathbb{Z}^{m+1}$. Тогда $b_{1i} = qb'_{1i}$, где $b'_{1i} \in \mathbb{Z}$, $i = 0, \ldots, m$. Так как $\boldsymbol{b}_1 \in \widetilde{L}$, то

$$h(x) \pmod{p^k} \,\Big|\, q\left(\sum_{i=0}^{m} b'_{1i}x^i \pmod{p^k}\right).$$

Поскольку $q \neq p$, то отсюда следует, что

$$h(x) \pmod{p^k} \,\Big|\, \left(\sum_{i=0}^{m} b'_{1i}x^i \pmod{p^k}\right),$$

т. е. $(b'_{10}, \ldots, b'_{1m}) \in \widetilde{L}$. Но это невозможно, так как $\boldsymbol{b}_1$ является элементом базиса решетки $\widetilde{L}$. □

§ 8.4. LLL-алгоритм факторизации: подъем разложения

В этом параграфе мы покажем, как соотношение

$$u_0(x)v_0(x) \equiv w(x) \pmod{p^m},$$

где p — простое число, $m \in \mathbb{N}$, $u_0(x)$, $v_0(x)$, $w_0(x) \in \mathbb{Z}[x]$, может быть преобразовано в соотношение

$$u_2(x)v_2(x) \equiv w(x) \pmod{p^{m_1}}$$

для некоторых эффективно вычислимых многочленов $u_2(x)$, $v_2(x) \in \mathbb{Z}[x]$ и некоторого $m_1 > m$.

В этом параграфе штрих не является знаком производной, а служит для обозначения какого-либо другого многочлена.

Для начала рассмотрим произвольное поле $\mathbb{K}$. Пусть $g(x)$, $h(x) \in \in \mathbb{K}[x]$, $\deg g(x) \geqslant \deg h(x) \geqslant 1$, $d(x) = \text{НОД}(g(x), h(x))$.

Алгоритм 1.

На входе алгоритма заданы $u(x)$, $v(x) \in \mathbb{Z}[x]$. На выходе получается представление $u(x)g(x) + v(x)h(x) = d(x)$.

1 шаг. Присвоить $(u_1, u_2, u_3) := (1, 0, g(x))$, $(v_1, v_2, v_3) := (0, 1, h(x))$.

2 шаг. Если $v_3 = 0$, то выдать значение $u(x) = u_1$, $v(x) = u_2$, $d(x) = u_3$ и закончить работу.

3 шаг. Разделить с остатком:

$$u_3 = qv_3 + r, \quad \deg r < \deg v_3.$$

4 шаг. Присвоить $(t_1, t_2, t_3) := (u_1, u_2, u_3) - q(v_1, v_2, v_3)$, $(u_1, u_2, u_3) := (v_1, v_2, v_3)$, $(v_1, v_2, v_3) := (t_1, t_2, t_3)$ и вернуться на 2 шаг.

Конец алгоритма.

Легко видеть, что найденные многочлены $u(x)$, $v(x)$, $d(x)$ действительно удовлетворяют равенству $u(x)g(x) + v(x)h(x) = d(x) = \text{НОД}(g(x), h(x))$. Это проверяется в точности так же, как в обычном алгоритме Евклида для целых чисел (см. Приложение). Можно показать (см. [25, § 4.6.1, упр. 3]), что $\deg v(x) < \deg g(x)$, $\deg u(x) < \deg h(x)$.

Теперь рассмотрим поле $\mathbb{K} = \mathbb{Z}/p\mathbb{Z}$.

Алгоритм 2.

На входе задано простое число p, $j \in \mathbb{N}$, многочлены $a(x)$, $b(x)$, $c(x)$, $g_j(x)$, $h_j(x) \in \mathbb{Z}/p\mathbb{Z}[x]$, такие, что старший коэффициент $h_j(x)$ обратим в $\mathbb{Z}/p^j\mathbb{Z}$ и $a(x)g_j(x) + b(x)h_j(x) = 1$ в $\mathbb{Z}/p^j\mathbb{Z}[x]$. На выходе получаем многочлены $a'(x)$, $b'(x) \in \mathbb{Z}/p^j\mathbb{Z}$ такие, что $a'(x)g_j(x) + b'(x)h_j(x) = c(x)$ в $\mathbb{Z}/p^j\mathbb{Z}[x]$, и при этом $\deg a'(x) < \deg h_j(x)$.

1 шаг. Ввиду обратимости старшего коэффициента $h_j(x)$ можно произвести деление с остатком:

$$a(x)c(x) = h_j(x) \cdot q(x) + r(x), \quad \deg r(x) < \deg h_j(x).$$

2 шаг. Присвоить

$$a'(x) := r(x), \quad b'(x) := b(x)c(x) + g_j(x)q(x).$$

Конец алгоритма.

Корректность работы алгоритма следует из равенств

$$\begin{aligned} g_j(x)a'(x) + h_j(x)b'(x) &= \\ &= g_j(x)(a(x)c(x) - h_j(x)q(x)) + h_j(x)(b(x)c(x) + g_j(x)q(x)) = \\ &= c(x)(g_j(x)a(x) + h_j(x)b(x)) = c(x). \end{aligned}$$

Лемма 8.7. *Пусть p — простое число, $j \in \mathbb{N}$, $t(x) \in \mathbb{Z}/p^{2j}\mathbb{Z}[x]$. Пусть $g_j(x)$, $h_j(x) \in \mathbb{Z}/p^j\mathbb{Z}[x]$, $t(x) \equiv g_j(x)h_j(x) \pmod{p^j}$, старший коэффициент $h_j(x)$ обратим в $\mathbb{Z}/p^j\mathbb{Z}[x]$ и существуют многочлены $a_j(x)$, $b_j(x) \in \mathbb{Z}/p^j\mathbb{Z}[x]$ такие, что $a_j(x)g_j(x) + b_j(x)h_j(x) = 1$*

в $\mathbb{Z}/p^j\mathbb{Z}[x]$. *Тогда можно определить многочлены* $a_{2j}(x)$, $b_{2j}(x)$, $g_{2j}(x)$, $h_{2j}(x) \in \mathbb{Z}/p^{2j}\mathbb{Z}[x]$ *такие, что*

$$t(x) \equiv g_{2j}(x)h_{2j}(x) \pmod{p^{2j}},$$
$$a_{2j}(x)g_{2j}(x) + b_{2j}(x)h_{2j}(x) = 1 \quad \text{в } \mathbb{Z}/p^{2j}\mathbb{Z}[x],$$
$$g_{2j}(x) \equiv g_j \pmod{p^j}, \quad h_{2j}(x) \equiv h_j(x) \pmod{p^j},$$
$$\deg h_{2j}(x) = \deg h_j(x).$$

Доказательство. Определим многочлен $c_j(x) \in \mathbb{Z}/p^j\mathbb{Z}[x]$ при помощи равенства

$$t(x) - g_j(x)h_j(x) = p^j c_j(x).$$

С помощью алгоритма 2 вычислим $a'_j(x)$, $b'_j(x)$ такие, что

$$a'_j(x)g_j(x) + b'_j(x)h_j(x) = c_j(x) \quad \text{в } \mathbb{Z}/p^j\mathbb{Z}[x], \quad \deg a'_j(x) < \deg h_j(x).$$

Положим $g_{2j}(x) = g_j(x) + p^j b'_j(x)$, $h_{2j}(x) = h_j(x) + p^j a'_j(x)$, $g_{2j}(x)$, $h_{2j}(x) \in \in \mathbb{Z}/p^{2j}\mathbb{Z}[x]$. Тогда очевидно, что $\deg h_{2j}(x) = \deg h_j(x)$ и что $g_{2j}(x) \equiv g_j(x) \pmod{p^j}$, $h_{2j}(x) \equiv h_j(x) \pmod{p^j}$. Далее,

$$g_{2j}(x)h_{2j}(x) \equiv g_j(x)h_j(x) + p^j(a'_j(x)g_j(x) + b'_j(x)h_j(x)) \pmod{p^{2j}} \equiv$$
$$\equiv t(x) - p^j c_j(x) + p^j c_j(x) \pmod{p^{2j}} \equiv t(x) \pmod{p^{2j}}.$$

Определим многочлен $c_{1j}(x) \in \mathbb{Z}/p^j\mathbb{Z}[x]$ равенством

$$g_{2j}(x)a_j(x) + h_{2j}(x)b_j(x) = 1 + p^j c_{1j}(x) \quad \text{в } \mathbb{Z}/p^{2j}\mathbb{Z}[x].$$

С помощью алгоритма 2 найдем $a''_j(x)$, $b''_j(x)$ такие, что в $\mathbb{Z}/p^j\mathbb{Z}[x]$ выполнено равенство

$$g_j(x)a''_j(x) + h_j(x)b''_j(x) = c_{1j}(x).$$

Положим

$$a_{2j}(x) = a_j(x) - p^j a''_j(x) \in \mathbb{Z}/p^{2j}\mathbb{Z}[x],$$
$$b_{2j}(x) = b_j(x) - p^j b''_j(x) \in \mathbb{Z}/p^{2j}\mathbb{Z}[x].$$

Тогда

$$g_{2j}(x)a_{2j}(x) + h_{2j}(x)b_{2j}(x) = g_{2j}(x)a_j(x) + h_{2j}(x)b_j(x) - p^j c_{1j}(x) = 1$$

в $\mathbb{Z}/p^{2j}\mathbb{Z}[x]$. Лемма доказана. □

Замечание 8.8. В лемме описан квадратичный подъем, т. е. переход от соотношений по модулю p^j к соотношениям по модулю p^{2j}. Можно

описать также линейный подъем, т. е. переход от соотношений по модулю p^j к соотношениям по модулю p^{j+1} (см. [1, гл. 6; 89, гл. 3]). Однако если мы хотим получить из соотношений по модулю p соотношения по модулю p^k, где k достаточно велико, то лучше использовать квадратичный подъем, так как он быстрее приведет нас к требуемому результату.

§ 8.5. LLL-алгоритм факторизации: полное описание

Теперь мы можем детально описать LLL-алгоритм факторизации многочленов в кольце $\mathbb{Z}[x]$.

Алгоритм (LLL-факторизация).

На входе алгоритма задан примитивный многочлен $f_0(x) \in \mathbb{Z}[x]$, $\deg f_0 = n_0 > 1$. На выходе получается разложение $f_0(x)$ на неприводимые множители в $\mathbb{Z}[x]$.

1 шаг. Вычисляем $g(x) = \text{НОД}(f_0(x), f_0'(x)) \in \mathbb{Z}[x]$, где $f_0'(x)$ — производная. Полагаем $f(x) = \dfrac{f_0(x)}{g(x)}$. Заметим, что, если

$$f_0(x) = f_1(x)^{\alpha_1} \dots f_k(x)^{\alpha_k} \tag{8.20}$$

— разложение $f_0(x)$ на неприводимые множители в $\mathbb{Z}[x]$, то оно же есть разложение $f_0(x)$ на неприводимые множители в $\mathbb{Q}[x]$ (см. лемму 8.2). Поэтому $g(x) = f_1(x)^{\alpha_1 - 1} \dots f_k(x)^{\alpha_k - 1}$ есть примитивный многочлен, и его можно найти с помощью алгоритма Евклида в $\mathbb{Q}[x]$. Тогда $f(x) = f_1(x) \dots f_k(x)$ — многочлен, не имеющий кратных неприводимых множителей.

Обозначим $n = \deg f(x)$. Далее мы считаем, что $n \geqslant 2$ ($f(x)$ неприводим при $n = 1$ и $f_0(x) = f(x)^{\alpha}$, где $\alpha = \deg f_0(x)$).

2 шаг. Вычисляем результант $\text{Res}(f, f') \in \mathbb{Z}$; $\text{Res}(f, f') \neq 0$ (так как f и f' взаимно просты). Определение и свойства результанта см. в [9, гл. 5].

3 шаг. Находим наименьшее простое число p, такое, что $p \nmid \text{Res}(f, f')$. Тогда p не делит старший коэффициент $f(x)$, и $f(x) \pmod p \in \mathbb{Z}/p\mathbb{Z}[x]$ не имеет кратных неприводимых делителей. Можно доказать (см. [160]), что p удовлетворяет неравенству

$$p \leqslant \max\left(101,\ \frac{n \log n + (2-1)\log |f|}{0{,}84}\right),$$

поэтому мы быстро найдем его, перебирая все простые числа подряд.

4 шаг. Раскладываем $f(x) \pmod p$ в $\mathbb{Z}/p\mathbb{Z}[x]$ на неприводимые унитарные множители с помощью алгоритма Берлекэмпа. Пусть S — список этих множителей. Заводим список S_1 неприводимых делителей $f(x)$ в $\mathbb{Z}[x]$, $S_1 := \varnothing$. Полагаем $F(x) = f(x)$, $N = \deg f(x)$. Если $|S| = 1$, то $f(x)$ неприводим в $\mathbb{Z}[x]$ (так как неприводим в $\mathbb{Z}/p\mathbb{Z}[x]$).

5 шаг. Берем какой-либо многочлен $\hat{h}_1(x) \in S$; пусть $\deg \hat{h}_1(x) = l_1$ (мы считаем, что $|S| \geqslant 2$). Пусть $F(x) = \hat{h}_1(x)\hat{g}_1(x) \pmod p$. Поскольку $\hat{h}_1(x)$ и $\hat{g}_1(x)$ взаимно просты в $\mathbb{Z}/p\mathbb{Z}[x]$, с помощью алгоритма 1 из § 8.4 находим представление $u(x)\hat{h}_1(x) + v(x)\hat{g}_1(x) = 1$ в $\mathbb{Z}/p\mathbb{Z}[x]$, где $\deg u(x) < \deg \hat{g}_1(x)$, $\deg v(x) < \deg \hat{h}_1(x)$. Находим минимальное натуральное число k такое, что

$$p^{kl_1} > 2^{N^2/2}(N+1)^{N/2}e^{N^2}|F(x)|^{2N}.$$

6 шаг. С помощью метода доказательства леммы 8.7 из § 8.4 (применяемого последовательно несколько раз), находим многочлен $h_1(x) \in \mathbb{Z}/p^k\mathbb{Z}[x]$, такой, что

$$h_1(x) \pmod{p^k} \mid F(x) \pmod{p^k},$$

$\deg h_1(x) = \deg \hat{h}_1(x)$, старший коэффициент $h_1(x)$ обратим по модулю p^k. Если в качестве коэффициентов $h(x) \in \mathbb{Z}/p^k\mathbb{Z}[x]$ рассматривать наименьшие неотрицательные вычеты элементов $\mathbb{Z}/p^k\mathbb{Z}$, то можно считать, что $h(x) \in \mathbb{Z}[x]$. Тогда для многочлена $h_1(x)$ выполнены свойства Б), В) и Г) из § 8.2. Домножив $h_1(x) \in \mathbb{Z}/p^k\mathbb{Z}[x]$ на некоторый элемент $a \in (\mathbb{Z}/p^k\mathbb{Z})^*$, $a \equiv 1 \pmod p$, можно обеспечить и условие А).

7 шаг. Перебирая последовательно значения

$$m = \left[\frac{N-1}{2^N}\right], \left[\frac{N-1}{2^{N-1}}\right], \ldots, \left[\frac{N-1}{2}\right], N-1,$$

будем для них выполнять шаг 8, переходя к шагу 9, как только найдем $h_0(x) \in \mathbb{Z}[x]$, $h_0(x) \mid F(x)$. Если делитель не будет найден для всех m, то $F(x)$ неприводим в $\mathbb{Z}[x]$ (это будет следовать из леммы 8.5).

8 шаг. Для данного m рассматриваем множества $L \subseteq \mathbb{Z}[x]$, $\widetilde{L} = \Phi_0(L) \subseteq \mathbb{Z}^{m+1}$, определенные в § 8.3. По базису

$$\{\Phi_0(p^k x^i) \mid 0 \leqslant i \leqslant l-1\} \cup \{\Phi_0(h_1(x)x^i) \mid 0 \leqslant i \leqslant m-l\}$$

решетки $\widetilde{L}$ находим приведенный базис $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{m+1}$. Если выполняется неравенство

$$|\boldsymbol{b}_1| < \left(\frac{p^{kl}}{|F(x)|^m}\right)^{1/N}, \tag{8.21}$$

то выполнены условия леммы 8.6. Тогда положим

$$h_0(x) = \frac{\text{НОД}(\Phi_0^{-1}(\boldsymbol{b}_1), \ldots, \Phi_0^{-1}(\boldsymbol{b}_t))}{p^\tau},$$

где t и τ — из той же леммы. Мы нашли многочлен $h_0(x) \in \mathbb{Z}[x]$, являющийся неприводимым делителем $F(x)$. Заносим $h_0(x)$ в список S_1 и идем на 9 шаг. Если же (8.21) не выполнено, то переходим к следующему значению m.

9 шаг. Полагаем $N := \deg(F(x)/h_0(x))$, $F(x) = F(x)/h_0(x)$ и из множества S удаляем делители многочлена $h_0(x) \pmod p$ (которые мы находим путем перебора по множеству S). Если $N > 1$ и $|S| > 1$, то возвращаемся на 5 шаг. Если $N = 0$ или $N = 1$ или $|S| = 1$, то многочлен $F(x)$ неприводим (или $F(x) = 1$). Тогда мы добавляем $F(x)$ в список S_1 (если $N > 0$) и переходим на 10 шаг.

10 шаг. Теперь список S_1 состоит из многочленов $f_1(s), \ldots, f_k(x)$ (см. формулу (8.20)). Пробными делениями находим показатели $\alpha_1, \ldots, \alpha_k$ и выдаем разложение $f_0(x) = f_1(x)^{\alpha_1} \ldots f_k(x)^{\alpha_k}$.

Конец алгоритма.

Корректность алгоритма легко следует из доказанных в § 8.2—8.4 утверждений.

Теорема 8.9 (см. [160]). *Алгоритм раскладывает $f_0(x)$ на неприводимые множители за $O(n^6 + n^5 \log |f|)$ арифметических операций.*

Замечание 8.10. В книге [38] приведено развернутое описание LLL-алгоритма факторизации многочленов, а также блок-схемы вспомогательных и основного алгоритмов.

§ 8.6. Практичный алгоритм факторизации

В книге [89, § 3.5.5] отмечено, что хотя описанный в § 8.5 алгоритм имеет полиномиальную сложность от длины входа, все же на практике он работает довольно медленно. В главе 3 этой книги приведен алгоритм факторизации, который, хотя и имеет экспоненциальную оценку сложности, на практике работает гораздо быстрее LLL-алгоритма. Мы вкратце опишем здесь его схему.

Алгоритм факторизации в $\mathbb{Z}[x]$.

На входе примитивный многочлен $f_0(x) \in \mathbb{Z}[x]$, который мы хотим разложить на неприводимые множители.

1 шаг. Как и в LLL-алгоритме из § 8.5, мы сводим задачу факторизации $f_0(x)$ к задаче факторизации примитивного многочлена $f(x) \in \mathbb{Z}[x]$, не имеющего кратных неприводимых делителей.

2 шаг. Находим наименьшее простое число p, для которого НОД$(f(x) \pmod p, f'(x) \pmod p) = 1$ в $\mathbb{Z}/p\mathbb{Z}[x]$. С помощью какого-либо из алгоритмов, описанных в гл. 6, находим разложение $f(x)$ на неприводимые множители в $\mathbb{Z}/p\mathbb{Z}[x]$ (это разложение будет бесквадратным).

3 шаг. Если $h(x) \in \mathbb{Z}[x]$, $h(x) \mid f(x)$, то можно оценить максимум модулей коэффициентов $h(x)$ некоторой величиной B, зависящей от $f(x)$ (см. [89, теорема 3.5.1] или формулу (8.14) из § 8.3, следующую из работы [183]). Находим B и затем находим наименьшее $e \in \mathbb{N}$, такое, что $p^e > 2l(f)B$, где $l(f)$ обозначает старший коэффициент $f(x)$ ($l(f) \in \mathbb{N}$). Положим $n_0 = \deg f(x)$. Заводим список S_1 неприводимых делителей $f(x)$, $S_1 := \varnothing$.

4 шаг. Используя результаты § 8.4, находим разложение

$$f(x) = l(f)g_1(x) \dots g_r(x) \pmod{p^e}$$

в кольце $\mathbb{Z}/p^e\mathbb{Z}[x]$.

Присваиваем $d := 1$, $S := \{g_1(x), \dots, g_r(x)\}$.

5 шаг. На этом шаге мы рассматриваем всевозможные комбинации

$$\overline{G}(x) = g_{i_1}(x) \dots g_{i_j}(x) \pmod{p^e}.$$

Мы вычисляем для каждой из них однозначно определенный многочлен $G(x) \in \mathbb{Z}[x]$ такой, что:

1) коэффициенты $G(x)$ принадлежат полуинтервалу $\left[-\frac{1}{2}p^e; \frac{1}{2}p^e\right)$;

2) $G(x) \equiv l(f)\overline{G}(x) \pmod{p^e}$, если $\deg \overline{G}(x) \leqslant \frac{1}{2}\deg f(x)$, и $G(x) \equiv$ $\equiv f(x)/\overline{G}(x) \pmod{p^e}$, если $\deg \overline{G}(x) > \frac{1}{2}\deg f(x)$.

Если $G(x)$ делит $l(f)f(x)$, то мы выносим наибольший общий делитель коэффициентов $G(x)$ и получаем неприводимый примитивный многочлен $G_1(x)$, делящий $f(x)$ в $\mathbb{Z}[x]$. Мы заносим его в список S_1 неприводимых делителей $f(x)$, удаляем соответствующие $g_{i_1}(x), \dots, g_{i_j}(x)$ из списка S и заменяем $f(x)$ на $f(x)/G_1(x)$.

6 шаг. Мы увеличиваем d на 1, т. е. $d := d + 1$. Если $d \leqslant n_0/2$, то возвращаемся на 5 шаг. Если же $d > n_0/2$, то оставшийся после выполнения 5 шага многочлен $f(x)$ будет неприводимым, и мы добавляем его в список S_1 (если он не равен константе).

Конец алгоритма.

Замечание 8.11. Дальнейшие детали см. в [89, § 3.5.4]. Аналогичные методы факторизации описаны в [1, гл. 6] и [25, § 4.6.2].

§ 8.7. Факторизация многочленов с использованием приближенных вычислений

В данном параграфе мы опишем два алгоритма из работы [160]. Один из них находит минимальный многочлен алгебраического числа, а второй раскладывает на неприводимые множители многочлены из $\mathbb{Q}[x]$ и $\mathbb{Z}[x]$. В этих алгоритмах также используются LLL-приведенные базисы решеток в сочетании с приближенным вычислением корней многочленов.

Напомним, что число $\alpha \in \mathbb{C}$ называется *алгебраическим*, если оно является корнем некоторого многочлена $h(x)$ из $\mathbb{Q}[x]$, не равного тождественно нулю. Можно считать, что $h(x) \in \mathbb{Z}[x]$, $h(x)$ неприводим и примитивен. В этом случае $h(x)$ называется *минимальным многочленом* α, его степень $\deg h(x)$ называется *степенью* $\deg \alpha$ *алгебраического числа*, корни $h(x)$ называются *сопряженными с* α *алгебраическими числами*, максимум модулей коэффициентов $h(x)$ называется *высотой алгебраического числа* α.

Для многочленов $g(x) = \sum\limits_{i=0}^{m} g_i x^i \in \mathbb{Q}[x]$ мы будем обозначать

$$|g(x)|_\infty = \max_{i=0,\ldots,m} |g_i|.$$

Норма многочлена $|g(x)|$ та же, что и в § 8.1.

В этом параграфе нам понадобятся решетки неполного ранга, т. е. множества $\Lambda = \mathbb{Z}\boldsymbol{b}_1 + \ldots + \mathbb{Z}\boldsymbol{b}_k \subseteq \mathbb{R}^n$, где $k \leqslant n$ и векторы $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k$ линейно независимы. В таких решетках приведенные базисы определяются так же, как и в случае $k = n$, и существует полиномиальный алгоритм их построения, см. [160]. При этом выполняется неравенство

$$|\boldsymbol{v}_1| \leqslant 2^{(k-1)/2} |\boldsymbol{w}| \tag{8.22}$$

для всех $\boldsymbol{w} \in \Lambda$, $\boldsymbol{w} \neq \boldsymbol{0}$ (см. [160]); здесь $\boldsymbol{v}_1$ — первый вектор из приведенного базиса решетки Λ.

Пусть α — алгебраическое число, которое известно нам лишь приближенно; d и H — известные оценки сверху для степени и высоты α. Тогда достаточно хорошее приближение $\overline{\alpha}$ к числу α позволит нам найти минимальный многочлен $h(x)$ для α.

Если $\overline{\alpha}$ фиксировано, то можно найти приближения $\overline{\alpha}_i$ к степеням α^i. Пусть они также фиксированы. Тогда для многочленов $g(x) = g = \sum\limits_i g_i x^i \in \mathbb{C}[x]$ через $g_{\overline{\alpha}}$ мы будем обозначать сумму

$$g_{\overline{\alpha}} = \sum_i g_i \overline{\alpha}_i. \tag{8.23}$$

Зафиксируем n, $1 \leqslant n \leqslant d$, и $s \in \mathbb{N}$. Рассмотрим решетку L_s в $\mathbb{R}^{n+3}$, порожденную строками матрицы

$$\begin{pmatrix} 1 & \dots & 0 & 2^s \operatorname{Re}\overline{\alpha}_0 & 2^s \operatorname{Im}\overline{\alpha}_0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 1 & 2^s \operatorname{Re}\overline{\alpha}_n & 2^s \operatorname{Im}\overline{\alpha}_n \end{pmatrix} \tag{8.24}$$

размера $(n+1)\times(n+3)$. Эти строки мы будем обозначать $\boldsymbol{b}_1, \dots, \boldsymbol{b}_{n+1} \in \in \mathbb{R}^{n+3}$. Для многочлена $g = g(x) = \sum\limits_{i=0}^{n} g_i x^i \in \mathbb{Z}[x]$ обозначим через

$$\tilde{\boldsymbol{g}} = \sum_{i=0}^{n} g_i \boldsymbol{b}_i \in L_s. \tag{8.25}$$

Легко видеть, что

$$|\tilde{\boldsymbol{g}}|^2 = g_0^2 + \dots + g_n^2 + 2^{2s}\left(\left(\operatorname{Re}\sum_{i=0}^{n} g_i\overline{\alpha}_i\right)^2 + \left(\operatorname{Im}\sum_{i=0}^{n} g_i\overline{\alpha}_i\right)^2\right) = |g(x)|^2 + 2^{2s}|g_{\overline{\alpha}}|^2.$$

Очевидно, что формула (8.25) задает взаимно однозначное соответствие между многочленами из $\mathbb{Z}[x]$ степени не выше n и векторами решетки L_s.

Идея метода нахождения минимального многочлена $h(x)$ для α заключается в следующем. Пусть $n = \deg h(x) = \deg \alpha$. Тогда для всех многочленов $g(x) \in \mathbb{Z}[x]$, $\deg g(x) \leqslant n$, таких, что $g(\alpha) \neq 0$, будет выполнено неравенство

$$|\tilde{\boldsymbol{g}}|^2 > 2^n |\tilde{\boldsymbol{h}}|^2, \tag{8.26}$$

если мы подходящим образом выберем значение s и если величины $|\alpha^i - \overline{\alpha}_i|$ достаточно малы. Найдем приведенный базис L_s. Пусть $\tilde{\boldsymbol{v}}$ — первый его вектор, и $v(x)$ — соответствующий ему (по формуле (8.25)) многочлен. Поскольку $\tilde{\boldsymbol{h}} \in L_s$, то в силу (8.22)

$$|\tilde{\boldsymbol{v}}|^2 \leqslant 2^n |\tilde{\boldsymbol{h}}|^2.$$

Тогда из (8.26) следует, что $v(\alpha) = 0$, откуда $h(x) \mid v(x)$. Поскольку $\deg v(x) \leqslant n = \deg h(x)$, то $v(x) = h(x) \cdot c$, где $c \in \mathbb{Z}$. Так как $\tilde{\boldsymbol{v}}$ — из базиса решетки L_s и $\tilde{\boldsymbol{h}} \in L_s$, то $c = \pm 1$, $v(x) = \pm h(x)$ — искомый минимальный многочлен.

Теперь приступим к аккуратному описанию и обоснованию метода.

Лемма 8.12. *Пусть* $\alpha, \overline{\alpha}_0, \dots, \overline{\alpha}_n \in \mathbb{C}$, $\overline{\alpha}_0 = 1$, $|\alpha^i - \overline{\alpha}_i| \leqslant \varepsilon$, $i = 1, \dots, n$. *Пусть* $f(x) \in \mathbb{C}[x]$, $\deg f(x) \leqslant n$. *Тогда*

$$|f(\alpha) - f_{\overline{\alpha}}| \leqslant \varepsilon n |f|_\infty.$$

Доказательство. В самом деле

$$|f(\alpha) - f_{\overline{\alpha}}| = \left|\sum_{i=1}^{n} a_i(\alpha^i - \overline{\alpha}_i)\right| \leqslant n\varepsilon|f|_\infty.$$

Лемма доказана. □

Лемма 8.13. *Пусть* $h(x)$, $g(x) \in \mathbb{Z}[x] \setminus \{0\}$, $\deg h(x) = n$, $\deg g(x) = m$. *Пусть* $\alpha \in \mathbb{C}$, $|\alpha| \leqslant 1$, $h(\alpha) = 0$. *Если* $h(x)$ *неприводим и* $g(\alpha) \neq 0$, *то выполнено неравенство*

$$|g(\alpha)| \geqslant \frac{1}{n}|h(x)|^{-m}|g(x)|^{-n+1}.$$

Доказательство. Если $m = 0$, то неравенство очевидно. Пусть $m \geqslant 1$. Рассмотрим матрицу M размера $(n+m) \times (n+m)$, у которой i-й столбец составляют коэффициенты многочлена $x^{i-1}h(x)$ при $1 \leqslant i \leqslant m$, и коэффициенты многочлена $x^{i-m-1}g(x)$ при $m+1 \leqslant i \leqslant n+m$. Пусть $R = |\det M| = |\operatorname{Res}(h(x), g(x))|$. Тогда по свойству результанта $R \neq 0$, так как $h(x)$ неприводим и $g(\alpha) \neq 0$. При $i = 2, \ldots, n+m$ прибавим i-ю строку матрицы M, умноженную на x^{i-1}, к первой строке и затем раскроем определитель M по первой строке. Получим равенство

$$R = |h(x)(a_0 + a_1x + \ldots + a_{m-1}x^{m-1}) + g(x)(c_0 + \ldots\ c_{n-1}x^{n-1})|,$$

где a_i, c_j — определители миноров M. При $x = \alpha$ получим $R = |g(\alpha)||c_o + \ldots + c_{n-1}\alpha^{n-1}|$. В силу неравенства Адамара для определителей матриц

$$|c_j| \leqslant\ |h(x)|^m|g(x)|^{n-1}.$$

Так как $|\alpha| \leqslant 1$, то $|c_0 + \ldots + c_{n-1}\alpha^{n-1}| \leqslant n|h(x)|^m|g(x)|^{n-1}$. Теперь из неравенства $R \geqslant 1$ следует утверждение леммы. □

Лемма 8.14. *Пусть* $s \in \mathbb{N}$, α — *алгебраическое число*, $|\alpha| \leqslant 1$, $h(x)$ — *минимальный многочлен* α, $\deg h(x) = d \geqslant 1$, *высота* $h(x)$ *не превосходит* H. *Пусть* $\overline{\alpha}_0, \ldots, \overline{\alpha}_d \in \mathbb{C}$, $\overline{\alpha}_0 = 1$, $|\overline{\alpha}_i - \alpha^i| \leqslant \dfrac{1}{2^s}$ *при* $1 \leqslant i \leqslant d$. *Пусть* $g(x) \in \mathbb{Z}[x]$, $\deg g(x) \leqslant d$, $g(\alpha) \neq 0$. *Предположим, что выполнено неравенство*

$$2^s \geqslant 2^{d^2/2}(d+1)^{(3d+4)/2}H^{2d}. \tag{8.27}$$

Тогда (*в обозначениях* (8.25))

$$|\tilde{h}| < (d+1)H, \quad |\tilde{g}| > 2^{d/2}(d+1)H.$$

Доказательство. Очевидно, что для любого многочлена $f(x) \in \mathbb{C}[x]$, $\deg f(x) \leqslant d$, выполнено неравенство $|f(x)|^2 \leqslant (d+1)|f(x)|_\infty^2$. Поскольку

$|\tilde{h}|^2 = |h(x)|^2 + 2^{2s}|h_{\overline{\alpha}}|^2$, и по лемме 8.12 $|h_{\overline{\alpha}}| = |h(\alpha) - h_{\overline{\alpha}}| \leqslant 2^{-s}dH$, то

$$|\tilde{h}|^2 \leqslant |h(x)|^2 + d^2H^2 \leqslant (d+1)H^2 + d^2H^2 < (d+1)^2H^2.$$

Теперь докажем неравенство для $|\tilde{g}|$. Если $|g(x)| > 2^{d/2}(d+1)H$, то из равенства $|\tilde{g}|^2 = |g(x)|^2 + 2^{2s}|g_{\overline{\alpha}}|^2$ следует требуемое неравенство. Допустим, что $|g(x)| \leqslant 2^{d/2}(d+1)H$. По лемме 8.13 тогда

$$|g(\alpha)| \geqslant \frac{1}{d} \cdot \frac{1}{|h(x)|^d} \cdot \frac{1}{|g(x)|^{d-1}} \geqslant \frac{1}{d} \cdot \frac{1}{((d+1)H^2)^{d/2}} \cdot \frac{1}{(2^{d/2}(d+1)H)^{d-1}} >$$

$$> 2^{-d(d-1)/2} \cdot (d+1)^{-\frac{3}{2}d} H^{-2d+1}.$$

Поэтому

$$|\tilde{g}| \geqslant 2^s|g_{\overline{\alpha}}| \geqslant 2^s(|g(\alpha)| - |g(\alpha) - g_{\overline{\alpha}}|) >$$

$$> 2^s(2^{-d(d-1)/2}(d+1)^{-\frac{3}{2}d} H^{-2d+1} - 2^{-s}d|g(x)|_\infty) =$$

$$= 2^{s-\frac{d(d-1)}{2}}(d+1)^{-\frac{3}{2}d} H^{-2d+1} - d|g(x)|_\infty.$$

Так как $|g(x)|_\infty \leqslant |g(x)| \leqslant 2^{d/2}(d+1)H$, то

$$|\tilde{g}| > H \cdot 2^{d/2}(2^{s-\frac{d^2}{2}}(d+1)^{-\frac{3}{2}d} H^{-2d} - d(d+1)),$$

откуда с учетом (8.27) получим требуемое неравенство $|\tilde{g}| > H \cdot 2^{d/2}(d+1)$. □

Теорема 8.15. *Пусть* s, α, $h(x)$, d, h *и числа* $\overline{a}_i \in 2^{-s}\mathbb{Z}[\sqrt{-1}]$, $i = 0, \ldots, d$, *удовлетворяют условиям леммы* 8.14, *включая неравенство* (8.27). *Пусть* $n \in \mathbb{N}$, $1 \leqslant n \leqslant L$, *и пусть LLL-алгоритм построения приведенного базиса, примененный к векторам* $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{n+1}$, *являющимися строками матрицы* (8.24), *дает приведенный базис с первым вектором* $\tilde{\boldsymbol{v}} = \sum_{i=0}^{n} v_i \boldsymbol{b}_i$. *Тогда три следующих условия эквивалентны*:

1) $|\tilde{\boldsymbol{v}}| \leqslant 2^{d/2}(d+1)H$;
2) α — *корень многочлена* $v(x) = \sum_{i=0}^{n} v_i x^i$;
3) $\deg \alpha \leqslant n$.

Далее, если $\deg \alpha = n$, *то* $h(x) = \pm v(x)$.

Доказательство. По формуле (8.22) выполняется неравенство $|\tilde{\boldsymbol{v}}| \leqslant 2^{n/2}|\tilde{\boldsymbol{w}}|$ для всех $\tilde{\boldsymbol{w}} \in L_s = \mathbb{Z}\boldsymbol{b}_1 + \ldots + \mathbb{Z}\boldsymbol{b}_{n+1}$, $\tilde{\boldsymbol{w}} \neq \boldsymbol{0}$. Пусть выполнено первое условие теоремы. Тогда из леммы 8.14 следует, что $v(\alpha) = 0$.

Очевидно, что из второго условия следует третье. Пусть выполнено третье условие. Тогда $\deg h(x) \leqslant n$, поэтому $\tilde{\boldsymbol{h}}$ — вектор из L_s. По лемме 8.14 имеем $|\tilde{\boldsymbol{h}}| < (d+1)H$. Тогда из (8.22) следует, что $|\tilde{\boldsymbol{v}}| \leqslant 2^{n/2}(d+1)H$, т. е. из третьего условия следует первое.

Наконец, пусть $\deg \alpha = n$. Тогда в силу доказанного $v(\alpha) = 0$. Так как $h(x)$ неприводим, то из леммы Гаусса (см. § 8.1) следует, что $h(x)$ делит $v(x)$ в $\mathbb{Z}[x]$; $v(x) = h(x) \cdot l$, где $l \in \mathbb{Z}$. Отсюда $\tilde{\boldsymbol{v}} = \tilde{\boldsymbol{h}} \cdot l$. Так как $\tilde{\boldsymbol{v}}$ — из базиса решетки L_s и $\tilde{\boldsymbol{h}} \in L_s$, то $l = \pm 1$. □

Алгоритм 1 (нахождение минимального многочлена).

На входе алгоритма заданы верхние оценки d и H для степени и высоты алгебраического числа α, $|\alpha| \leqslant 1$; приближение $\overline{\alpha} \in \mathbb{Q} + \sqrt{-1}\mathbb{Q}$, $|\overline{\alpha}| \leqslant 1$, такое, что $|\alpha - \overline{\alpha}| \leqslant 2^{-s}/(4d)$, где s — минимальное натуральное число, удовлетворяющее неравенству

$$2^s \geqslant 2^{d^2/2}(d+1)^{(3d+4)/2}H^{2d}.$$

На выходе — минимальный многочлен для α.

1 шаг. Вычисляем $\overline{\alpha}_i \in 2^{-s}(\mathbb{Q} + \sqrt{-1}\mathbb{Q})$, $i = 0, \ldots, d$, такие, что $\overline{\alpha}_0 = 1$, $|\overline{\alpha}^i - \overline{\alpha}_i| \leqslant 2^{-s-\frac{1}{2}}$, $1 \leqslant i \leqslant d$. Для этого округляем вещественную и мнимую части числа $\overline{\alpha}^i$ до s битов после запятой. Заметим, что тогда $|\alpha^i - \overline{\alpha}_i| \leqslant 2^{-s}$, поскольку

$$|\alpha^i - \overline{\alpha}_i| \leqslant |\alpha^i - \overline{\alpha}^i| + |\overline{\alpha}^i - \overline{\alpha}_i| \leqslant$$
$$\leqslant |\alpha - \overline{\alpha}| \cdot \sum_{j=0}^{i-1} |\alpha|^j |\overline{\alpha}|^{i-1-j} + 2^{-s-\frac{1}{2}} \leqslant \frac{2^{-s}}{4d} \cdot d + 2^{-s-\frac{1}{2}} \leqslant \frac{1}{2^s}.$$

Это означает, что выполнено условие теоремы 8.15.

2 шаг (цикл). Для $n = 1, 2, \ldots, d$ выполняем шаги 3 и 4, пока не найдем минимальный многочлен для α.

3 шаг. Применяя LLL-алгоритм, находим приведенный базис решетки $L_s = \mathbb{Z}\boldsymbol{b}_0 + \ldots + \mathbb{Z}\boldsymbol{b}_n$, порожденной строками матрицы (8.24).

4 шаг. Если первый вектор $\tilde{\boldsymbol{v}}$ приведенного базиса удовлетворяет неравенству $|\tilde{\boldsymbol{v}}| \leqslant 2^{d/2}(d+1)H$, то выдаем многочлен $v(x) = \sum\limits_{i=0}^{n} v_i x^i$ $\left(\text{где } \tilde{\boldsymbol{v}} = \sum\limits_{i=0}^{n} v_i \boldsymbol{b}_i\right)$ в качестве минимального многочлена для α.

Конец алгоритма.

Корректность алгоритма следует из теоремы 8.15.

Замечание 8.16. Неравенство $|\alpha| \leqslant 1$ не является серьезным ограничением. Если $|\alpha| > 1$ и α — корень многочлена $h(x)$ то $1/\alpha$ — корень

многочлена $x^{\deg h(x)} \cdot h\left(\frac{1}{x}\right)$, $\left|\frac{1}{\alpha}\right| < 1$. При этом, если $\overline{\alpha}$ — приближение к α, $|\alpha - \overline{\alpha}| \leqslant \varepsilon$, где $0 < \varepsilon \leqslant \frac{1}{2}$, и β — приближение к $1/\overline{\alpha}$, $|\beta - 1/\overline{\alpha}| \leqslant \varepsilon$, то

$$\left|\frac{1}{\alpha} - \overline{\beta}\right| \leqslant \left|\frac{1}{\alpha} - \frac{1}{\overline{\alpha}}\right| + \left|\overline{\beta} - \frac{1}{\overline{\alpha}}\right| \leqslant \frac{\varepsilon}{|\overline{\alpha}|} + \varepsilon.$$

Поскольку $|\overline{\alpha}| \geqslant |\alpha| - |\alpha - \overline{\alpha}| \geqslant |\alpha| - |\alpha|\varepsilon \geqslant \frac{1}{2}$, то $|\frac{1}{\alpha} - \overline{\beta}| \leqslant 3\varepsilon$, т. е. $\overline{\beta}$ приближает $1/\alpha$ с точностью 3ε.

Теорема 8.17 (см. [143]). *Пусть α — алгебраическое число, d и H — оценки на степень и высоту α, $\overline{\alpha}$ — приближение к α такое, что $|\alpha - \overline{\alpha}| \leqslant \frac{2^{-s}}{12d}$, где s — минимальное натуральное число, для которого выполнено неравенство* (8.27). *Тогда алгоритм 1 находит минимальный многочлен для α за $O(d^5(d + \log H))$ арифметических операций с целыми числами, имеющими величину $O(d^2(d + \log H))$ битов.*

Теперь перейдем к факторизации многочленов. Пусть $f(x) \in \mathbb{Z}[x]$ — фиксированный примитивный многочлен, $\deg f(x) = d \geqslant 2$, и мы хотим разложить $f(x)$ на неприводимые множители в $\mathbb{Z}[x]$. Если $h(x) \in \mathbb{Z}[x]$, $h(x)$ — делитель $f(x)$, $\deg h(x) = n$, то высота $h(x)$ не превосходит $\binom{n-1}{j}|f(x)| + \binom{n-1}{j-1}|a_d|$ при всех $j = 1, \ldots, n-1$, где a_d — старший коэффициент $f(x)$ (см. [25, упр. 4.6.2.20; 89, теорема 3.5.1]). Отсюда, считая, что $n \leqslant d/2$ (если $f(x)$ приводим), мы можем выбрать оценку H сверху для коэффициентов искомого $h(x)$, которая понадобится для применения алгоритма 1.

Алгоритм 2 (факторизация $f(x)$ в $\mathbb{Z}[x]$).

На входе алгоритма заданы $f(x)$, d, H. Алгоритм последовательно выдает неприводимые делители $f(x)$, пока полностью не разложит $f(x)$ на множители в $\mathbb{Z}[x]$.

1 шаг. Если $d \leqslant 1$, то $f(x)$ неприводим, и алгоритм заканчивает работу.

2 шаг. Найдем наименьшее $s \in \mathbb{N}$ такое, что

$$2^s \geqslant 2^{d^2/2}(d+1)^{(3d+4)/2} H^{2d}.$$

3 шаг. Вычислим (каким-либо способом) такое приближение $\overline{\alpha} \in \mathbb{Q} + i\mathbb{Q}$ к некоторому корню α многочлена $f(x)$, что $|\alpha - \overline{\alpha}| \leqslant \frac{2^{-s}}{12d}$ $\left(\text{если } |\alpha| \leqslant 1, \text{ то достаточно выполнения неравенства } |\alpha - \overline{\alpha}| \leqslant \frac{2^{-s}}{4d}\right)$.

4 шаг. Применяя алгоритм 1, находим минимальный многочлен $h(x)$ для алгебраического числа α.

5 шаг. Пробными делениями находим наибольшее $k \in \mathbb{N}$ такое, что $h(x)^k \mid f(x)$, и выдаем $h(x)$ и k. Этот шаг можно не делать, если многочлен $f(x)$ бесквадратен; бесквадратность можно обеспечить так же, как в алгоритме из § 8.5.

6 шаг. Заменяем d на $d - k \deg h(x)$, $f(x)$ на $f(x)/h(x)^k$ и возвращаемся на 1 шаг.

Конец алгоритма.

Корректность алгоритма 2 очевидна. Оценим его сложность. Нам осталось выбрать метод приближенного вычисления корней многочлена из $\mathbb{Z}[x]$ на 3 шаге алгоритма 2. Существуют различные методы для решения этой задачи, см. [6; 89, гл. 3]. В частности, существуют алгоритмы, имеющие полиномиальную сложность от d, $\log|f(x)|$ и количества требуемых битов, см. [143; 242]. Использование этих алгоритмов позволяет указать полиномиальную оценку сложности и для алгоритма 2 (см. [143, теорема 3.7]).

§ 8.8. Заключение

Появление в начале 80-х годов XX века LLL-алгоритма для факторизации многочленов из $\mathbb{Z}[x]$ с полиномиальной сложностью от длины входа вызвало огромный интерес у математиков во всем мире. До сих пор сложность имеющихся алгоритмов для факторизации целых чисел гораздо хуже, как мы видели в гл. 2—4. LLL-приведенные базисы решеток нашли применение во многих задачах математики; часть из них была описана в гл. 7. Дальнейшее развитие метода работы [160] было получено в ряде работ отечественных авторов, см., например, [21; 43].

Кроме описанных в данной главе алгоритмов факторизации многочленов из $\mathbb{Z}[x]$ и $\mathbb{Q}[x]$ имеются алгоритмы факторизации многочленов с коэффициентами из числовых полей, см., например, [89, гл. 3].

В последние годы появились работы, связанные с применением LLL-приведенных базисов в алгоритмах решета числового поля для факторизации и дискретного логарифмирования, см., например, [204]. В основном это применение относится к оптимальному выбору многочленов, порождающих числовые поля.

Глава 9. Дискретное преобразование Фурье и его приложения

§ 9.1. Введение. Дискретное преобразование Фурье и его свойства

Дискретное преобразование Фурье имеет важные приложения в теории чисел и алгебре. Оно подробно описано во многих книгах, см., например, [5; 37]. Мы не претендуем здесь на полноту изложения. Будут описаны лишь некоторые основные свойства и некоторые важные приложения дискретного преобразования Фурье. В частности, будет доказана теорема, необходимая для обоснования алгоритма Полларда—Штрассена из главы 2; ее доказательство взято из работы [271].

Введем некоторые обозначения. Пусть $t \in \mathbb{N}$, $n = 2^t$. Пусть R — коммутативное кольцо с единицей 1, содержащее 2^{-1} — элемент, обратный к 2. Пусть также R содержит элемент ζ_{2n} — некоторый фиксированный корень уравнения уравнения $x^n + 1 = 0$. Положим $\zeta_n = \zeta_{2n}^2$.

Лемма 9.1. *Элемент ζ_{2n} порождает в мультипликативной группе обратимых элементов кольца R циклическую подгруппу порядка $2n$.*

Доказательство. Так как $\zeta_{2n}^{2^t} = -1$, то мультипликативный порядок элемента ζ_{2n} равен $2^{t+1} = 2n$. □

Определение 9.2. Пусть $(f_0, \ldots, f_{n-1}) \in R^n$ — произвольный вектор.

Дискретным преобразованием Фурье 1-го типа для этого вектора называют n-мерный вектор $(\hat{f}_0, \ldots, \hat{f}_{n-1}) \in R^n$, определяемый равенствами

$$\hat{f}_i = \sum_{j=0}^{n-1} \zeta_n^{ij} f_j, \quad i = 0, \ldots, n-1.$$

Дискретным преобразованием Фурье 2-го типа называют n-мерный вектор $(\check{f}_1, \check{f}_3, \ldots, \check{f}_{2n-1}) \in R^n$, определяемый равенствами

$$\check{f}_i = \sum_{j=0}^{n-1} \zeta_{2n}^{ij} f_j, \quad i = 1, 3, \ldots, 2n-1.$$

Замечание 9.3. Элементы $\hat{f}_i$ являются значениями многочлена $F(x) = \sum\limits_{j=0}^{n-1} f_j x^j \in R[x]$ в точках $x = \zeta_n^i$; элементы $\check{f}_i$ являются значениями многочлена $F(x)$ в точках $x = \zeta_{2n}^i$ при нечетных i.

В следующей лемме содержатся формулы обращения для преобразования Фурье.

Лемма 9.4. *Справедливы следующие равенства*:

$$f_i = n^{-1} \sum_{j=0}^{n-1} \hat{f}_j \zeta_n^{-ij}, \tag{9.1}$$

$$f_i = n^{-1} \sum_{\substack{1 \leqslant j \leqslant 2n-1 \\ j \text{ — нечетно}}} \check{f}_j \zeta_{2n}^{-ij}, \tag{9.2}$$

где $n^{-1} = (2^{-1})^t \in R$.

Доказательство. Пусть $k \in \mathbb{Z}$. Тогда $\zeta_{2n}^{-k} = \zeta_{2n}^{2nr-k}$, где $r \in \mathbb{Z}$, $2nr - k \geqslant 0$. Кроме того, поскольку мультипликативный порядок элемента ζ_n равен n, то справедливы равенства

$$\sum_{j=0}^{n-1} \zeta_n^{lj} = n \quad \text{при } l \equiv 0 \pmod{n}, \tag{9.3}$$

$$\sum_{j=0}^{n-1} \zeta_n^{lj} = 0 \quad \text{при } l \not\equiv 0 \pmod{n}. \tag{9.4}$$

Равенство (9.1) выполняется, поскольку

$$\sum_{j=0}^{n-1} \hat{f}_j \zeta_n^{-ij} = \sum_{j=0}^{n-1} \sum_{k=0}^{n-1} f_k \zeta_n^{jk} \zeta_n^{-ij} = \sum_{k=0}^{n-1} f_k \sum_{j=0}^{n-1} \zeta_n^{(k-i)j} = n f_i$$

в силу (9.3) и (9.4).

Справедливость (9.2) следует из

$$\sum_{\substack{1 \leqslant j \leqslant 2n-1 \\ j \text{ — нечетно}}} \check{f}_j \zeta_{2n}^{-ij} = \sum_{k=0}^{n-1} \check{f}_{2k+1} \zeta_{2n}^{-i(2k+1)} =$$

$$= \sum_{k=0}^{n-1} \sum_{l=0}^{n-1} f_l \zeta_{2n}^{l(2k+1)} \zeta_{2n}^{-i(2k+1)} = \sum_{l=0}^{n-1} f_l \zeta_{2n}^{l-i} \sum_{k=0}^{n-1} \zeta_n^{(l-i)k}.$$

С учетом (9.3) и (9.4) получаем равенство (9.2). □

Заметим, что вычисление дискретного преобразования Фурье непосредственно по формулам (9.1) и (9.2) требует $O(n^2)$ операций сложения и умножения в кольце R. В следующем параграфе мы приведем метод, который позволяет вычислить дискретное преобразование Фурье за $O(n \log n)$ операций. Этот метод называется *быстрым преобразованием Фурье*.

§ 9.2. Вычисление дискретного преобразования Фурье

В этом параграфе мы сохраняем обозначения и предположения из § 9.1.

Теорема 9.5. *Дискретное преобразование Фурье* $(\hat{f}_0, \ldots, \hat{f}_{n-1})$ *может быть вычислено за* nt *сложений в кольце* R *и* nt *умножений в кольце* R *на степени элемента* ζ_n.

Дискретное преобразование Фурье $(\check{f}_1, \ldots, \check{f}_{2n-1})$ *может быть вычислено за* nt *сложений в кольце* R *и* nt *умножений в кольце* R *на степени элемента* ζ_{2n}.

Если задан вектор $(\hat{f}_0, \ldots, \hat{f}_{n-1})$ *или* $(\check{f}_1, \ldots, \check{f}_{2n-1})$, *то вектор* $(f_0, \ldots, f_{n-1})$ *может быть вычислен за* nt *сложений в кольце* R, nt *умножений в кольце* R *на степени элемента* ζ_{2n} *и* n *умножений на* $n^{-1} \in R$.

Доказательство. Обозначим

$$F(x) = \sum_{j=0}^{n-1} f_j x^j = \sum_{\substack{0 \leqslant j \leqslant n-1 \\ j \text{ — четно}}} f_j x^j + \sum_{\substack{0 \leqslant j \leqslant n-1 \\ j \text{ — нечетно}}} f_j x^j = F_0(x^2) + xF_1(x^2),$$

где $\deg F_0(x)$, $\deg F_1(x) < \dfrac{n}{2} = 2^{t-1}$. Отсюда

$$F(\zeta_n^i) = F_0(\zeta_n^{2i}) + \zeta_n^i F_1(\zeta_n^{2i}), \qquad i = 0, \ldots, n-1. \tag{9.5}$$

Положим $\zeta_{n/2} = \zeta_n^2$. Тогда

$$\{\zeta_n^{2i} \mid 0 \leqslant i \leqslant n-1\} = \left\{\zeta_{n/2}^i \,\middle|\, 0 \leqslant i \leqslant \frac{n}{2} - 1\right\}.$$

Далее рассуждаем по индукции. Вычисление дискретного преобразования Фурье первого типа, т. е. вычисление набора $(F(\zeta_n^0), \ldots, F(\zeta_n^{n-1}))$ по формуле (9.5) можно выполнить за n сложений в кольце R и n умножений в R на степени ζ_n, если известны наборы значений $F_0(\zeta_{n/2}^i)$ и $F_1(\zeta_{n/2}^i)$, $i = 0, \ldots, \dfrac{n}{2} - 1$. Если $t = 1$ и $n = 2 = 2^t$ (это основание

индукции), то для вычисления $(\hat{f}_0, \hat{f}_1)$ надо найти $F_0 + \zeta_2^i F_1$, где F_0 и F_1 — элементы кольца R, $i = 0, 1$. То есть при $n = 2$ требуется $2 = nt$ умножений в R на степени $\zeta_n = \zeta_2$ и $2 = nt$ сложений в R.

Теперь предположим, что при всех $j < t$ для вычисления дискретного преобразования Фурье первого типа от 2^j-мерного вектора требуется $2^j \cdot j$ умножений в R на степени $\zeta_{2^j} = (\zeta_n)^{2^{t-j}}$ и $2^j \cdot j$ сложений в R. Тогда при $j = t$ вычисление вектора $(\hat{f}_0, \ldots, \hat{f}_{n-1})$ по формуле (9.5) состоит в вычислении $F_0(\zeta_{n/2}^i)$ и $F_1(\zeta_{n/2}^i)$ при $i = 0, \ldots, n-1$, т. е. в вычислении преобразований Фурье для векторов длины $n/2$, состоящих из коэффициентов $F_0(x)$ и $F_1(x)$, плюс еще n сложений в R и n умножений в R на степени ζ_n. Тогда, учитывая индуктивное предположение, вычисление вектора $(\hat{f}_0, \ldots, \hat{f}_{n-1})$ потребует не более чем n сложений в R, плюс n умножений в R, плюс $2 \cdot 2^{t-1}(t-1)$ сложений в R, плюс $2 \cdot 2^{t-1}(t-1)$ умножений в R на степени ζ_n. То есть требуется $n + n(t-1) = nt$ сложений в R и $n + n(t-1) = nt$ умножений в R на степени ζ_n. Это доказывает первое утверждение теоремы.

Второе утверждение теоремы доказывается аналогично, достаточно лишь заменить ζ_n на ζ_{2n}.

Третье утверждение теоремы следует из первых двух и леммы 9.4. □

§ 9.3. Дискретное преобразование Фурье и умножение многочленов

В этом параграфе мы покажем, как с помощью дискретного преобразования Фурье можно быстро умножать многочлены. Мы сохраняем обозначения и предположения из § 9.1.

Лемма 9.6. *Одно умножение в кольце $R[x]/(x^{2^t} + 1)$ можно выполнить за $n = 2^t$ умножений в R, $3tn$ сложений в R, $3tn$ умножений в R на степени элемента ζ_{2n} и n умножений в R на элемент n^{-1}.*

Замечание 9.7. Лемма неприменима в случае $R = \mathbb{Z}$ и $R = \mathbb{Q}$, поскольку в $\mathbb{Z}$ и $\mathbb{Q}$ нет элемента ζ_{2n}.

Доказательство. Пусть $F = \sum_{i=0}^{n-1} f_i x^i$, $G = \sum_{i=0}^{n-1} g_i x^i$, $F, G \in R[x]$ и являются представителями каких-либо классов факторкольца $R[x]/(x^{2^t} + 1)$. Пусть $H = \sum_{i=0}^{n-1} h_i x^i \in R[x]$, $FG \equiv H \pmod{x^n + 1}$, т. е. H есть результат умножения F на G в факторкольце; его нужно вычислить.

Преобразование Фурье 2-го типа для векторов $(f_0, \ldots, f_{n-1})$ и $(g_0, \ldots, g_{n-1})$ дает нам равенства

$$\check{f}_i \check{g}_i = F(\zeta_{2n}^i) G(\zeta_{2n}^i) = H(\zeta_{2n}^i) = \check{h}_i$$

при нечетных i, $1 \leqslant i \leqslant 2n-1$, так как $\zeta_{2n}^n + 1 = 0$. Если все $\check{f}_i$ и $\check{g}_i$ нам известны, то все $\check{h}_i$ вычисляются за n умножений в R. По теореме 9.5 все элементы $\check{f}_i$ и $\check{g}_i$ можно вычислить за $2tn$ сложений в R и $2tn$ умножений в R на степени элемента ζ_{2n}. По той же теореме все h_i можно вычислить, зная $\check{h}_i$, за tn сложений в R, tn умножений в R на степени элемента ζ_{2n} и n умножений в R на элемент n^{-1}. Отсюда легко следует утверждение леммы. □

Следствие 9.8. *Пусть T — коммутативное кольцо с единицей, $2^{-1} \in T$, $\zeta_{4n} = \zeta_{2^{t+2}} \in T$ — корень уравнения $x^{2n} + 1 = 0$. Если $F(x)$, $G(x) \in T[x]$, $\deg F(x) < n$, $\deg G(x) < n$, то произведение многочленов $F(x) \cdot G(x) \in T[x]$ можно вычислить за $2n$ умножений в T, $6n(t+1)$ сложений в T, $6n(t+1)$ умножений в T на степени элемента ζ_{4n} и $2n$ умножений в T на элемент $(2^{-1})^{t+1}$.*

Доказательство. Из-за ограничений на степени многочленов $F(x)$ и $G(x)$ многочлен $F(x) \cdot G(x)$ имеет степень меньше, чем $2n$, и поэтому равен остатку от деления $F(x) \cdot G(x)$ на $2^{2n} + 1$. Поэтому результат умножения $F(x)$ на $G(x)$ в кольце $R[x]$ можно найти за одно умножение в кольце $R[x]/(x^{2n} + 1)$. Утверждение следствия вытекает из леммы 9.1, где n заменено на $2n$ и t заменено на $t+1$. □

Пусть далее T обозначает коммутативное кольцо с единицей, содержащее элемент 2^{-1}. Говоря об операции сложения в T, мы будем подразумевать также и операцию вычитания. Все постоянные в символах $O(\cdot)$ далее абсолютные. Справедлива следующая фундаментальная лемма.

Лемма 9.9. *Если $t \geqslant 2$, то одно умножение в кольце $T[x]/(x^{2^t} + 1)$ можно выполнить за $O(2^t \cdot t)$ умножений в T и $O(2^t \cdot t \cdot \log t)$ сложений в T.*

Замечание 9.10. Этот результат применим и к кольцу целых чисел $\mathbb{Z}$, если рассматривать в качестве T следующее кольцо:

$$T = \left\{ \frac{m}{2^k} \;\middle|\; m \in \mathbb{Z},\ k \in \mathbb{Z}_{\geqslant 0} \right\}, \qquad T \supseteq \mathbb{Z}.$$

В компьютере элементы кольца T представимы точно, например, в виде пар (m, k).

Прежде чем привести довольно длинное доказательство леммы 9.9, выведем из нее следующую важную теорему.

Теорема 9.11. *Произведение двух многочленов из кольца $T[x]$ степеней, не превосходящих n ($n \geqslant 3$), может быть вычислено за $M(n) = O(n \log n)$ умножений в T и $A(n) = O(n \log n \log\log n)$ сложений в T.*

Доказательство. Пусть $t \in \mathbb{N}$ такое, что $2^{t-1} \leqslant 2n < 2^t$. Очевидно, что $t \geqslant 2$, $2n < 2^t \leqslant 4n$. Поэтому умножение двух многочленов степени не выше n по модулю $x^{2^t} + 1$ есть обычное их умножение. Тогда по лемме 9.9 мы можем найти их произведение за $M(n) = O(2^t \cdot t) = O(n \cdot \log n)$ умножений в T и $A(n) = O(2^t \cdot t \log t) = O(n \cdot \log n \log\log n)$ сложений в T. $\square$

Доказательство леммы 9.9. Пусть $F = F(x)$ и $G = G(x)$ — многочлены степеней не выше, чем $2^t - 1$, $H = H(x) = \sum\limits_{i=0}^{2^t-1} H_i x^i$, причем $H \equiv FG \pmod{x^{2^t} + 1}$. Мы должны вычислить $H_0, \ldots, H_{2^t-1}$ по коэффициентам многочленов F и G. Пусть k — натуральный параметр, $1 \leqslant k < t$, который мы выберем ниже. Представим F и G в виде

$$F = \sum_{i=0}^{2^{t-k}-1} f_i(x) x^{i \cdot 2^k}, \qquad G = \sum_{i=0}^{2^{t-k}-1} g_i(x) x^{i \cdot 2^k},$$

где $f_i(x)$, $g_i(x) \in T[x]$, $\deg f_i(x) \leqslant 2^k - 1$, $\deg g_i(x) \leqslant 2^k - 1$.

Алгоритм вычисления $F \cdot G$ заключается в следующем.

1 шаг. Умножаем $\sum\limits_{i=0}^{2^{t-k}-1} f_i(x) Y^i$ на $\sum\limits_{i=0}^{2^{t-k}-1} g_i(x) Y^i$ в кольце $T[x, Y]/(Y^{2^{t-k}} - 1)$. Результат обозначим через $\widetilde{H} = \sum\limits_{i=0}^{2^{t-k}-1} h_i(x) Y^i$.

2 шаг. Подставим $Y = x^{2^k}$ в $\widetilde{H}$ и приведем по модулю $x^{2^t} + 1 = Y^{2^{t-k}} + 1$. Тогда

$$F(x) \cdot G(x) = \sum_{l=0}^{2^{t-k}-1} f_l(x) x^{2^k l} \cdot \sum_{j=0}^{2^{t-k}-1} g_j(x) x^{2^k j} \equiv$$
$$\equiv \sum_{i=0}^{2^{t-k}-1} h_i(x) x^{2^k i} \pmod{(x^{2^k})^{2^{t-k}} + 1}.$$

Отсюда на шаге 2 мы найдем $H(x) = \sum\limits_{i=0}^{2^t-1} H_i x^i$. Надо только понять, как на шаге 2 из набора $\{h_i(x)\}$ при приведении по модулю $x^{2^t} + 1$ получаются коэффициенты H_i.

На 1 шаге мы перемножаем

$$\sum_{l=0}^{2^{t-k}-1} f_l(x)Y^l \cdot \sum_{j=0}^{2^{t-k}-1} g_j(x)Y^j \equiv \sum_{i=0}^{2^{t-k}-1} h_i(x)Y^i \pmod{Y^{2^{t-k}}+1}.$$

Здесь $l+j \leqslant 2^{t-k+1}-2 < 2^{t-k+1}-1$. При этом возможны два случая.

1 случай. Если $0 \leqslant l+j \leqslant 2^{t-k}-1$, то

$$Y^l \cdot Y^j \equiv Y^{l+j} \pmod{Y^{2^{t-k}}+1}.$$

2 случай. Если $2^{t-k} \leqslant l+j \leqslant 2 \cdot 2^{t-k}-2$, то

$$Y^{l+j} \equiv -Y^i \pmod{Y^{2^{t-k}}+1},$$

где $i = l+j-2^{t-k}$.

Поэтому

$$h_i(x) = \sum_{\substack{l,j \\ l+j=i}} f_l(x)g_i(x) - \sum_{\substack{l,j \\ l+j=i+2^{t-k}}} f_l(x)g_i(x),$$

где $i = 0, \ldots, 2^{t-k}-1$. Отсюда следует, что $\deg h_i(x) \leqslant \deg f_l(x) = \deg g_j(x) \leqslant 2^{k+1}-2 < 2^{k+1}-1$. Поскольку $k < t$, то $\deg h_i < 2^t - 1$.

2 шаг выполняется не более чем за 2^t сложений в T, поскольку если мы уже знаем $h_i(x) = \sum\limits_{j=0}^{2^{k+1}-1} h_{ij}x^j$, то при подстановке $Y = x^{2^k}$ мы получим выражение

$$\sum_{i=0}^{2^{t-k}-1} \sum_{j=0}^{2^{k+1}-1} h_{ij}x^{j+i2^k} \pmod{x^{2^t}+1}.$$

Приведение по модулю $x^{2^t}+1$ будет происходить с теми слагаемыми, у которых $j+i2^k \geqslant 2^t$. Тогда при $j+i2^k = r2^t + l$, где $0 \leqslant l < 2^t$, будет происходить замена x^{j+i2^k} на $(-1)^r \cdot x^l$, т. е. величину $(-1)^r \cdot h_{ij}$ надо будет прибавить к коэффициенту при x^l (т. е. надо выполнить одно сложение или вычитание). Таких сложений будет не больше, чем число слагаемых, т. е. не больше, чем $2^{t-k} \cdot 2^{k+1} = 2^{t+1}$. Но на самом деле при $0 \leqslant i \leqslant 2^{t-k-1}$ выполнено неравенство

$$j + 2^k \cdot i \leqslant 2^{k+1} - 1 + 2^{t-1} - 2^k = 2^k + 2^{t-1} - 1 \leqslant 2^t - 1,$$

поскольку $k \leqslant t-1$. То есть для таких i приведение выполнять не нужно. Остаются значения i в интервале $2^{t-k-1} \leqslant i \leqslant 2^{t-k}-1$; таких значений будет не более чем 2^{t-k-1}. Следовательно, пар (i, j) для которых

на втором шаге придется выполнять приведение, будет не больше, чем $2^{t-k-1} \cdot 2^{k+1} = 2^t$. Мы показали, что 2 шаг выполняется не более чем за 2^t сложений в T.

Теперь рассмотрим 1 шаг. Умножение мы можем выполнять не в кольце $T[x, Y]/(Y^{2^{t-k}} - 1)$, а в кольце $R[Y]/(Y^{2^{t-k}} + 1)$, где $R = T[x]/(x^{2^{k+1}} + 1)$. В самом деле, поскольку $\deg f_i(x) < 2^k$, $\deg g_i(x) < 2^k$, то их произведение в кольце $T[x]$ совпадает с их произведением в кольце $T[x]/(x^{2^{k+1}} + 1)$. В кольце R есть элемент $\zeta_{2^{k+2}} \equiv x \pmod{x^{2^{k+1}} + 1}$ — корень уравнения $X^{2^{k+1}} + 1 = 0$.

Положим $k = [t/2] \geqslant 1$. Тогда

$$k \geqslant \frac{t-1}{2}, \quad k \leqslant \frac{t}{2} < t. \tag{9.6}$$

Поскольку $2^{t-k+1} \leqslant 2^{k+2}$, в R есть элемент $\zeta_{2^{t-k+1}}$ — это степень элемента $\zeta_{2^{k+2}}$.

Одно умножение на степень элемента $\zeta_{2^{t-k+1}}$ в кольце R выполняется не более чем за 2^{k+1} сложений в T. В самом деле, поскольку $x \equiv \zeta_{2^{k+2}} \pmod{x^{2^{k+1}} + 1}$, то

$$R = \{a_0 + a_1\zeta_{2^{k+2}} + \ldots + a_{2^{k+1}-1}\zeta_{2^{k+2}}^{2^{k+1}-1} \mid a_i \in T,\ i = 0, \ldots, 2^{k+1} - 1\}.$$

Умножение на $\zeta_{2^{t+k-1}}^j = \zeta_{2^{k+2}}^{j_1}$ с учетом равенства $\zeta_{2^{k+2}}^{2^{k+1}} = -1$ приводит к перестановке коэффициентов $a_0, a_1, \ldots, a_{2^{k+1}-1}$, и при этом некоторые из них меняют знак.

Применим лемму 9.6 для оценки сложности одного умножения в кольце $R[Y]/(Y^{2^{t-k}} + 1)$. Нам нужно выполнить 2^{t-k} умножений в R, $3 \cdot (t-k) \cdot 2^{t-k}$ сложений в R (что составляет $3 \cdot (t-k) \cdot 2^{t-k} \cdot 2^{k+1}$ сложений в T, поскольку элементы кольца R представимы в виде многочленов из $T[x]$ степени не выше $2^{k+1} - 1$), $3 \cdot (t-k) \cdot 2^{t-k}$ умножений в R на степени элемента $\zeta_{2^{t-k+1}}$, 2^{t-k} умножений в R на элемент $(2^{-1})^{t-k}$ (что опять-таки составляет $2^{t-k} \cdot 2^{k+1} = 2^{t+1}$ умножений в T на элемент $(2^{-1})^{t-k}$). Всего получается 2^{t-k} умножений в R, $6 \cdot (t-k) \cdot 2^{t+1}$ сложений в T и 2^{t+1} умножений в T на элемент $(2^{-1})^{t-k}$.

Учтем еще 2^t сложений в T на 2 шаге. Положим

$$k(t) = k + 1 = \left[\frac{t}{2}\right] + 1 \geqslant 2. \tag{9.7}$$

Тогда одно умножение в $T[x]/(x^{2^{k+1}} + 1)$ выполняется за $2^{t-k(t)+1}$ умножений в кольце $R = T[x]/(x^{2^{k(t)}} + 1)$ плюс не более, чем $12 \cdot t \cdot 2^t$ сложений в T, плюс 2^{t+1} умножение в T.

Если $t \geqslant 3$, то $k(t) < t$. Мы сводим описанным выше способом умножение в кольце $T[x]/(x^{2^t}+1)$ к умножению в кольце $T[x]/(x^{2^{k(t)}}+1)$ и далее вниз, пока не попадем в кольцо $T[x]/(x^4+1)$, где умножение выполняем любым способом за $O(1)$ операций сложения и умножения в T.

Обозначим через $M_1(2^j)$ и $A_1(2^j)$ число умножений в T и сложений в T, соответственно необходимых для выполнения указанным выше способом одного умножения в кольце $T[x]/(x^{2^j}+1)$. Тогда, при $t \geqslant 3$ выполнены неравенства

$$M_1(2^t) \leqslant 2^{t-k(t)+1} M_1(2^{k(t)}) + 2^{t+1}, \tag{9.8}$$

$$A_1(2^t) \leqslant 2^{t-k(t)+1} A_1(2^{k(t)}) + 12 \cdot t \cdot 2^t, \tag{9.9}$$

где $k(t)$ из (9.7). Положим

$$\alpha(j) = A_1(2^j)/2^j, \quad j = 2, 3, \ldots \tag{9.10}$$

Тогда $\alpha(t) \leqslant 2\alpha(k(t)) + 12t$. Предположим, что при $2 \leqslant j < t$ выполнено неравенство $\alpha(j) \leqslant cj \log j$ при некоторой абсолютной постоянной c. Тогда выполнены неравенства

$$\alpha(t) \leqslant 2ck(t)\log k(t) + 12t \leqslant 2c\left(\frac{t}{2}+1\right)\log\left(\frac{t}{2}+1\right) + 12t < ct\log t,$$

если постоянная c достаточно велика. Итак, $A_1(2^t) \leqslant 2^t \cdot ct\log t = O(2^t \cdot t\log t)$ — это доказывает утверждение 2 леммы 9.9 о количестве сложений.

Теперь положим

$$\beta(j) = \frac{M_1(2^j)}{2^j}, \quad j = 2, 3, \ldots \tag{9.11}$$

Справедливо неравенство $\beta(t) \leqslant 2\beta(k(t)) + 2$. Отсюда

$$\beta(t) \leqslant 2(2\beta(k(k(t))) + 2) + 2 = 2^2\beta(k(k(t))) + 2^2 + 2,$$

и т. д. Поскольку $k(t) \leqslant \frac{t}{2}+1$, то $k((k(t)) \leqslant \frac{1}{2}\left(\frac{t}{2}+1\right)+1 = \frac{t}{2^2}+1+\frac{1}{2}$; $k(\underset{j}{k(\ldots}(k(t))\ldots)) < \frac{t}{2^j}+2$ для всех $j \geqslant 1$. Поэтому при $j = [\log_2 t]$ будет выполнено неравенство

$$\beta(t) \leqslant 2^j \cdot c_1 + 2 + 2^2 + \ldots + 2^j < 2^j \cdot c_1 + 2^{j+1} \leqslant c_2 \cdot t,$$

где c_1, c_2 — абсолютные постоянные. Итак, $M_1(t) = O(t \cdot 2^t)$. Лемма 9.9 полностью доказана. □

§ 9.4. Дискретное преобразование Фурье и деление многочленов

Мы сохраняем обозначения и предположения из § 9.1. Функции $M(n)$ и $A(n)$ — те же, что в теореме 9.11.

Теорема 9.12. *Пусть T — коммутативное кольцо с единицей, в котором содержится элемент 2^{-1}. Пусть $F(x), G(x) \in T[x]$, $\deg F(x) \leqslant n$, $\deg G(x) \leqslant n$, где $n \geqslant 3$, и пусть задан элемент кольца T, обратный к старшему коэффициенту многочлена $G(x)$. Тогда, если мы обозначим результат деления с остатком $F(x) = Q(x)G(x) + R(x)$, где $Q(x)$, $R(x) \in T[x]$, $\deg R(x) < \deg G(x)$, то $Q(x)$ и $R(x)$ можно вычислить за $O(n \log n)$ умножений в T и $O(n \log n \log\log n)$ сложений в T.*

Доказательство. Можно предположить, что $n = \deg F(x)$, $m = \deg G(x)$ и $m \leqslant n$. Пусть $F(x) = \sum\limits_{i=0}^{n} f_i x^i$, $G(x) = \sum\limits_{j=0}^{m} g_j x^j$. Нам нужно найти элементы $q_0, \ldots, q_{n-m}, r_0, \ldots, r_{m-1} \in T$ такие, что

$$\sum_{i=0}^{n} f_i x^i = \sum_{i=0}^{n-m} q_i x^i \cdot \sum_{j=0}^{m} g_j x^j + \sum_{i=0}^{m-1} r_i x^i. \tag{9.12}$$

Заменив x на x^{-1} и умножив на x^n, преобразуем (9.12) в соотношение

$$\sum_{i=0}^{n} f_i x^{n-i} \equiv \sum_{i=0}^{n-m} q_i x^{n-m-i} \cdot \sum_{j=0}^{m} g_j x^{m-j} \pmod{x^{n-m+1}}. \tag{9.13}$$

Обозначим через $T[[x]]$ кольцо формальных степенных рядов вида $a_0 + a_1 x + a_2 x^2 + \ldots$ с коэффициентами из T. Пусть $H = H(x) \in T[[x]]$ — формальный степенной ряд, обратный к $G^*(x) = \sum\limits_{i=0}^{m} g_i x^{m-i}$. Такой ряд H существует, поскольку по условию теоремы коэффициент g_m обратим в T. Тогда, найдя $H \pmod{x^{N_1}}$ для некоторого $N_1 \in \mathbb{N}$ и умножив на него (9.13), мы найдем $Q(x) \pmod{x^{N_1}}$.

Для элемента $P \in T[[x]]$ мы обозначаем $f(P) = \dfrac{1}{P} - G^* \in T[[x]]$, если элемент $\dfrac{1}{P} \in T[[x]]$ определен. Очевидно, что $f(H) = 0$. Для $P \in T[[x]]$

положим

$$\Phi(P) = 2P - P^2 G^* = H - G^*(P-H)^2 \in T[[x]]. \tag{9.14}$$

Предположим, что $P \equiv H \pmod{x^k}$ для некоторого $k \in \mathbb{N}$. Тогда из (9.14) следует, что $\Phi(P) \equiv H \pmod{x^{2k}}$. Если задан элемент P такой, что $P \equiv H \pmod{x^k}$, то $\Phi(P) \pmod{x^{2k}}$ вычисляем следующим образом: обрезаем ряд G^* на x^{2k}, умножаем на $(-P)$, прибавляем 2 и также обрезаем на x^{2k}; полученный элемент $2 + (-P)G^* \pmod{x^{2k}}$ мы умножаем на P и обрезаем ряд на x^{2k}. Получится соотношение

$$\Phi(P) \pmod{x^{2k}} \equiv P(2 - PG^*) \pmod{x^{2k}} \equiv H \pmod{x^{2k}}.$$

То есть если $P \equiv H \pmod{x^k}$, то $\Phi(P) \equiv H \pmod{x^{2k}}$, и мы можем найти $\Phi(P) \pmod{x^{2k}}$ за два умножения в $T[x]$ многочленов меньшей степени, чем $2k$, и одно сложение в T (прибавление элемента 2).

Мы начинаем итерацию с $P = g^{-1} \in T$, где $g = g_m$ — известный нам старший коэффициент $G(x)$. Тогда $P \equiv H \pmod{x}$. Затем мы вычисляем $\Phi(P) \pmod{x^2}$, $\Phi(\Phi(P)) \pmod{x^4}$, …, $\Phi^j(P) \pmod{x^{2^j}}$, …, где Φ^j обозначает отображение Φ, примененное j раз. При этом $\Phi^j(P) \equiv H \pmod{x^{2^j}}$, и, в частности, $\Phi^j(P) \in T[[x]]$ будут обратимы. Обозначим через $A^*(k)$ и $M^*(k)$ число сложений и умножений в T, необходимых, чтобы вычислить $H \pmod{x^k}$. Мы показали, что

$$A^*(2k) \leqslant A^*(k) + 2A(2k) + 1, \tag{9.15}$$

$$M^*(2k) \leqslant M^*(k) + 2M(2k). \tag{9.16}$$

При этом $A^*(1) = M^*(1) = 0$, поскольку $g^{-1} \in T$ нам известен. Из § 9.3 следует, что $M(n) = O(n \log n)$, и можно считать, что $M(n) = C_M n \log n$ при $n > 1$, $M(1) = C_M$, где C_M — некоторая абсолютная постоянная. Аналогично можно считать, что $A(n) = C_A n \log n \log\log n$ при $n \geqslant 4$ и $A(n) = C_A$ при $n = 1, 2, 3$, где C_A — абсолютная постоянная. Тогда $M(x)$, $A(x)$, $\dfrac{M(x)}{x}$, $\dfrac{A(x)}{x}$ — растущие функции. Из (9.16) по индукции легко следует, что

$$M^*(2^t) \leqslant 4M(2^t). \tag{9.17}$$

Далее, из (9.15) следует, что

$$A^*(2^t) \leqslant 6A(2^t). \tag{9.18}$$

В самом деле,

$$A^*(2^t) \leqslant A^*(2^{t-1}) + 2A(2^t) + 1 \leqslant \ldots$$
$$\ldots \leqslant A^*(1) + 2(A(2) + A(4) + \ldots + A(2^t)) + t =$$
$$= t + 2\frac{A(2^t)}{2^t} \sum_{j=1}^{t} \frac{A(2^j) \cdot 2^t}{A(2^t)} =$$
$$= t + 2\frac{A(2^t)}{2^t} \sum_{j=1}^{t} 2^j \frac{A(2^j)/2^j}{A(2^t)/2^t} \leqslant t + 2^{1-t} A(2^t) \cdot \sum_{j=1}^{t} 2^j;$$

последнее неравенство верно, так как числа $A(2^j)/2^j$ возрастают. Отсюда

$$A^*(2^t) \leqslant t + 2A(2^t) \cdot \frac{2^{t+1} - 1}{2^t} \leqslant t + 4A(2^t) \leqslant 6A(2^t),$$

поскольку $t \leqslant 2A(2^t)$.

Выберем наименьшее $t \in \mathbb{N}$ такое, что $2^t \geqslant n - m + 1$. Тогда $2^t \leqslant 2(n-m)$, если $n > m$ (в случае $n = m$ вычисление $H \pmod{x^{n-m+1}}$ тривиально). Вычисление $H \pmod{x^{n-m+1}}$ будет выполнено не более, чем за $A^*(2^t)$ сложений и $M^*(2^t)$ умножений в T. Из (9.17) и (9.18) следует, что нам понадобится не более чем $6A(2(n-m))$ сложений и $4M(2(n-m))$ умножений в T.

Теперь вычисляем

$$H \pmod{x^{n-m+1}} \cdot \left(\sum_{i=0}^{n} f_i x^{n-i} \pmod{x^{n-m+1}} \right)$$

и обрезаем ряд на x^{n-m+1}. В силу (9.13) мы найдем $\sum\limits_{i=0}^{n-m} q_i x^{n-m-i}$. То есть мы определили коэффициенты искомого частного в формуле (9.12). Далее находим произведение $\left(\sum\limits_{i=0}^{n-m} q_i x^i \right) \cdot \left(\sum\limits_{j=0}^{m} g_j x^j \right)$ за одно умножение многочленов степени $n-m$ и m в $T[x]$, а затем определяем $\sum\limits_{i=0}^{m-1} r_i x^i$ за одно вычитание многочленов степени не выше n.

Оценим число выполняемых операций сложения и умножения. Сложений было сделано не более чем

$$6A(2(n-m)) + A(n-m) + A(\max(m, n-m)) + n. \tag{9.19}$$

Умножений было сделано не более чем

$$4M(2(n-m))+M(n-m)+M(\max(m,\,n-m)). \tag{9.20}$$

Величина (9.19) не превосходит

$$6A(2n)+A(n)+A(n)+n=O(n\log n\log\log n),$$

а величина (9.20) не превосходит $4M(2n)+2M(n)=O(n\log n)$.

Теорема доказана. □

§ 9.5. Применение дискретного преобразования Фурье в алгоритме Полларда—Штрассена

Здесь мы докажем теорему, необходимую для полного обоснования алгоритма Полларда—Штрассена из § 6.2. Доказательство теоремы можно найти в [271; 67]. Мы сохраняем обозначения и предположения из § 9.1; под сложениями мы подразумеваем и вычитания в T.

Теорема 9.13. *Пусть T — коммутативное кольцо с единицей, содержащее элемент 2^{-1}. Пусть заданы элементы $x_1,\,\ldots,\,x_n\in T$ и многочлен $F(x)\in T[x]$, причем $F(x)$ представлен либо в виде $F(x)=\sum\limits_{i=0}^{n} f_i x^i$, либо в виде $F(x)=\prod\limits_{i=1}^{n}(x-y_i)$, и $n=\deg F(x)\geqslant 3$. Тогда элементы $F(x_1),\,\ldots,\,F(x_n)$ могут быть вычислены за $O(n\log^2 n\times\log\log n)$ сложений в T и $O(n\log^2 n)$ умножений в T.*

Доказательство. Обозначим через t наименьшее натуральное число такое, что $n\leqslant 2^t$. Положим $x_i=0$ при $n<i\leqslant 2^t$.

Сначала мы вычисляем коэффициенты вспомогательных многочленов

$$G_{ij}(X)=\prod_{k=(j-1)\cdot 2^i+1}^{j\cdot 2^i}(X-x_k),\qquad 0\leqslant i\leqslant t,\qquad 1\leqslant j\leqslant 2^{t-i}. \tag{9.21}$$

При $i=0$ имеем $G_{0j}(X)=X-x_j$; $j=1,\,\ldots,\,2^t$. Предположим, что для некоторого i, $0\leqslant i\leqslant t-1$, мы уже вычислили коэффициенты 2^{t-i} многочленов $G_{ij}(X)$ степени 2^i. Тогда вычисляем набор $2^{t-(i+1)}$ многочленов $G_{i+1,j}(X)$, $j=1,\,\ldots,\,2^{t-(i+1)}$, степени 2^{i+1} за $2^{t-(i+1)}$ умножений двух соседних многочленов $G_{ij}(X)$ степени 2^i. Поэтому весь набор многочленов (9.21) может быть найден за не более чем $\sum\limits_{i=0}^{t-1}2^{t-(i+1)}A(2^i)$ сложений в T и не более чем $\sum\limits_{i=0}^{t-1}2^{t-(i+1)}M(2^i)$ умножений в T (функции $A(x)$ и $M(x)$ те же, что в теореме 9.11). Функция $A(x)/x$ по определению является

возрастающей, так как мы берем не точное значение $A(x)$, а верхнюю оценку вида $C_A \log x \log\log x$, где C_A — некоторая постоянная. Аналогично, в качестве M мы используем величину $C_M x \log x$. Тогда

$$\sum_{i=0}^{t-1} 2^{t-1} \cdot \frac{A(2^i)}{2^i} \leqslant t \cdot 2^{t-1} \cdot \frac{A(2^{t-1})}{2^{t-1}} = tA(2^{t-1}),$$

$$\sum_{i=0}^{t-i-1} 2^{t-i-1} M(2^i) \leqslant tM(2^{t-1}).$$

Дальнейшее доказательство разделяется на два случая.

1 случай. Многочлен $F(x)$ задан в виде $\sum\limits_{i=0}^{n} f_i x^i$, т. е. известны коэффициенты $f_0, \ldots, f_n \in T$. Положим $F_{t,1}(x) = F(x)$, и далее при $0 \leqslant i < t$, $1 \leqslant j \leqslant 2^{t-i}$ положим $F_{ij}(x)$ равным остатку от деления $F_{i+1,\left[\frac{j+1}{2}\right]}(x)$ на $G_{ij}(x)$ (с уменьшением i диапазон изменения j растет). Тогда при $0 \leqslant i \leqslant t$, $1 \leqslant j \leqslant 2^{t-i}$ и при $(j-1)2^i < k \leqslant j \cdot 2^i$ справедливы равенства

$$F_{ij}(x_k) = F_{i-1,2j-1}(x_k), \qquad \text{если } (2j-2)2^{i-1} < k \leqslant (2j-1)2^{i-1}; \tag{9.22}$$

$$F_{ij}(x_k) = F_{i-1,2j}(x_k), \qquad \text{если } (2j-1)2^{i-1} < k \leqslant 2j2^{i-1}. \tag{9.23}$$

Это следует из того, что если $G_{ij}(x_k) = 0$, то значение делимого и остатка в точке x_k совпадают. В самом деле, если

$$F_{ij}(x) = H(x) \cdot G_{i-1,2j-1}(x) + F_{i,2j-1}(x),$$

то при доказательстве (9.22) мы пользуемся тем, что $G_{i-1,2j-1}(x_k) = 0$ при $(2j-2)2^{i-1} < k \leqslant (2j-1)2^{i-1}$; аналогично доказывается (9.23). Следовательно, вычисление $F(x) = F_{t,1}(x)$ в точках $x_1, \ldots, x_{2^t}$ сводится к вычислению $F_{t-1,1}(x)$ и $F_{t-1,2}(x)$ в точках $x_1, \ldots, x_{2^{t-1}}$ и $x_{2^{t-1}+1}, \ldots, x_{2^t}$ соответственно. И вообще, вычисление $F_{i,j}(x_k)$, где $(j-1)2^i < k \leqslant j \cdot 2^i$, сводится к вычислению $F_{i-1,2j-1}(x_k)$ при $(2j-2)2^{i-1} < k \leqslant (2j-1) \cdot 2^{i-1}$ и $F_{i-1,2j}(x_k)$ при $(2j-1)2^{i-1} < k \leqslant 2j \cdot 2^{i-1}$ соответственно. Так мы спускаемся по первому индексу вниз, пока не доходим до $F_{0j}(x)$, где $1 \leqslant j \leqslant 2^t$. Но $F_{0j}(x)$ — константы, это остатки от деления исходного $F(x)$ на $G_{0j}(x) = x - x_j$. То есть $F_{0j}(x) = F(x_j)$ — искомые величины. Оценим число операций, необходимых для вычисления $F_{0j}(x)$. Если все многочлены $G_{ij}(x)$ уже найдены, то мы последовательно находим остатки $F_{ij}(x)$. Так как $\deg F_{i+1,j}(x) < \deg G_{i+1,j}(x) = 2^{i+1}$, то по теореме 9.12 (которая применима, так как все $G_{ij}(x)$ — унитарные многочлены) для

нахождения остатка $F_{ij}(x)$ требуется

$$O(2^{i+1}\log 2^{i+1}\log\log 2^{i+1}) \leqslant C'_A \cdot A(2^{i+1})$$

сложений в T и

$$O(2^{i+1}\log 2^{i+1}) \leqslant C'_M \cdot M(2^{i+1})$$

умножений в T (C'_A и C'_M — некоторые постоянные). Поэтому для нахождения всех многочленов $F_{ij}(x)$ при заданных $G_{ij}(x)$ требуется не более, чем $\sum\limits_{i=0}^{t-1} C'_A \cdot A(2^{i+1})2^{t-i}$ сложений в T и $\sum\limits_{i=0}^{t-1} C'_M \cdot M(2^{i+1})2^{t-i}$ умножений в T. Аналогично доказательству теоремы из предыдущего параграфа число сложений тогда оценится величиной $C''_A tA(2^t)$, а число умножений — величиной $C''_M tM(2^t)$, где C''_A и C''_M — некоторые постоянные. С учетом доказанной выше оценки сложности для вычисления всех многочленов $G_{ij}(X)$ и неравенства $2^t \leqslant 2n$, из которого следует, что $t = O(\log n)$, мы получаем утверждение теоремы в рассматриваемом случае.

2 случай. Пусть многочлен $F(x)$ задан в виде произведения $\prod\limits_{i=1}^{n}(x-y_i)$. Тогда мы уже показали, что коэффициенты многочлена $G_{t,1}(x) = \prod\limits_{k=1}^{2^t} = x^{2^t-n}\prod\limits_{k=1}^{n}(x-x_k)$ могут быть вычислены в рамках указанного в утверждении теоремы количества сложений и умножений в T. После этого вычисления мы оказываемся в условиях 1 случая, для которого теорема уже доказана.

Теорема полностью доказана. □

§ 9.6. Заключение

Применение дискретного преобразования Фурье в арифметике многочленов действительно является эффективным на практике. Дискретное преобразование Фурье можно также использовать в целочисленной арифметике. В частности, с его помощью доказывается теорема Шенхаге—Штрассена: произведение двух n-разрядных двоичных чисел можно вычислить за $O(n\log n\log\log n)$ битовых операций. Однако на практике алгоритм Шенхаге—Штрассена неэффективен, хотя и имеет наилучшую известную оценку сложности среди алгоритмов умножения целых чисел. Постоянная в символе $O(\cdot)$ в указанной оценке сложности алгоритма слишком велика, что делает его практичным лишь для чисел, записываемых несколькими тысячами десятичных цифр, см. [60; 243; 292]. В следующей главе мы опишем эффективные алгоритмы для умножения больших целых чисел.

Глава 10. Целочисленная арифметика многократной точности

§ 10.1. Введение. Сложение и вычитание

В данной главе мы описываем основные алгоритмы для выполнения арифметических операций с большими целыми числами, а также некоторые алгоритмы в кольцах вычетов $\mathbb{Z}/n\mathbb{Z}$.

Мы считаем, что числа записаны в b-ичной системе счисления, где b — фиксированное натуральное число, $b \geqslant 2$. При этом натуральное число, записываемое не более чем n цифрами в b-ичной системе счисления, мы обозначаем $u_1 \dots u_n$ (допуская, что несколько старших разрядов $u_1, \dots, u_k$ могут равняться нулю). Основание b не всегда равно 2; иногда оно соответствует размеру машинного слова, отведенному под запись обычных целых чисел. В этом случае мы работаем с массивом, содержащим большое целое число.

При работе с большими целыми числами удобно хранить знак такого числа в отдельной ячейке или переменной. Если мы хотим, например, перемножить два числа, то знак произведения мы вычисляем отдельно.

Заметим, что программы, реализующие арифметику многократной точности, лучше всего писать на ассемблере.

Опишем сложение и вычитание.

Алгоритм A (сложение неотрицательных целых чисел).
Для двух неотрицательных чисел $u_1 \dots u_n$ и $v_1 \dots v_n$ вычисляется их сумма $w_0 \dots w_n$; при этом w_0 — цифра переноса — всегда равна 0 или 1.

1 шаг. Присвоить $j := n$, $k := 0$ (здесь j идет по разрядам, k следит за переносом).

2 шаг. Присвоить $w_j := u_j + v_j + k \pmod b$, w_j — наименьший неотрицательный вычет в данном классе вычетов;

$$k = \left[\frac{u_j + v_j + k}{b}\right].$$

(Заметим, что w_j — очередная цифра, k — перенос; всегда $k = 0$ или $k = 1$. При этом, если $b = 2$ или b — размер машинного слова, то для

вычисления w_j и k не нужно использовать деления, достаточно взять соответствующий разряд (или разряды) в записи $u_j + v_j + k$.)

3 шаг. Присвоить $j := j - 1$. Если $j > 0$, то идти на шаг 2; если $j = 0$, то присвоить $w_0 := k$ и закончить работу.

Конец алгоритма.

Обоснование корректности алгоритма очевидно.

Алгоритм S (вычитание неотрицательных целых чисел).

По двум n-разрядным неотрицательным целым числам $u = u_1 \dots u_n \geqslant v = v_1 \dots v_n \geqslant 0$ вычисляется их разность $w = w_1 \dots w_n = u - v$.

Замечание 10.1. Для того, чтобы в общем случае установить, что $u_1 \dots u_n \geqslant v_1 \dots v_n$, надо пройти по цифрам, вычисляя $u_j - v_j$. Это простая проверка; с ее помощью находится знак разности $u - v$ в общем случае.

1 шаг. Присвоить $j := n$, $k := 0$ (переменная k — это заем из старшего разряда).

2 шаг. Присвоить $w_j := u_j - v_j + k \pmod b$ — наименьший неотрицательный вычет в данном классе вычетов;

$$k := \left[\frac{u_j - v_j + k}{b}\right].$$

3 шаг. $j := j - 1$. Если $j > 0$, то идти на шаг 2; при $j = 0$ закончить работу.

Конец алгоритма.

Обоснование алгоритма достаточно очевидно. При $j = n$ мы находим $w_n = u_n - v_n$, если $u_n \geqslant v_n$, и $w_n = b + u_n - v_n$, если $u_n < v_n$. Соответственно, $k = 0$ и $k = -1$ — это заем из $n - 1$ разряда. Дальнейшие рассуждения аналогичны.

§ 10.2. Умножение

Здесь мы опишем несколько методов умножения целых чисел, приводящих к эффективным на практике алгоритмам.

Алгоритм M (умножение неотрицательных целых чисел столбиком).

Для чисел $u = u_1 \dots u_n$ и $v = v_1 \dots v_m$ в системе счисления с основанием b мы находим их произведение $w = uv = w_1 \dots w_{m+n}$.

1 шаг. Присвоить $w_{m+1} := 0, \dots, w_{m+n} := 0$, $j := m$. (Значение j перемещается по номерам разрядов v от младших к старшим.)

2 шаг. Если $v_j = 0$, то присвоить $w_j := 0$ и перейти на шаг 6. (Этот шаг можно пропустить. Однако если b мало, например, $b = 2$, то v_j равно нулю с достаточно большой вероятностью. В этом случае выполнение шага 2 дает значительную экономию.)

3 шаг. Присвоить $i := n$, $k := 0$. (Значение i идет по номерам разрядов числа u, k отвечает за перенос.)

4 шаг. Присвоить $t := u_i \cdot v_j + w_{i+j} + k$, $w_{i+j} := t \pmod b$ — наименьший неотрицательный вычет в данном классе вычетов, $k := [t/b]$. (По прежнему, как и в § 10.1, при вычислении w_{i+j} и k в ряде случаев можно обходиться без деления. Легко показать, что выполняются неравенства $0 \leqslant t < b^2$, $0 \leqslant k < b$.)

5 шаг. $i := i - 1$. Если $i > 0$, то идти на шаг 4. Если $i = 0$, то присвоить $w_j := k$.

6 шаг. $j := j - 1$. Если $j > 0$, то идти на шаг 2. Если $j = 0$, то закончить работу.

Конец алгоритма.

Проведем обоснование корректности алгоритма по индукции.

Пусть $j = m$. Если на 2 шаге $v_m = 0$, то в последних $n+1$ разрядах w будут стоять нули. Если же $v_m > 0$, то на 4-м шаге мы фактически находим $u_i b^{n-i} v_m b^{m-m} = u_i v_j b^{m+n-(i+j)}$ и добавляем в разряд w_{i+j} с учетом переноса k. При этом определяется истинное значение цифры w_{i+j} и очередной перенос k. В итоге при $j = m$ после шагов 2—5 мы нашли $w_m w_{m+1} \dots w_{m+n} = u \cdot v_n$. Это основание индукции.

Теперь предположим, что мы прошли 2—6 шаги l раз и верно определили $u \cdot (v_{m-l+1} v_{m-l+2} \dots v_m) = w_{m-(l-1)} w_{m-(l-2)} \dots w_{m+n}$. Тогда после 6 шага $j = m - l$, мы уходим на 2 шаг и вычисляем $u \cdot v_{m-l} \cdot b^l$. Снова идем по разрядам u_n, …, u_1, находим $u_i b^{n-i} v_{m-l} b^l = u_i v_j b^{n+l-i}$ и соответственно изменяем цифру с номером $n + m - (n + l - i) = i + j$. На этом мы завершим наше краткое обоснование алгоритма М.

Теперь мы опишем метод умножения, предложенный Комбой, см. [92; 91]. Мы будем называть этот метод «быстрым столбиком». Нам нужно умножить $u = u_1 \dots u_n = \sum_{i=1}^{n} u_i b^{n-i}$ на $v = v_1 \dots v_m = \sum_{j=1}^{n} v_j b^{m-j}$. Будем считать, что $n \geqslant m$. В алгоритме М нам требовалось кроме mn умножений $u_i v_j$ и некоторого количества сложений определенное количество чтений записей памяти. А именно, элементы v_j мы читали m раз, элементы u_i мы читали mn раз (по n раз при каждом фиксированном j), элементы w_{i+j} мы mn раз читали и mn раз записывали для них новые значения. То есть нам нужно mn умножений и около $3mn$ чтений

записей памяти. В алгоритме M мы вычисляли uv по формуле

$$uv = \sum_{j=1} uv_j b^{m-j}. \tag{10.1}$$

Однако справедлива и другая формула:

$$uv = \sum_{s=0}^{m+n-2} b^s \left(\sum_{i=0}^{s} u_{n-i} v_{m-s+i} \right), \tag{10.2}$$

где при $l \leqslant 0$ мы полагаем $u_l = v_l = 0$. В самом деле,

$$uv = \sum_{s=0}^{m+n-2} b^s \sum_{\substack{i+j=s \\ 0 \leqslant i \leqslant n-1 \\ 0 \leqslant j \leqslant m-1}} u_{n-i} v_{m-j} = \sum_{i=0}^{n-1} \sum_{i \leqslant s \leqslant m-1+i} u_{n-i} v_{m-s+i} b^s =$$

$$= \sum_{s=0}^{m+n-2} b^s \sum_{s-m+1 \leqslant i \leqslant s} u_{n-i} v_{m-s+i}. \tag{10.3}$$

Формула (10.3) эквивалентна формуле (10.2). Действительно, если $s - m + 1 < 0$, т. е. $s < m - 1$, то при $i < 0$ величина u_{n-i} равна 0 по определению. Если же $s - m + 1 > 0$, т. е. $s > m - 1$, то при $0 \leqslant i < s - m + 1$ значение $i + m - s < 1$, и $v_{m-s+i} = 0$ по определению.

Теперь воспользуемся формулой (10.2), чтобы умножить u на v.

Алгоритм FM («быстрый столбик»)..

1 шаг. $t := 0$.

2 шаг. (цикл) Для s от 0 до $m + n - 1$ с шагом 1 выполнить шаги 3 и 4.

3 шаг. Для i от 0 до s с шагом 1 выполнить присвоение

$$t := t + u_{n-i} \cdot v_{m-s+i}.$$

4 шаг. Присвоить $w_{m+n-s} := t \pmod b$ — наименьший неотрицательный вычет по модулю b (опять-таки, это не деление, а чтение записи памяти, если $b = 2$ или b — размер машинного слова);

$$t := [t/b].$$

Конец алгоритма.

Корректность алгоритма следует из формулы (10.2).

Операций умножения здесь столько же, сколько в алгоритме M, поскольку каждое u_i надо умножить на каждое v_j, только в другом порядке. Однако количество чтений записей памяти сокращается: мы читаем u_i и v_j столько раз, сколько перемножаем — т. е. $2mn$ раз (mn раз

читаем u_i и mn раз читаем v_j). И еще требуется $m+n$ записей в память значений w_{m+n-s}. В итоге число чтений записей памяти в «быстром столбике» существенно меньше, чем в алгоритме M, и реализация алгоритма FM дает реальный выигрыш во времени на практике (если писать программу на ассемблере и если b — это размер слова в ассемблере).

Теперь опишем *метод Карацубы* для умножения целых чисел (см. [23]). Предположим, что у нас имеется два $2n$-разрядных числа в двоичной системе счисления:

$$u = u_{2n-1} \dots u_0, \qquad v = v_{2n-1} \dots v_0.$$

Положим $u = 2^n u' + u''$, $v = 2^n v' + v''$, где

$$u' = u_{2n-1} \dots u_n, \qquad u'' = u_{n-1} \dots u_0,$$
$$v' = v_{2n-1} \dots v_n, \qquad v'' = v_{n-1} \dots v_0.$$

Тогда

$$uv = (2^{2n} + 2^n)u'v' + 2^n(u' - u'')(v'' - v') + (2^n + 1)u''v''. \tag{10.4}$$

Поэтому для умножения двух $2n$-разрядных чисел по формуле (10.4) нужно три умножения n-разрядных чисел и $O(1)$ сложений, вычитаний и сдвигов $4n$-разрядных чисел, которые делаются за $O(n)$ битовых операций. Если обозначить через $T(n)$ число битовых операций для умножения двух n-разрядных чисел, то

$$T(2n) \leqslant 3T(n) + cn, \tag{10.5}$$

где c — некоторая абсолютная постоянная. Из (10.5) по индукции следует неравенство

$$T(2^k) \leqslant c(3^k - 2^k), \qquad k = 1, 2, 3, \dots \tag{10.6}$$

Действительно, при $k = 1$ $T(2) \leqslant c$ (это обеспечивается за счет выбора постоянной c). Далее, если (10.6) верно для k, то

$$T(2^{k+1}) \leqslant 3T(2^k) + c2^k \leqslant 3c(3^k - 2^k) + c2^k = c(3^{k+1} - 2^{k+1}).$$

Поскольку умножение двух n-разрядных чисел сводится к умножению $2^{[\log_2 n]+1}$-разрядных чисел (старшие биты при необходимости полагаем равными нулю), то из (10.6) получим неравенство

$$T(n) \leqslant T(2^{[\log_2 n]+1}) \leqslant c3^{[\log_2 n]+1} \leqslant c_1 3^{\log_2 n} = c_1 n^{\log_2 3}.$$

Это есть оценка сверху количества битовых операций, требуемых для умножения двух n-разрядных чисел методом Карацубы.

Метод Карацубы становится более эффективным, чем «быстрый столбик», для достаточно больших чисел. По разным источникам, граница находится где-то между числами порядка 2^{450} и 2^{640}. Поскольку в методе Карацубы умножение $2n$-разрядных чисел сводится к умножению n-разрядных, то, сделав несколько тактов алгоритма Карацубы, мы приходим к умножению чисел того размера, где более эффективен «быстрый столбик», и с его помощью завершаем умножение. В этом и заключается практичный алгоритм умножения целых чисел.

Замечание 10.2. Алгоритм Карацубы был впоследствии обобщен Тоомом и Куком, см. [39; 94]. В частности, можно показать, что в алгоритме Тоома—Кука

$$T(n) = O\bigl(n2^{\sqrt{2\log_2 n}}\log_2 n\bigr).$$

Алгоритм Тоома—Кука описан в [25, § 4.3.3]. Там же описан модулярный метод Шенхаге. Еще лучшую, чем алгоритм Тоома—Кука, оценку сложности имеет алгоритм Шенхаге—Штрассена, о котором было вкратце рассказано в гл. 9.

В работах [115] и [279] описаны методы распараллеливания умножения целых чисел многократной точности.

§ 10.3. Деление

В этом параграфе мы опишем методы деления целых чисел многократной точности.

Для начала рассмотрим алгоритм деления многоразрядного числа на одноразрядное.

Алгоритм DO (деление на одноразрядное число).

Находим частное $w = w_1 \ldots w_n$ от деления числа $u = u_1 \ldots u_n$ в системе счисления с основанием b на одноразрядное число v, $1 \leqslant v < b$, и остаток $r = u - vw$.

1 шаг. $r := 0$, $j := 1$.

2 шаг. $w - j := \left[\dfrac{rb + u_j}{v}\right]$; $r := rb + u_j \pmod v$ — наименьший неотрицательный вычет в данном классе вычетов.

3 шаг. $j := j + 1$. Если $j \leqslant n$, то идти на 2 шаг. Если $j > n$, то выдать $w = w_1 \ldots w_n$ и r.

Конец алгоритма.

Корректность алгоритма DO очевидна — это обычное деление углом.

Теперь рассмотрим простой алгоритм деления многоразрядных целых чисел из [180, гл. 14]. Пусть b — основание системы счисления, $u = u_n \dots u_1 u_0$, $v = v_t \dots v_1 v_0$ — натуральные числа, $n \geqslant t \geqslant 1$, $v_t \neq 0$ (случай $t = 0$, т. е. деление на одноразрядное число, рассмотрен выше в алгоритме DO).

Алгоритм деления.

На входе u, v; на выходе частное $q = q_{n-t} \dots q_0$ и остаток $r = r_t \dots r_0$, $u = qv + r$, $0 \leqslant r < v$.

1 шаг. Для j от 0 до $n - t$ присвоить $q_j := 0$.

2 шаг. До тех пор, пока $u \geqslant vb^{n-t}$, выполнять следующие действия:

$$q_{n-t} := q_{n-t} + 1, \quad u := u - vb^{n-t}$$

(здесь определяется старшая цифра частного).

3 шаг. Для $i = n, n-1, \dots, t+1$ выполнять пункты 1—4:

1) если $u_i \geqslant v_t$, то присвоить $q_{i-t-1} := b - 1$, иначе присвоить

$$q_{i-t-1} := \left[\frac{u_i b + u_{i-1}}{v_t}\right];$$

2) до тех пор, пока $q_{i-t-1}(v_t b + v_{t-1}) > u_i b^2 + u_{i-1} b + u_{i-2}$, выполнять

$$q_{i-t-1} := q_{i-t-1} - 1;$$

(заметим, что всегда будет выполняться неравенство $q_{i-t-1} \geqslant 0$);

3) присвоить $u := u - q_{i-t-1} b^{i-t-1} v$;

4) если $u < 0$, то присвоить

$$u := u + vb^{i-t-1}, \quad q_{i-t-1} := q_{i-t-1} - 1.$$

4 шаг. $r := u$. Выдать q и r.

Конец алгоритма.

Покажем, что алгоритм работает верно. После 1 шага $q = 0$. После второго шага найдена старшая цифра q_{n-t} частного, и текущее значение u меньше, чем vb^{n-t}. Возьмем первое значение $i = n$; нам нужно определить цифру частного q_{n-t-1}. Если $u_n \geqslant v_t$ (неравенство 1 пункта 3 шага), то наибольшее возможное значение q_{n-t-1} равно $b - 1$ (например, при $u_n = b - 1$, $v_t = 1$). Если же $u_n < v_t$, то наибольшее возможное значение q_{n-t-1} равно $\left[\frac{u_n b + u_{n-1}}{v_t}\right]$. Действительно,

$$u_n b^n + u_{n-1} b^{n-1} + \dots + u_0 < \left(\left[\frac{u_n b + u_{n-1}}{v_t}\right] + 1\right) b^{n-t-1} v,$$

так как справедливо даже более сильное неравенство:

$$u_n b^n + \ldots + u_0 < \left(\left[\frac{u_n b + u_{n-1}}{v_t}\right] + 1\right) b^{n-t-1} v_t b^t = \\ = \left(\left[\frac{u_n b + u_{n-1}}{v_t}\right] + 1\right) b^{n-1} v_t.$$

Последнее верно, так как

$$u_n b^n + u_{n-1} b^{n-1} = (u_n b + u_{n-1}) b^{n-1} < \left(\left[\frac{u_n b + u_{n-1}}{v_t}\right] + 1\right) b^{n-1} v_t,$$

и поскольку обе части строгого неравенства делятся нацело на b^{n-1}, добавление к левой части величины $u_{n-2} b^{n-2} + \ldots + u_0$, меньшей, чем b^{n-1}, сохраняет неравенство. Итак, мы обосновали 1-й пункт 3-го шага. Здесь величина q_{i-t-1} не меньше истинного значения цифры частного с номером $i - t - 1$, и не больше, чем $b - 1$, так как при $v_t \geqslant u_n + 1$ имеем

$$\left[\frac{u_n b + u_{n-1}}{v_t}\right] \leqslant \left[\frac{u_n b + u_{n-1}}{u_n + 1}\right] = \left[b + \frac{u_{n-1} - b}{u_n + 1}\right] < b.$$

Во втором пункте 3 шага мы находим более точное значение q_{i-t-1}. В самом деле, пусть

$$(q_{n-t-1} + 1)(v_t b + v_{t-1}) > u_n b^2 + u_{n-1} b + u_{n-2}, \tag{10.7}$$

$$q_{n-t-1}(v_t b^t + v_{t-1}) \leqslant u_n b^2 + u_{n-1} b + u_{n-2}. \tag{10.8}$$

Из первого неравенства следует, что

$$b^{n-t-1}(q_{n-t-1} + 1)(v_t b^t + v_{t-1} b^{t-1}) > u_n b^n + u_{n-1} b^{n-1} + u_{n-2}.$$

Так как обе части делятся на b^{n-2} и $u_{n-3} b^{n-3} + \ldots + u_0 < b^{n-2}$, то

$$b^{n-t-1}(q_{n-t-1} + 1)(v_t b^t + v_{t-1} b^{t-1}) > u.$$

Тем более $b^{n-t-1}(q_{n-t-1} + 1)v > u$; значит, $q_{n-t-1} + 1$ больше искомой цифры частного, а q_{n-t-1} — не меньше.

Пусть для найденного значения q_{n-t-1} выполнено неравенство (10.8). Покажем, что тогда q_{n-t-1} либо равно истинной цифре частного, либо на единицу больше. Предположим противное. Тогда q_{n-t-1} больше истинной цифры хотя бы на 2, т. е.

$$u < (q_{n-t-1} - 1) b^{n-t-1} v. \tag{10.9}$$

Отсюда

$$u_n b^n + u_{n-1} b^{n-1} + u_{n-2} b^{n-2} < (q_{n-t-1} - 1) b^{n-t-1} (v_t b^t + \ldots\ v_0).$$

Применяя (10.8), получим

$$q_{n-t-1}(v_t b + v_{t-1})b^{n-2} < \\ < (q_{n-t-1} - 1)b^{n-t-1}(v_t b^t + v_{t-1}b^{t-1} + v_{t-2}b^{t-2} + \ldots + v_0).$$

Пользуясь делимостью на b^{n-2}, неравенствами $v_{t-2}b^{t-2} + \ldots + v_0 < b^{t-1}$ и $0 \leqslant q_{n-t-1} < b$, получим, что

$$q_{n-t-1}(v_t b + v_{t-1})b^{n-2} \leqslant (q_{n-t-1} - 1)b^{n-t-1}(v_t b^t + v_{t-1}b^{t-1}),$$

а это невозможно.

Итак, найденное во 2 пункте 3 шага значение q_{n-t-1} равно истинному значению цифры частного, или на единицу больше. Отсюда следует обоснование 3 и 4 пунктов 3 шага.

После прохождения 3 шага для $i = n$ будет выполнено неравенство

$$u = u_n b^n + \ldots + u_0 < vb^{n-t-1} = b^{n-t-1}(v_t b^t + \ldots + v_0).$$

Отсюда следует, что $u_n = 0$. Поэтому мы возвращаемся на 3-й шаг к значению $i = n - 1$, и т. д.

Мы обосновали корректность работы 3-го шага для $i = n$. Для меньших значений i все рассуждения повторяются дословно.

На этом завершается обоснование данного алгоритма деления.

Теперь мы опишем более тонкий алгоритм деления целых чисел многократной точности, следуя [25, гл. 4]. Мы дадим лишь несколько более подробные доказательства вспомогательных утверждений, чем это сделано в книге [25]. Заметим, что в книге [25] приведена также блок-схема алгоритма деления.

Пусть $u = u_0 \ldots u_n$ и $v = v_1 \ldots v_n$ — неотрицательные целые числа, записанные в системе счисления с основанием b, причем $v_1 > 0$, $u/v < b$. Нам нужно найти частное от деления u на v

$$q = [u/v], \tag{10.10}$$

$0 \leqslant q \leqslant b - 1$. Нахождение q при сделанных нами предположениях мы будем называть *основной задачей*.

Заметим, что $u/v < b$ тогда и только тогда, когда $u/b < v$, что равносильно неравенству

$$u_0 \ldots u_{n-1} < v_1 \ldots v_n. \tag{10.11}$$

Обозначим

$$\hat{q} = \min\Big(\Big[\frac{u_0 b + u_1}{v_1}\Big],\ b - 1\Big). \tag{10.12}$$

Значение $\hat{q}$ вычислить нетрудно, нужно найти частное от деления двухразрядного числа u_0b+u_1 на одноразрядное v_1 (см. алгоритм DO выше).

Теорема 10.3. *В условиях основной задачи справедливо неравенство* $\hat{q} \geqslant q$.

Доказательство. Поскольку $q=[u/v] \leqslant b-1$, теорема верна при $\hat{q}=b-1$. Пусть $\hat{q}<b-1$. Тогда $\hat{q}=\left[\dfrac{u_0b+u_1}{v_1}\right]$. Из этого следует, что

$$\hat{q}v_1 \geqslant u_0b+u_1-v_1+1. \tag{10.13}$$

В самом деле, если $\dfrac{u_0b+u_1}{v_1} \in \mathbb{Z}$, то (10.13) верно, так как $v_1 \geqslant 1$. Если же $\dfrac{u_0b+u_1}{v_1}$ — не целое число, то $\dfrac{u_0b+u_1}{v_1}=\left[\dfrac{u_0b+u_1}{v_1}\right]+\dfrac{k}{v_1}$, где $k \in \mathbb{N}$, $1 \leqslant k \leqslant v_1-1$. Тогда

$$\hat{q}=\frac{u_0b+u_1}{v_1}-\frac{k}{v_1} \geqslant \frac{u_0b+u_1}{v_1}-\frac{v_1-1}{v_1},$$

откуда следует (10.13).

С помощью (10.13) получаем неравенства

$$\begin{aligned} u-\hat{q}v \leqslant u-\hat{q}v_1b^{n-1} \leqslant \\ \leqslant u_0b^{n-2}+\ldots+u_n-(u_0b^n+u_1b^{n-1}-v_1b^{n-1}+b^{n-1})= \\ =u_2b^{n-2}+\ldots u_n-b^{n-1}+v_1b^{n-1}<v_1b^{n-1} \leqslant v. \end{aligned}$$

Так как $u \geqslant qv$ и $u<(\hat{q}+1)v$, то $\hat{q} \geqslant q$, что и требовалось доказать. □

Теорема 10.4. *Если в условиях основной задачи также выполнено неравенство* $v_1 \geqslant [b/2]$, *то*

$$\hat{q}-2 \leqslant q.$$

Доказательство. Так как (по теореме 10.3) $q \leqslant \hat{q}$, то, предположив, что $q<\hat{q}-2$, мы придем к противоречию. Пусть $\hat{q} \geqslant q+3$. По определению

$$\hat{q} \leqslant \frac{u_0b+u_1}{v_1}=\frac{u_0b^n+u_nb^{n-1}}{v_1b^{n-1}}<\frac{u}{v-b^{n-1}}.$$

При этом $v>b^{n-1}$, так как если $v=b^{n-1}$, то $q=\left[\dfrac{u}{v}\right]=u_0b+u_1$, и по условию $q \leqslant b-1$. Из этого следует, что $\hat{q}=u_0b+u_1=q$, а это противоречит сделанному предположению.

Поскольку q — частное, то $q>u/v-1$. Тогда

$$3 \leqslant \hat{q}-q<\frac{u}{v-b^{n-1}}-\frac{u}{v}+1=\frac{u}{v}\cdot\frac{b^{n-1}}{v-b^{n-1}}+1.$$

Значит

$$\frac{u}{v}\cdot\frac{b^{n-1}}{v-b^{n-1}}>2,$$

$$\frac{u}{v}>2\frac{v-b^{n-1}}{b^{n-1}}=2\Big(\frac{v}{b^{n-1}}-1\Big)\geqslant 2(v_1-1).$$

Далее, так как $\hat q\leqslant b-1$, то

$$b-u\geqslant\hat q-3\geqslant q=[u/v]\geqslant 2v_1-2.$$

Поэтому $v_1\leqslant\dfrac{b}{2}-1$, $v_1<[b/2]$, что противоречит условию теоремы. □

Замечание 10.5. Мы показали, что если $v_1\geqslant[b/2]$, то пробное частное $\hat q$ никогда не отличается от истинного частного q больше, чем на две единицы.

Теперь обратимся к *задаче нормализации*: как, находясь в условиях основной задачи, попасть в условия теоремы 10.4? Оказывается, следует умножить u и v на $\left[\dfrac{b}{v_1+1}\right]$. При этом частное q не изменится, и будет выполнена следующая теорема.

Теорема 10.6. *В условиях основной задачи у числа* $v'=v\left[\dfrac{b}{v_1+1}\right]$ *старший разряд будет не меньше, чем* $[b/2]$, *и число* v' *останется* n*-разрядным. При этом число* $u'=u\left[\dfrac{b}{v_1+1}\right]$ *будет не более чем* $(n+1)$*-разрядным.*

Доказательство. Покажем сначала, что если $a,\,c\in\mathbb{N}$, $1\leqslant a<c$, то

$$\left[\frac{c}{2}\right]\leqslant a\left[\frac{c}{a+1}\right]<(a+1)\left[\frac{c}{a+1}\right]\leqslant c. \tag{10.14}$$

Второе и третье неравенства в (10.14) очевидны; докажем первое. Рассмотрим сначала случай $a\geqslant[c/2]$. Тогда, поскольку $c\geqslant a+1$, то $a\left[\dfrac{c}{a+1}\right]\geqslant a\geqslant\left[\dfrac{c}{2}\right]$.

Пусть теперь $1\leqslant a<[c/2]$. Тогда

$$a\left[\frac{c}{a+1}\right]>a\Big(\frac{c}{a+1}-1\Big)\geqslant\frac{c}{2}-1\geqslant\left[\frac{c}{2}\right]-1.$$

Здесь мы воспользовались тем, что

$$a\Big(\frac{c}{a+1}-1\Big)-\frac{c}{a+1}+1=\frac{2ac-2a^2-2a-ca-c+2a+2}{2a+2}=$$

$$=\frac{ac-2a^2-c+2}{2a+2}=\frac{(a-1)(c-2a-2)}{2a+2}\geqslant 0,$$

так как из неравенства $a < [c/2] \leqslant c/2$ следует, что $a + 1 \leqslant c/2$, $c \geqslant 2a + 2$. Итак, формула (10.14) доказана.

Теперь докажем теорему 10.6. Положим в (10.14) $a = v_1$, $c = b$. Тогда

$$\left[\frac{b}{2}\right] b^{n-1} \leqslant v_1 b^{n-1} \left[\frac{b}{v_1 + 1}\right], \tag{10.15}$$

$$v_1 \left[\frac{b}{v_1 + 1}\right] < (v_1 + 1)\left[\frac{b}{v_1 + 1}\right] \leqslant b. \tag{10.16}$$

Так как одно из неравенств (10.16) строгое, то из (10.15) и (10.16) получим, что

$$\left[\frac{b}{2}\right] b^{n-1} \leqslant v_1 \left[\frac{b}{v_1 + 1}\right] b^{n-1} < b \cdot b^{n-1}. \tag{10.17}$$

Однако $v_1 \left[\dfrac{b}{v_1 + 1}\right]$ еще не является старшим разрядом числа v', так как возможен перенос из младших разрядов. Но если этот перенос будет строго меньше, чем $\left[\dfrac{b}{v_1 + 1}\right]$, то старший разряд числа v' будет строго меньше, чем $(v_1 + 1)\left[\dfrac{b}{v_1 + 1}\right]$, а это число не превосходит b. Поэтому v' останется n-разрядным. Осталось показать лишь, что перенос из младших разрядов v' в n-й строго меньше, чем $\left[\dfrac{b}{v_1 + 1}\right]$.

Перенос из последнего разряда v' в предпоследний составит не более чем

$$\left[\frac{(b-1)}{b}\left[\frac{b}{v_1 + 1}\right]\right] = \left[\left(1 - \frac{1}{b}\right)\left[\frac{b}{v_1 + 1}\right]\right] < \left[\frac{b}{v_1 + 1}\right],$$

поскольку $1 - \dfrac{1}{b} < 1$. Далее применим индукцию: предположим, что перенос из $(j+1)$-го разряда в j-й строго меньше, чем $\left[\dfrac{b}{v_1 + 1}\right]$. Тогда перенос из j-го в $(j-1)$-й разряд будет не превосходить

$$\left[\frac{\left(\left[\dfrac{b}{v_1 + 1}\right] - 1\right) b^{n-j} + v_j \left[\dfrac{b}{v_1 + 1}\right] b^{n-j}}{b^{n-j+1}}\right] \leqslant$$

$$\leqslant \left[\frac{\left[\dfrac{b}{v_1 + 1}\right] - 1 + (b-1)\left[\dfrac{b}{v_1 + 1}\right]}{b}\right] =$$

$$= \left[\left[\frac{b}{v_1 + 1}\right] - \frac{1}{b}\right] < \left[\frac{b}{v_1 + 1}\right].$$

Итак, мы доказали, что v' останется n-разрядным. Из (10.17) следует, что старший разряд v' будет не меньше $[b/2]$. Наконец, u' будет не более чем $(n+1)$-разрядным, так как $\dfrac{u'}{v'} = \dfrac{u}{v} < b$. $\square$

Теперь опишем алгоритм деления целых чисел многократной точности, основанный на теоремах 10.3—10.6.

Алгоритм D (деление неотрицательных целых чисел).

Для чисел $u = u_1 \dots u_{m+n}$ и $v = v_1 \dots v_n$, записанных в системе счисления с основанием b, где $v_1 \neq 0$, находим частное $\left[\dfrac{u}{v}\right] = q_0 \dots q_m$ и остаток $r = u - qv = r_1 \dots r_n$. Здесь $m \geqslant 0$. Если $n = 1$, то применяем алгоритм DO. Далее считаем, что $n \geqslant 2$.

1 шаг (нормализация). Присвоить $d := \left[\dfrac{b}{v_1 + 1}\right]$,

$$u_0 u_1 \dots u_{m+n} := u_1 \dots u_{m+n} \cdot d, \qquad v_1 \dots v_n := v_1 \dots v_n \cdot d.$$

(Из теоремы 10.6 следует, что новое значение v_1 больше или равно $[b/2]$.)

2 шаг. $j := 0$.

3 шаг (обеспечение выполнения условий основной задачи). Присвоить $l := 0$. Если $\dfrac{u_j \dots u_{j+n}}{v} \geqslant b$, то выполнены условия основной задачи; мы переходим на 4 шаг. Если $\dfrac{u_j \dots u_{j+n}}{v} < b$, то

$$u_j \dots u_{j+n} := u_j \dots \ u_{j+n} - bv, \qquad l := l + 1.$$

Снова проверить выполнение неравенства $\dfrac{u_j \dots u_{j+n}}{v} < b$; если да, то перейти на 4 шаг. Иначе еще раз присвоить

$$u_j \dots u_{j+n} := u_j \dots u_{j+n} - bv, \qquad l := l + 1.$$

Замечание 10.7. Покажем, что после 3 шага будет выполнено неравенство $\dfrac{u}{v} < b$, т. е. выполнены условия основной задачи. Поскольку $v_1 \geqslant [b/2]$, то $v = v_1 \dots v_n \geqslant [b/2] b^{n-1}$. Также $u_j \dots u_{j+n} \leqslant b^{n+1} - 1$. Покажем, что

$$u_j \dots u_{j+n} < 3bv. \tag{10.18}$$

Действительно,

$$3bv \geqslant 3bv_1 b^{n-1} \geqslant 3b^n [b/2] > b^{n+1} - 1 \geqslant u_j \dots \ u_{j+n},$$

так как при $b = 2$ неравенство

$$3 \cdot 2^n [2/2] > 2^{n+1} - 1$$

очевидно, а при $b \geqslant 3$ либо $[b/2] = b/2$, либо $[b/2] = \frac{b}{2} - \frac{1}{2}$, и тогда

$$1 + 3b^n[b/2] \geqslant 1 + 3b^n\left(\frac{b}{2} - \frac{1}{2}\right) = 1 + \frac{3}{2}b^{n+1} - \frac{3}{2}b^n > b^{n+1},$$

так как $\frac{1}{2}b^{n+1} \geqslant \frac{3}{2}b^n$. Итак, после 3 шага $u_j \dots u_{j+n} < bv$.

Замечание 10.8. Значение переменной l показывает, что из первоначального значения $u_j \dots u_{j+n}$ мы вычли lbv. Впоследствии нам надо будет добавить lb в соответствующий разряд итогового частного на 9 шаге.

4 шаг (нахождение $\hat{q}$). На этом шаге выполнено неравенство $u_j \leqslant v_1$, так как, если $u_j > v_1$, то

$$u_j u_{j+1} \dots u_{j+n} - b \cdot v_1 \dots v_n \geqslant u_j b^n - (v_1 b^n + v_2 b^{n-1} + \dots + v_n b) \geqslant$$
$$\geqslant b^n - (v_2 b^{n-1} + \dots v_n b) > 0,$$

что противоречит условию основной задачи, которое выполнено.

Если $u_j = v_1$, то $\left[\frac{u_j b + u_{j+1}}{v_1}\right] = \left[b + \frac{u_{j+1}}{v_1}\right] \geqslant b$, и тогда мы присваиваем $\hat{q} := b - 1$.

Если $u_j < v_1$, то $\left[\frac{u_j b + u_{j+1}}{v_1}\right] \leqslant b - 1$, так как иначе

$$\frac{u_j b + u_{j+1}}{v_1} \geqslant b, \quad u_j b + u_{j+1} \geqslant b v_1,$$

что невозможно, поскольку

$$b - 1 \geqslant u_{j+1}, \quad b(v_1 - u_j) \geqslant b.$$

В этом случае мы присваиваем

$$\hat{q} = \left[\frac{u_j b + u_{j+1}}{v_1}\right].$$

5 шаг. Проверить выполнение неравенства

$$v_2 \hat{q} > (u_j b + u_{j+1} - \hat{q} v_1) b + u_{j+2}. \tag{10.19}$$

Если оно выполнено, то $\hat{q} := \hat{q} - 1$. Снова проверить (10.19), и если оно выполнено, то присвоить еще раз $\hat{q} := \hat{q} - 1$.

Замечание 10.9. Поскольку $n \geqslant 2$, то v — не менее чем двухразрядное, а число $u = u_0 \dots u_{m+n}$ — не менее чем трехразрядное; значит, цифра u_{j+2} с номером $j + 2$ определена.

Замечание 10.10. После 4 шага выполнены условия теорем 10.3 и 10.4. Значит, $\hat{q} - 2 \leqslant q \leqslant \hat{q}$, т. е. $\hat{q}$ равно либо q, либо $q + 1$, либо

$q+2$. Положим $\tilde{r} := u_j b + u_{j+1} - \hat{q} v_1$ — выражение, стоящее в формуле (10.19) в скобках. Покажем, что если

$$v_2 \hat{q} > \tilde{r} b + u_{j+2}, \tag{10.20}$$

то $\hat{q} > q$. В самом деле, надо доказать, что

$$u_j \dots u_{j+n} - \hat{q} v < 0.$$

Справедливы следующие соотношения:

$$\begin{aligned} u_j \dots u_{j+n} - \hat{q} v &\leqslant u_j \dots u_{j+n} - \hat{q} v_1 b^{n-1} - \hat{q} v_2 b^{n-2} = \\ &= u_{j+2} b^{n-2} + \dots + u_{j+n} + u_j b^n + u_{j+1} b^{n-1} - \hat{q} v_1 b^{n-1} - \hat{q} v_2 b^{n-2} = \\ &= \tilde{r} b^{n-1} + u_{j+2} b^{n-2} + \dots + u_{j+n} - \hat{q} v_2 b^{n-2} < \\ &< \tilde{r} b^{n-1} + (u_{j+2} + 1) b^{n-2} - \hat{q} v_2 b^{n-2} = \\ &= b^{n-2} (\tilde{r} b + u_{j+2} + 1 - \hat{q} v_2). \end{aligned} \tag{10.21}$$

Если выполнено неравенство (10.20), то в формуле (10.21)

$$\tilde{r} b + u_{j+2} + 1 - \hat{q} v_2 < \tilde{r} b + u_{j+2} + 1 - (\tilde{r}\ b + u_{j+2}) = 1.$$

Поэтому

$$\tilde{r} b + u_{j+2} + 1 - \hat{q}\ v_2 < 0,$$

и из (10.21) следует, что

$$u_j \dots u_{j+n} - \hat{q}\ v < 0,$$

т. е. $\hat{q} \geqslant q+1$. Следовательно, надо заменить $\hat{q}$ на $\hat{q} - 1$ — это будет лучшим приближением к q. Эта процедура 5 шага делается не более двух раз, так как $\hat{q} \leqslant q+2$ по теореме 10.4.

Замечание 10.11. Сейчас для текущего значения $\hat{q}$, удовлетворяющего неравенству $q \leqslant \hat{q} \leqslant q+2$, и для значения $\tilde{r} = u_j b + u_{j+1} - \hat{q} v_1$ выполняется неравенство

$$v_2 \hat{q} \leqslant b \tilde{r} + u_{j+2}. \tag{10.22}$$

Докажем, что в этом случае $\hat{q}$ равно либо q, либо $q+1$. Предположим противное, т. е. что $q = \hat{q} - 2$. Но тогда

$$\begin{aligned} u_j \dots u_{j+n} &< (\hat{q} - 1) v < \hat{q} (v_1 b^{n-1} + (v_2 + 1) b^{n-2}) - v < \\ &< \hat{q} v_1 b^{n-1} + \hat{q} v_2 b^{n-2} + b^{n-1} - v, \end{aligned}$$

поскольку $\hat{q}b^{n-2} < b^{n-1}$ по определению $\hat{q}$. Отсюда, с учетом (10.22), получим

$$u_j \dots u_{j+n} < \hat{q}v_1 b^{n-1} + (b\tilde{r} + u_{j+2})b^{n-2} + b^{n-1} - v =$$
$$= \hat{q}v_1 b^{n-1} + (u_j b^2 + u_{j+1}b + u_{j+2} - \hat{q}v_1 b)b^{n-2} + b^{n-1} - v \leqslant$$
$$\leqslant u_j b^n + u_{j+1}b^{n-1} + u_{j+2}b^{n-2}, \quad (10.23)$$

поскольку $b^{n-1} - v \geqslant 0$. Формула (10.23) означает, что $u_j \dots u_{j+n} < < u_j \dots u_{j+n}$, но это невозможно.

Итак, $\hat{q}$ равно q или $\hat{q}$ равно $q+1$.

6 шаг. Находим разность

$$u_j \dots u_{j+n} - \hat{q}v_1 \dots v_n \quad (10.24)$$

и заносим ее абсолютную величину в $u_j \dots u_{j+n}$, а знак в переменную δ.

7 шаг. $q_j := \hat{q}$. Если разность (10.24) неотрицательна, то уходим на 9 шаг.

8 шаг. Присваиваем $q_j := q_j - 1$ (так как разность (10.24) в этом случае отрицательна, то $\hat{q} = q+1$), а также

$$u_j \dots u_{j+n} := v_1 \dots v_n - u_j \dots \ u_{j+n}.$$

9 шаг. К значению q_j прибавляем величину lb из 3 шага.

Замечание 10.12. На 3-м шаге мы обеспечивали выполнение условий основной задачи, вычитая из исходного числа $u_j \dots u_{j+n}$ число lbv. Поэтому к найденному частному q_j от деления нового (после 3 шага) числа $u_j \dots u_{j+n}$ на v надо добавить lb. После этого значение q_j уже не всегда будет цифрой в b-ичной системе счисления, оно может стать больше, чем b.

10 шаг. Сейчас в текущем значении числа $u_j \dots u_{j+n}$ цифра u_j равна 0, так как в нем фактически стоит остаток от деления на n-разрядное число $v_1 \dots v_n$. Присваиваем $j := j+1$. Если $j \leqslant m$, то возвращаемся на 3 шаг.

11 шаг. Искомое частное от деления u на v равно

$$q := q_0 b^m + q_1 b^{m-1} + \dots \ q_{m-1}b + q_m$$

(здесь q_j уже могут не быть цифрами в b-ичной системе счисления, см. замечание на шаге 9). Остаток от деления u на v равен

$$r = r_1 \dots r_n := \frac{u_{m+1} \dots \ u_{m+n}}{d},$$

где d — из 1 шага алгоритма, $u_{m+1} \dots u_{m+n}$ — значение, полученное после последнего выполнения 10 шага.

Конец алгоритма.

Корректность работы алгоритма деления мы обосновали по ходу его описания.

Замечание 10.13. Описание некоторых алгоритмов для выполнения арифметики многократной точности можно найти в книгах [101, гл. 9; 180, гл. 14]. Обзор арифметики многократной точности содержится в работе [70].

§ 10.4. Некоторые алгоритмы модулярной арифметики

В этом параграфе мы описываем некоторые алгоритмы для вычисления в кольцах вычетов $\mathbb{Z}/n\mathbb{Z}$.

Начнем с *китайской теоремы об остатках*. В Приложении мы привели формулировку этой теоремы о решении системы сравнений

$$\begin{cases} x \equiv a_1 \pmod{m_1}, \\ \dots\dots\dots\dots\dots \\ x \equiv a_k \pmod{m_k}, \end{cases} \tag{10.25}$$

где $(m_i, m_j) = 1$ при $i \neq j$. Формулировка является конструктивной, т. е. представляет собой метод решения (10.25). Однако в ряде случаев более эффективным является алгоритм Гарнера, см. [25, § 4.3.2; 124; 180, гл. 14]. По сравнению с классической китайской теоремой об остатках он не требует приведения по модулю $M = m_1 \dots m_k$.

Алгоритм Гарнера.

На входе алгоритма заданы числа $a_1, \dots, a_k, m_1, \dots, m_k$, $(m_i, m_j) = 1$ при $i \neq j$, и число $M = m_1 \dots m_k$. На выходе получается решение x системы (10.25), $0 \leqslant x < M$.

1 шаг. Для $i = 2, \dots, k$ выполнить пункты 1) и 2):

1) $c_i := 1$;

2) для $j = 1, \dots, i-1$ выполнить: $u := m_j^{-1} \pmod{m_i}$ (нахождение обратного элемента можно делать с помощью обобщенного бинарного алгоритма, см. Приложение), $c_i := uc_i \pmod{m_i}$.

2 шаг. $u := a_1$, $x := u$.

3 шаг. Для $i = 2, \dots, k$ вычислить $u := (a_i - x)c_i \pmod{m_i}$ — наименьший неотрицательный вычет по модулю m_i,

$$x := x + u \prod_{j=1}^{i-1} m_j.$$

Полученное значение x является искомым решением.

Конец алгоритма.

Покажем, что алгоритм Гарнера выдает верный ответ, и что $0 \leqslant x < M$. Мы считаем, что $0 \leqslant a_i \leqslant m_i - 1$, $i = 1, \ldots, k$. Тогда по построению

$$0 \leqslant x \leqslant m_1 - 1 + \sum_{i=2}^{k} (m_i - 1) \prod_{j=1}^{i-1} m_j = m_1 \ldots m_k - 1 = M - 1.$$

Далее, при $i \geqslant 2$ $c_i \equiv (m_1 \ldots m_{i-1})^{-1} \pmod{m_i}$. Очевидно, что $x \equiv a_1 \pmod{m_1}$. Обозначим через x_i значение x в алгоритме, которое получается после выполнения 3 шага для значения переменной i $(2 \leqslant i \leqslant k)$; $x_1 = a_1$. Тогда

$$x_i = x_{i-1} + ((a_i - x_{i-1}) c_i \pmod{m_i}) \prod_{j=1}^{i-1} m_j,$$

$$x_i \pmod{m_i} \equiv x_{i-1} (a_i - x_{i-1}) \cdot \left(c_i \prod_{j=1}^{i-1} m_j \right) \pmod{m_i} \equiv a_i \pmod{m_i}.$$

Поскольку итоговое значение x в алгоритме по построению удовлетворяет сравнениям $x \equiv x_i \pmod{m_i}$, $i = 1, \ldots, k$, алгоритм Гарнера работает верно.

Теперь опишем вкратце модулярное умножение и возведение в степень по Монтгомери, см. [101, гл. 9; 180, гл. 14; 191]. Зафиксируем натуральное число $N > 1$ и натуральное число R, $R > N$, $(R, N) = 1$. Число R выбирается так, чтобы вычисления по модулю R были достаточно легкими (например, R — размер машинного слова или степень двойки). Пусть заданы R', $N' \in \mathbb{Z}$, такие, что

$$0 < R' < N,\ RR' - NN' = 1$$

(R' и N' можно найти с помощью обобщенного алгоритма Евклида, см. Приложение). Для целого числа T, $0 \leqslant T < RN$, можно быстро найти наименьший неотрицательный вычет в классе вычетов $TR' \pmod{N}$ с помощью следующего алгоритма (фактически здесь $R' \equiv R^{-1} \pmod{N}$).

Алгоритм REDC (приведение по Монтгомери).

1 шаг. $m := T \pmod{R}$ — наименьший неотрицательный вычет.

2 шаг. $m := mN' \pmod{R}$ — наименьший неотрицательный вычет.

3 шаг. $t := \dfrac{T + mN}{R}$.

4 шаг. Если $t \geqslant N$, то выдать $t - N$, иначе выдать t.

Конец алгоритма.

Покажем, что алгоритм работает верно. Поскольку значение m, получающееся на 2 шаге, удовлетворяет сравнению

$$mN \equiv TN'N \equiv -T \pmod{R},$$

то значение t, определяемое на 3 шаге, будет целым числом. Далее, $tR \equiv T + mN \equiv T \pmod{N}$, т. е. $t \equiv TR^{-1} \pmod{N}$. Наконец, $0 \leqslant T + mN < RN + RN$, откуда $0 \leqslant t < 2N$. Это завершает обоснование корректности алгоритма REDC.

Заметим, что поскольку мы считаем вычисления по модулю R (в том числе, деление на R) быстрыми, то сложность выполнения алгоритма REDC в основном заключается в двух умножениях целых чисел, по модулю не превосходящих R.

Аналогично можно описать приведение по Монтгомери для целых чисел многократной точности, умножение по Монтгомери (т. е. для $x, y \in \mathbb{Z}$ и $N \in \mathbb{N}$ мы находим $xyR^{-1} \pmod{N}$) и возведение в степень по Монтгомери (т. е. для x, N, $e \in \mathbb{N}$ вычисляется $x^e \pmod{N}$), см. [101, гл. 9; 180, гл. 14; 191]. Возведение в степень по Монтгомери оказывается эффективным и рекомендуется для практического использования, см. также [69].

Еще один алгоритм для эффективного модулярного возведения в степень был предложен в работе [132]. Авторы этой работы утверждают, что их алгоритм эффективнее алгоритма Монтгомери.

Для вычисления $x \pmod{m}$ при заданных x и m эффективен алгоритм Баррета, см. [69; 180, гл. 14]. В [101, гл. 9] описаны быстрые алгоритмы для вычисления по модулям специального вида, например, по модулю $N = 2^q + c$, $c \in \mathbb{Z}$, $q \in \mathbb{N}$, или по модулям Прота, имеющим вид $N = k \cdot 2^q + c$.

Ряд методов возведения в степень элементов конечных групп и способов записи и представления показателей описан в [180, § 14.6, 14.7].

Теперь опишем еще один алгоритм Монтгомери, который при заданном $N \in \mathbb{N}$ и заданных $a_1, \ldots, a_k \in \mathbb{Z}$ находит $b_1 \ldots, b_k \in \mathbb{Z}$ такие, что $a_i b_i \equiv 1 \pmod{N}$, $i = 1, \ldots, k$. Этот алгоритм описан в [89, гл. 10]; он может применяться в алгоритме факторизации целых чисел с помощью эллиптических кривых, см. гл. 4. С его помощью можно работать одновременно с несколькими кривыми.

Алгоритм.

1 шаг. Присвоить $c_1 := a_1$ и для $i = 2, \ldots, k$

$$c_i := c_{i-1} \cdot a_i \pmod{N}.$$

2 шаг. С помощью обобщенного алгоритма Евклида или одной из его модификаций (см. Приложение) найти u, d, v такие, что $uc_k + vN = d = \text{НОД}(c_k, N)$.

3 шаг. Для $i = k, k-1, \ldots, 2$ сделать следующее:

1) $b_i := uc_{i-1} \pmod{N}$;
2) выдать b_i;
3) $u := ua_i \pmod{N}$.

Полученное значение u — искомая величина.

Конец алгоритма.

Обоснуем правильность работы алгоритма в предположении, что $(a_i, N) = 1$, $i = 1, \ldots, k$, т. е. на 2 шаге алгоритма $d = 1$. Очевидно, что $c_i = a_1 \ldots a_i$, $i = 1, \ldots, k$. Поэтому после 2 шага $u \equiv \equiv (a_1 \ldots a_k)^{-1} \pmod{N}$. Тогда на 3 шаге при $i = k$ $b_k \equiv a_k^{-1} \pmod{N}$, и затем $u = (a_1 \ldots a_{k-1})^{-1} \pmod{N}$. Теперь очевидно, что и для всех i, $1 \leqslant i \leqslant k$, $b_i \equiv a_i^{-1} \pmod{N}$.

Ряд алгоритмов для вычисления в конечных полях и кольцах вычетов был предложен Ву, см. [287; 288; 289; 286]. В частности, в работе [287] предложено развитие метода Монтгомери для умножения и возведения в степень в полях $GF(2^m)$, а в работе [289] рассматривается приведение по Монтгомери для модулей специального вида, например, $N = 2^n - 2^m - 1$, $0 < m < \dfrac{n+1}{2}$.

Глава 11. Решение систем линейных уравнений над конечными полями

§ 11.1. Введение

В этой главе мы рассмотрим некоторые алгоритмы решения систем линейных уравнений над конечными полями. Как мы видели в гл. 3 и 5, такие системы возникают в алгоритмах факторизации и дискретного логарифмирования, использующих факторные базы. В алгоритмах факторизации это разреженные системы линейных уравнений над полем $\mathbb{Z}/2\mathbb{Z}$. В алгоритмах дискретного логарифмирования по простому модулю p это системы линейных уравнений над кольцом вычетов $\mathbb{Z}/(p-1)\mathbb{Z}$, однако их решение сводится к решению систем линейных уравнений над конечным простым полем. Действительно, пусть

$$\sum_{j=1}^{N} a_{ij}x_j \equiv b_i \pmod{p-1}, \quad i = 1, \ldots, M, \tag{11.1}$$

— система уравнений относительно неизвестных $x_1, \ldots, x_N$, которую мы хотим решить. Если $p-1 = \prod_{k=1}^{t} q_k^{\alpha_k}$ есть разложение $p-1$ на простые множители, то по китайской теореме об остатках решение системы (11.1) сводится к решению систем

$$\sum_{j=1}^{N} a_{ij}x_j \equiv b_j \pmod{q_k^{\alpha_k}}, \quad j = 1, \ldots, M, \tag{11.2}$$

для $k = 1, \ldots, t$, т. е. к нахождению $x_j \pmod{q_k^{\alpha_k}}$. Представим для фиксированного k неизвестные значения $x_j \pmod{q_k^{\alpha_k}}$ в виде

$$x_j \equiv x_{j0} + x_{j1}q_k + \ldots + x_{j,\alpha_k-1}q_k^{\alpha_k-1} \pmod{q_k^{\alpha_k}}, \tag{11.3}$$

где $0 \leqslant x_{jl} \leqslant q_k - 1$, $l = 0, 1, \ldots, \alpha_k - 1$. Редуцируя систему (11.2)

к модулю q_k, мы получим систему линейных уравнений

$$\sum_{j=1}^{N} a_{ij}x_{j0} \equiv b_j \pmod{q_k}, \quad j = 1, \ldots, M, \tag{11.4}$$

над конечным простым полем $\mathbb{Z}/q_k\mathbb{Z}$. Если мы найдем все x_{j0}, $j = 1, \ldots, N$, то, подставляя x_j в виде (11.3) с известными x_{i0} в систему (11.2), редуцируя ее к модулю q_k^2 и затем поделив на q_k, мы получим систему линейных уравнений над полем $\mathbb{Z}/q_k\mathbb{Z}$ относительно неизвестных x_{j1}, $j = 1, \ldots, N$, и так далее. В конечном счете, найдя значение $x_j \pmod{q_k^{\alpha_k}}$ для всех k, мы найдем $x_j \pmod{p-1}$ по китайской теореме об остатках.

Далее в этой главе мы рассмотрим методы решения систем линейных уравнений, наиболее часто используемые в современных алгоритмах факторизации и дискретного логарифмирования. Кроме того, в § 11.2 мы опишем методы решения систем линейных уравнений в целых числах.

§ 11.2. Решение систем линейных уравнений в целых числах

В этом параграфе мы рассмотрим два алгоритма решения систем линейных уравнений над кольцом целых чисел. По сути эти алгоритмы являются обобщениями алгоритма Евклида.

Сначала рассмотрим случай, когда система состоит из одного уравнения

$$a_1x_1 + \ldots + a_nx_n = d, \tag{11.5}$$

где $a_1, \ldots, a_n, d \in \mathbb{Z}$. Мы хотим описать алгоритм нахождения всех решений (11.5) в целых числах $x_1, \ldots, x_n$. Для этого составим матрицу

$$A = \begin{pmatrix} a_1 & \ldots & a_n \\ 1 & \ldots & 0 \\ \ldots & \ldots & \ldots \\ 0 & \ldots & 1 \end{pmatrix}$$

размера $(n+1) \times n$, у которой в первой строке стоят коэффициенты $a_1, \ldots, a_n$, а под ними единичная матрица. За один проход алгоритма мы выполняем следующие действия.

1) Выбираем в первой строке матрицы A наименьший по абсолютной величине ненулевой элемент a_i.

2) Выбираем номер $j \neq i$ такой, что $a_j \neq 0$.

3) Делим с остатком: $a_j = qa_i + r$, $0 \leqslant r < |a_i|$.

4) Вычитаем из j-го столбца матрицы A i-й столбец, умноженный на q.

В результате этих действий получится новая матрица A, у которой на месте элемента a_j появится либо 0 (если $r = 0$), либо элемент, который будет меньше модуля самого маленького по абсолютной величине ненулевого элемента первой строки у предыдущей матрицы A.

Очевидно, что после нескольких проходов алгоритма (т. е. нескольких выполнений действий 1—4), исходная матрица A превратится в матрицу

$$\begin{pmatrix} 0 & \dots & 0 & \lambda & 0 & \dots & 0 \\ c_{11} & \dots & c_{1,s-1} & c_{1s} & c_{1,s+1} & \dots & c_{1n} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ c_{n1} & \dots & c_{n,s-1} & c_{ns} & c_{n,s+1} & \dots & c_{nn} \end{pmatrix} = \begin{pmatrix} 0 & \dots & \lambda & \dots & 0 \\ & & C & & \end{pmatrix}, \qquad (11.6)$$

где λ, $c_{ij} \in \mathbb{Z}$, $\lambda \neq 0$. Если $\lambda \nmid d$, то система (11.5) не имеет решений в целых числах. Если же $\lambda \mid d$, то общее решение системы (11.5) в целых числах имеет вид

$$\begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} = t\boldsymbol{c}_1 + \dots + t_{s-1}\boldsymbol{c}_{s-1} + \frac{d}{\lambda}\boldsymbol{c}_s + t_{s+1}\boldsymbol{c}_{s+1} + \dots + t_n\boldsymbol{c}_n, \qquad (11.7)$$

где $t_1, \dots, t_{s-1}, t_{s+1}, \dots, t_n$ пробегают все целые числа, а векторы $\boldsymbol{c}_1, \dots, \boldsymbol{c}_n$ обозначают столбцы матрицы C из (11.6).

Докажем, что формула (11.7) действительно дает все решения системы (11.5) в целых числах. Очевидно, что в результате одного прохода алгоритма (т. е. выполнения действий 1—4) матрица A переходит в матрицу AD_{ij}, где у матрицы D_{ij} размера $n \times n$ на диагонали стоят единицы, элемент в i-й строке и j-м столбце равен $-q$, а все остальные элементы равны 0. Матрица D_{ij} является целочисленной с определителем $\det D_{ij} = 1$; значит, обратная матрица D_{ij}^{-1} также целочисленная. Запишем уравнение (11.5) в векторном виде:

$$(a_1, \dots, a_n) \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} = b.$$

Тогда

$$(a_1, \dots, a_n) D_{ij} D_{ij}^{-1} \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} = b.$$

Рассмотрим новые переменные

$$\begin{pmatrix} y_1 \\ \dots \\ y_n \end{pmatrix} = D_{ij}^{-1} \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix}.$$

Относительно этих переменных получим уравнение $a'_1 y_1 + \dots + a'_n y_n = b$, где $(a'_1, \dots, a'_n) = (a_1, \dots, a_n) D_{ij}$. Значения $y_1, \dots, y_n$ являются целочисленными тогда и только тогда, когда $x_1, \dots, x_n \in \mathbb{Z}$.

Если мы выполнили k проходов алгоритма для значений индексов $i_1, j_1, i_2, j_2, \dots, i_k, j_k$, то пришли к системе уравнений относительно неизвестных $z_1, \dots, z_n$,

$$\begin{pmatrix} z_1 \\ \dots \\ z_n \end{pmatrix} = D_{i_k j_k}^{-1} \dots D_{i_1 j_1}^{-1} \begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix}.$$

Если при этом первая строка матрицы A стала равной

$$(0, \dots, 0, \lambda, 0, \dots, 0) = (a_1, \dots, a_n) D_{i_1 j_1} \dots D_{i_k j_k},$$

то система уравнений примет вид

$$\lambda z_1 = b. \tag{11.8}$$

Если $\lambda \nmid b$, то решений нет, а если $\lambda \mid b$, то общее решение (11.8) имеет вид $(z_1, \dots, z_n) = \left(t_1, \dots, t_{s-1}, \dfrac{b}{\lambda}, t_{s+1}, \dots, t_n\right)$, где $t_1, \dots, t_n$ пробегают все целые числа. Тогда

$$\begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} = D_{i_1, j_1} \dots D_{i_k, j_k} \begin{pmatrix} t_1 \\ \dots \\ t_{s-1} \\ b/\lambda \\ t_{s+1} \\ \dots \\ t_n \end{pmatrix}.$$

Осталось лишь заметить, что матрица C из (11.6) равна $D_{i_1, j_1} \dots D_{i-k, j_k}$, поскольку с остальными строками исходной матрицы A мы делали те же действия, что и с первой строкой, т. е. единичная $(n \times n)$-подматрица матрицы A последовательно умножалась на $D_{i_1 j_1}, D_{i_2 j_2}, \dots, D_{i_k j_k}$.

Теперь рассмотрим произвольную систему линейных уравнений

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n = b_1, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + \ldots + a_{mn}x_n = b_m, \end{cases} \tag{11.9}$$

где a_{ij}, $b_l \in \mathbb{Z}$. Опишем алгоритм нахождения всех решений (11.9) в целых числах. Составим матрицу A размера $m \times n$ и матрицу B размера $(m+n) \times (n+1)$, в которой под матрицей A стоит единичная матрица размера $n \times n$:

$$A = \begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \ldots\ldots & \ldots & \ldots\ldots \\ a_{m1} & \ldots & a_{mn} \end{pmatrix}, \quad B = \begin{pmatrix} & & & -b_1 \\ & A & & \ldots \\ & & & -b_m \\ 1 & \ldots & 0 & 0 \\ \ldots\ldots & \ldots & \ldots\ldots & \ldots\ldots \\ 0 & \ldots & 1 & 0 \end{pmatrix}.$$

С первыми n столбцами матрицы B можно производить следующие действия:

1) переставлять их;

2) вычитать из одного столбца другой, умноженный на целое число. Также можно переставлять какие-либо из первых m строк матрицы B.

С помощью этих действий мы преобразуем матрицу B к частично-треугольному виду

$$B_1 = \begin{pmatrix} u_{11} & 0 & \ldots & 0 & 0 & \ldots & 0 & -b'_1 \\ u_{21} & u_{22} & \ldots & 0 & 0 & \ldots & 0 & -b'_2 \\ \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots & \ldots\ldots \\ u_{k1} & u_{k2} & \ldots & u_{kk} & 0 & \ldots & 0 & -b'_k \\ u_{k+1,1} & u_{k+1,2} & \ldots & u_{k+1,k} & 0 & \ldots & 0 & -b'_{k+1} \\ \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots & \ldots\ldots \\ u_{m1} & u_{m2} & \ldots & u_{mk} & 0 & \ldots & 0 & -b'_m \\ c_{11} & c_{12} & \ldots & c_{1k} & c_{1,k+1} & \ldots & c_{1n} & 0 \\ \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots & \ldots\ldots & \ldots & \ldots\ldots & \ldots\ldots \\ c_{n1} & c_{n2} & \ldots & c_{nk} & c_{n,k+1} & \ldots & c_{nn} & 0 \end{pmatrix} \tag{11.10}$$

при некотором $k \geqslant 1$, $k \leqslant \min(m, n)$, и $u_{11} \neq 0, \ldots, u_{kk} \neq 0$. (Элементы $b'_1, \ldots, b'_m$ — перестановка элементов $b_1, \ldots, b_m$, которая получилась, когда мы переставляли первые m строк матрицы B.) Для того чтобы привести B к виду (11.10), надо сначала в первой строке B (без

последнего элемента $-b_1$) обнулить все элементы, кроме одного (как в предыдущем алгоритме), и потом столбец, содержащий этот ненулевой элемент, поставить на первое место. Затем так же поступить со второй строкой получившейся матрицы (не затрагивая ее первый столбец) и т. д.

После того как матрица B_1 вида (11.10) построена, мы преобразуем ее к виду

$$B_2 = \begin{pmatrix} u_{11} & 0 & \dots & 0 & 0 & \dots & 0 \\ u_{21} & u_{22} & \dots & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ u_{k1} & u_{k2} & \dots & u_{kk} & 0 & \dots & 0 \\ u_{k+1,1} & u_{k+1,2} & \dots & u_{k+1,k} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ u_{m1} & u_{m2} & \dots & u_{mm} & 0 & \dots & 0 \\ c_{11} & c_{12} & \dots & c_{1k} & c_{1,k+1} & \dots & f_{1n} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & c_{nk} & c_{n,k+1} & \dots & f_{nn} \end{pmatrix}. \qquad (11.11)$$

Для этого мы умножаем первые k столбцов матрицы B_1 на целые числа и прибавляем их к последнему столбцу. Точнее, сначала мы прибавляем первый столбец B_2, умноженный на число b'_1/u_{11}, которое должно быть целым, к последнему столбцу и получаем ноль в первом элементе $(n+1)$-го столбца. Затем с помощью второго столбца и элемента u_{22} мы получаем ноль во втором элементе $(n+1)$-го столбца и так далее.

Если такой переход от матрицы (11.10) к матрице (11.11) указанным способом невозможен, то система (11.9) не имеет решений в целых числах. Если же мы нашли матрицу B_2 из (11.11), то общее решение системы (11.9) в целых числах имеет вид

$$\begin{pmatrix} x_1 \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} f_1 \\ \dots \\ f_n \end{pmatrix} + t_1 \begin{pmatrix} c_{1,k+1} \\ \dots \\ c_{n,k+1} \end{pmatrix} + \dots + t_{n-k} \begin{pmatrix} c_{1n} \\ \dots \\ c_{nn} \end{pmatrix},$$

где $t_1, \dots, t_{n-k}$ пробегают все множество целых чисел.

Обоснование данного алгоритма решения системы (11.9) аналогично обоснованию алгоритма решения (11.5); мы предлагаем читателю придумать это обоснование самостоятельно.

§ 11.3. Гауссово и структурированное гауссово исключение

Пусть $\mathbb{K}$ — произвольное поле. Рассмотрим систему линейных уравнений

$$\sum_{j=1}^{n} a_{ij}x_j = b_i, \quad i = 1, \ldots, m \tag{11.12}$$

над полем $\mathbb{K}$. Общеизвестным методом решения этой системы является гауссово исключение, т. е. приведение системы к треугольному виду. Сложность этого метода составляет $O(mn^2)$ арифметических операций в поле $\mathbb{K}$. С помощью приведения к треугольному виду можно также вычислять определитель квадратной матрицы размера $n \times n$ за $O(n^3)$ арифметических операций, находить обратную матрицу, находить ядро и образ линейного отображения конечномерных линейных пространств. Мы не приводим здесь соответствующие алгоритмы ввиду их очевидности и общеизвестности. При желании эти алгоритмы можно найти в [6; 19; 89, гл. 2].

Применительно к решению систем линейных уравнений, возникающих в алгоритмах дискретного логарифмирования, был разработан метод *структурированного гауссова исключения*, см. [210; 151; 150]. Этот метод применяется в случае, когда матрица системы уравнений является разреженной. Суть его заключается в следующем. Столбцы матрицы коэффициентов системы уравнений разбиваются на «легкие» (содержащие мало ненулевых элементов) и «тяжелые». Неизвестные x_j, соответствующие «легким» столбцам, мы также называем «легкими», а соответствующие «тяжелым» — «тяжелыми» неизвестными. Строки матрицы (соответствующие линейным уравнениям) мы также разделяем на 2 категории: на «легкую» и «тяжелую» часть матрицы. Первоначально в «легкую» часть матрицы мы относим все те уравнения, которые содержат какую-либо из переменных, встречающихся только в одном из уравнений системы (это означает, что такая переменная с помощью данного уравнения однозначно выражается через остальные). Столбец, соответствующий такой переменной, содержит лишь один ненулевой элемент; мы объявляем его «легким».

Далее, возьмем какой-либо из столбцов, не являющихся «легкими», но такой, что в нем все же мало ненулевых элементов. При этом столбец выбирается так, чтобы одна из строк, содержащих его ненулевые элементы, содержала нули почти во всех других столбцах. Тогда, вычитая строки матрицы, умноженные на подходящие числа, мы

можем сделать так, чтобы в этом столбце все элементы, за исключением одного, соответствующего выбранной строке, стали равными нулю. После этого данный столбец станет «легким», и соответствующая его ненулевому элементу строка также объявляется «легкой». Другие же столбцы в большинстве своем не утяжелятся ввиду выбора строки, которую мы вычитали из других. Переносим эту строку в «легкую» часть матрицы. Так мы получим еще несколько «легких» столбцов. Однако, поскольку мы вычитали строки матрицы, в ней появится некоторое количество новых ненулевых элементов. То есть матрица станет менее разреженной, чем была ранее. Тогда мы некоторые строки переносим в «тяжелую» часть матрицы. Эти строки мы выбираем так, чтобы они имели наибольшее количество ненулевых элементов в наиболее разреженных столбцах. В результате перенесения строк в «тяжелую» часть эти столбцы станут еще более разреженными в «легкой» части матрицы. Тогда мы сделаем некоторые из них «легкими» (описанным выше способом, т. е. вычитанием строк), и так далее.

Через некоторое время в нашей системе линейных уравнений выделится подматрица, матрица которой уже не является разреженной, и переменные этой подсистемы являются «тяжелыми» (т. е. «тяжелыми» являются соответствующие им столбцы). Тогда мы решим эту систему каким-нибудь способом (например, гауссовым исключением или с помощью алгоритма Ланцоша). Найдя значения «тяжелых» переменных, мы затем с помощью «легкой» части матрицы найдем и «легкие» переменные.

Такова приблизительная схема метода структурированного гауссова исключения. Эффективность ее реализации во многом зависит от квалификации программиста.

Структурированное гауссово исключение применялось не только в алгоритмах дискретного логарифмирования, но и при факторизации методом решета числового поля, см. [162; 159].

§ 11.4. Алгоритм Ланцоша

Алгоритм Ланцоша впервые был описан в работе [153]. Впоследствии этот алгоритм был модифицирован таким образом, чтобы применяться для решения систем линейных уравнений, возникающих в алгоритмах факторизации и дискретного логарифмирования, см. [150; 104; 276]. В работе [193] описан блочный алгоритм Ланцоша, который является эффективным при решении линейных систем над полем $GF(2)$.

В работе [104] описана модификация алгоритма Ланцоша, допускающая распараллеливание.

Пусть $\mathbb{K}$ — поле, A — матрица размера $n \times n$ над $\mathbb{K}$, $\boldsymbol{b}$ — n-мерный вектор, $\boldsymbol{b} \neq \boldsymbol{0}$. Мы хотим решить систему линейных уравнений

$$A\boldsymbol{x} = \boldsymbol{b}. \tag{11.13}$$

Предположим дополнительно, что матрица A — симметричная, а вектор $\boldsymbol{b}$ удовлетворяет условию $\boldsymbol{b}^T A\boldsymbol{b} \neq 0$. Рассмотрим последовательность Крылова S, состоящую из n-мерных векторов $\boldsymbol{s}_0$, $\boldsymbol{s}_1$, $\boldsymbol{s}_2$, …, где

$$\boldsymbol{s}_0 = \boldsymbol{b}, \quad \boldsymbol{s}_i = A^i\boldsymbol{b} = A\boldsymbol{s}_{i-1}, \quad i = 1, 2, \ldots \tag{11.14}$$

Лемма 11.1. *Пусть $m \in \mathbb{N}$, $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{m-1}$ — линейно независимы над $\mathbb{K}$, а $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_m$ — линейно зависимы. Тогда любой элемент из множества S линейно выражается через $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{m-1}$.*

Доказательство. Поскольку $\boldsymbol{s}_m = \sum\limits_{j=0}^{m-1} \alpha_j \boldsymbol{s}_j$, где $\alpha_j \in \mathbb{K}$, то в силу (11.14) получаем, что $\boldsymbol{s}_{m+1} = A\boldsymbol{s}_m = \sum\limits_{j=0}^{m-2} \alpha_j \boldsymbol{s}_{j+1} + \alpha_{m-1} \sum\limits_{j=0}^{m-1} \alpha_j \boldsymbol{s}_j$. Дальнейшие рассуждения очевидны. □

Везде далее параметр m — из условия леммы 11.1.

Рассмотрим стандартное скалярное произведение $(\boldsymbol{x}, \boldsymbol{y}) = \sum\limits_{i=1}^{n} x_i y_i$, где $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{y} = (y_1, \ldots, y_n)$. Для векторов $\boldsymbol{x}$ и $\boldsymbol{y}$ положим

$$(\boldsymbol{x}, \boldsymbol{y})_A = (\boldsymbol{x}, A\boldsymbol{y}) = \boldsymbol{x}^T A\boldsymbol{y} = (A\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{K};$$

выражение $(\boldsymbol{x}, \boldsymbol{y})_A$ мы также будем называть скалярным произведением векторов.

Попробуем ортогонализировать последовательность Крылова относительно скалярного произведения $(\boldsymbol{x}, \boldsymbol{y})_A$ методом Грама—Шмидта, т. е. найти последовательность векторов $\boldsymbol{w}_0$, $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, …, удовлетворяющую условию

$$(\boldsymbol{w}_i, \boldsymbol{w}_j)_A = 0 \quad \text{при} \quad i \neq j \tag{11.15}$$

и такую, что

$$\begin{cases} \boldsymbol{w}_0 = \boldsymbol{s}_0, \\ \boldsymbol{w}_1 = \boldsymbol{s}_1 - \alpha_{10}\boldsymbol{w}_0, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \boldsymbol{w}_i = \boldsymbol{s}_i - \alpha_{i0}\boldsymbol{w}_0 - \ldots - \alpha_{i,i-1}\boldsymbol{w}_{i-1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{cases} \tag{11.16}$$

где $\alpha_{ij} \in \mathbb{K}$. При этом, если $i > j$ и $(\boldsymbol{w}_i, \boldsymbol{w}_j)_A \neq 0$, то коэффициент α_{ij} мы будем определять с помощью равенства

$$(\boldsymbol{w}_i, \boldsymbol{w}_j)_A = (\boldsymbol{s}_i, \boldsymbol{w}_j)_A - \alpha_{ij}(\boldsymbol{w}_j, \boldsymbol{w}_j)_A = 0.$$

Процесс ортогонализации можно продолжать до тех пор, пока мы не дойдем до номера k такого, что

$$(\boldsymbol{w}_k, \boldsymbol{w}_k)_A = 0. \tag{11.17}$$

Лемма 11.2. *В процессе ортогонализации при некотором $k \leqslant m$ будет выполнено равенство* (11.17).

Доказательство. В силу (11.16) векторы $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_i$ выражаются через векторы $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_i$ с помощью треугольной матрицы, на диагонали которой стоят единицы. Поэтому векторы $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_i$ линейно независимы тогда и только тогда, когда $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_i$ линейно независимы. Предположим, что мы нашли векторы $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_m$, т. е. при $k < m$ равенство (11.17) не выполняется.Следовательно, векторы $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_{m-1}$ линейно независимы, а $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_m$ зависимы, и поэтому $\boldsymbol{w}_m = \sum\limits_{i=0}^{m-1} c_i \boldsymbol{w}_i$, $c_i \in \mathbb{K}$. Но отсюда в силу ортогональности $0 = (\boldsymbol{w}_m, \boldsymbol{w}_i) = c_i$, $i = 0, \ldots, m-1$, т. е. $\boldsymbol{w}_m = \boldsymbol{0}$. □

Лемма 11.3. *Пусть процесс ортогонализации закончился вектором $\boldsymbol{w}_k$. Если $\boldsymbol{w}_k = \boldsymbol{0}$, то $k = m$; если же $\boldsymbol{w}_k \neq \boldsymbol{0}$, то $k < m$.*

Доказательство. Из доказательства леммы 11.2 следует, что при $k < m$ вектор $\boldsymbol{w}_k$ не может быть равен нулю, так как $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_k$ линейно независимы. Поэтому, если $\boldsymbol{w}_k = 0$, то $k = m$. Если же $k = m$, то $\boldsymbol{w}_m = \boldsymbol{0}$ (см. доказательство леммы 11.2). □

Вернемся к системе уравнений (11.13). Предположим, что ее решение имеет при некотором $r \geqslant 0$ вид

$$\boldsymbol{x} = \sum_{i=0}^{r} c_i \boldsymbol{w}_i, \qquad c_i \in \mathbb{K}. \tag{11.18}$$

Так как $A\boldsymbol{x} = \sum\limits_{i=0}^{r} c_i A\boldsymbol{w}_i = \boldsymbol{b}$, то

$$(\boldsymbol{w}_i, \boldsymbol{b}) = \left(\boldsymbol{w}_i, \sum_{j=0}^{r} c_j A\boldsymbol{w}_j\right) = c_i(\boldsymbol{w}_i, \boldsymbol{w}_i)_A.$$

При $(\boldsymbol{w}_i, \boldsymbol{w}_i)_A \neq 0$ мы получим, что $c_i = (\boldsymbol{w}_i, \boldsymbol{b})/(\boldsymbol{w}_i, \boldsymbol{w}_i)_A$. Теперь мы

можем предположить, что решение (11.13) имеет вид

$$\boldsymbol{x} = \sum_{i=0}^{r} \frac{(\boldsymbol{w}_i,\, \boldsymbol{b})}{(\boldsymbol{w}_i,\, \boldsymbol{w}_i)} \boldsymbol{w}_i. \tag{11.19}$$

Лемма 11.4. *Пусть процесс ортогонализации завершился при* $k = m$, *при этом были построены линейно независимые векторы* $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_{m-1}$, *а* $\boldsymbol{w}_m = \boldsymbol{0}$. *Тогда формула* (11.19) *при* $r = m-1$ *дает решение* (11.13).

Доказательство. Для вектора $\boldsymbol{x}$ из (11.19) выполняются равенства

$$(\boldsymbol{w}_i,\, A\boldsymbol{x}) = \sum_{j=0}^{m-1} \frac{(\boldsymbol{w}_j,\, \boldsymbol{b})}{(\boldsymbol{w}_j,\, \boldsymbol{w}_j)_A} (\boldsymbol{w}_i,\, A\boldsymbol{w}_j) = (\boldsymbol{w}_i,\, \boldsymbol{b}), \quad i = 0, \ldots, m-1.$$

Поэтому вектор $A\boldsymbol{x} - \boldsymbol{b}$ ортогонален (относительно стандартного скалярного произведения) векторам $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_{m-1}$, и, значит, ортогонален линейному пространству $\Omega = L_{\text{об}}(S)$. Так как $A\Omega \subseteq \Omega$, то для любого $\boldsymbol{w} \in \Omega$ выполнено равенство

$$(A\boldsymbol{w},\, A\boldsymbol{x} - \boldsymbol{b}) = 0. \tag{11.20}$$

Поскольку $\boldsymbol{b} = \boldsymbol{s}_0 \in \Omega$ и $A\boldsymbol{x} \in \Omega$, то вектор $A\boldsymbol{x} - \boldsymbol{b} = \sum\limits_{i=0}^{m-1} \lambda_i \boldsymbol{w}_i$, $\lambda_i \in \mathbb{K}$. Из (11.20) и симметричности матрицы A следует, что

$$0 = \left(A\boldsymbol{w}_j,\, \sum_{i=0}^{m-1} \lambda_i \boldsymbol{w}_i\right) = \sum_{i=0}^{m-1} \lambda_i (\boldsymbol{w}_j,\, \boldsymbol{w}_i)_A = \lambda_j (\boldsymbol{w}_j,\, \boldsymbol{w}_j)_A, \quad j = 0, \ldots, m-1.$$

Поэтому $\lambda_0 = \ldots = \lambda_{m-1} = 0$, $A\boldsymbol{x} = \boldsymbol{b}$, что и требовалось доказать. □

Замечание 11.5. Если в процессе ортогонализации последовательности Крылова мы получим нулевой вектор, то найдем решение системы уравнений (11.13).

Алгоритм Ланцоша основан на некоторой модификации процесса ортогонализации (11.16). Рассмотрим последовательность векторов $\boldsymbol{w}'_0, \boldsymbol{w}'_1, \ldots$, удовлетворяющую равенствам

$$\begin{cases} \boldsymbol{w}'_0 = \boldsymbol{s}_0 = \boldsymbol{b}, \\ \boldsymbol{w}'_1 = A\boldsymbol{w}'_0 - \alpha_{10}\boldsymbol{w}'_0, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \boldsymbol{w}'_i = A\boldsymbol{w}'_{i-1} - \sum_{j=0}^{i-1} \alpha_{ij}\boldsymbol{w}'_j, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{cases} \tag{11.21}$$

(где $\alpha'_{ij} \in \mathbb{K}$) и являющуюся ортогональной, т. е.

$$(\boldsymbol{w}'_i,\, \boldsymbol{w}'_j)_A = 0 \quad \text{при} \quad i \neq j. \tag{11.22}$$

Коэффициенты α'_{ij} находятся по формуле

$$\alpha'_{ij} = \frac{(A\boldsymbol{w}'_{i-1},\, \boldsymbol{w}'_j)_A}{(\boldsymbol{w}'_j,\, \boldsymbol{w}'_j)_A}, \tag{11.23}$$

при условии, что $(\boldsymbol{w}'_j,\, \boldsymbol{w}'_j)_A \neq 0$.

Лемма 11.6. *Если для номера i векторы $\boldsymbol{w}_i$ и $\boldsymbol{w}'_i$ определены, то $\boldsymbol{w}'_i = \boldsymbol{w}_i$.*

Доказательство. Для $i = 0$ выполняется равенство $\boldsymbol{w}'_0 = \boldsymbol{s}_0 = \boldsymbol{w}_0$. Предположим, что $\boldsymbol{w}'_i = \boldsymbol{w}_i$ при всех $i \leqslant t$, и пусть $\boldsymbol{w}_{t+1}$ и $\boldsymbol{w}'_{t+1}$ определены. Тогда $\boldsymbol{w}'_{t+1} = A\boldsymbol{w}_t - \sum\limits_{j=0}^{t} \alpha'_{t+1,j}\boldsymbol{w}_j$. Поскольку векторы $\boldsymbol{w}_i$ удовлетворяют (11.16) и $\boldsymbol{s}_i = A^i\boldsymbol{b}$, то $A\boldsymbol{w}_t = A^{t+1}\boldsymbol{b} - A\left(\sum\limits_{l=0}^{t-1} \alpha_{tl}\boldsymbol{w}_l\right) = A^{t+1}\boldsymbol{b} - \sum\limits_{l=0}^{t} \alpha''_{tl}\boldsymbol{w}_l$ при некоторых $\alpha''_{tl} \in \mathbb{K}$. Следовательно, $\boldsymbol{w}'_{t+1} = A^{t+1}\boldsymbol{b} - \sum\limits_{l=0}^{t} \alpha'''_{t+1,l}\boldsymbol{w}_l$. Для вектора $\boldsymbol{w}_{t+1}$ из (11.16) выполняется аналогичное равенство $\boldsymbol{w}_{t+1} = A^{t+1}\boldsymbol{b} - \sum\limits_{l=0}^{t} \alpha_{t+1,l}\boldsymbol{w}_l$. Тогда вектор $\boldsymbol{w}'_{t+1} - \boldsymbol{w}_{t+1} = \sum\limits_{l=0}^{t} \beta_l\boldsymbol{w}_l$ (где $\beta_l \in \mathbb{K}$) по предположению индукции ортогонален векторам $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_t$. Значит, $0 = \left(\sum\limits_{l=0}^{t} \beta_l\boldsymbol{w}_l,\, \boldsymbol{w}_j\right)_A = \beta_j(\boldsymbol{w}_j,\, \boldsymbol{w}_j)_A = 0$ при $j = 0, \ldots, t$. Так как $(\boldsymbol{w}_j,\, \boldsymbol{w}_j)_A \neq 0$, то $\beta_j = 0$, $j = 0, \ldots, l$, и $\boldsymbol{w}'_{t+1} = \boldsymbol{w}_{t+1}$. $\square$

Из (11.21), (11.22) и леммы 11.6 получим, что если векторы $\boldsymbol{w}_i$ и $\boldsymbol{w}'_i$ определены, то формула (11.23) может быть записана в виде

$$\alpha'_{ij} = \frac{(A\boldsymbol{w}_{i-1},\, \boldsymbol{w}_j)_A}{(\boldsymbol{w}_j,\, \boldsymbol{w}_j)_A}, \qquad 0 \leqslant j \leqslant i-1. \tag{11.24}$$

Следовательно, при $j \leqslant i-1$ выполнено равенство

$$\alpha'_{ij} = \frac{(\boldsymbol{w}_{i-1},\, A\boldsymbol{w}_j)_A}{(\boldsymbol{w}_j,\, \boldsymbol{w}_j)_A} = \frac{(\boldsymbol{w}_{i-1},\, \boldsymbol{w}_{j+1} + \sum_{l=0}^{j} \alpha'_{j+1,l}\boldsymbol{w}_l)_A}{(\boldsymbol{w}_j,\, \boldsymbol{w}_j)_A}.$$

Отсюда при $j+1<i-1$ следует, что $\alpha'_{ij}=0$. Это означает, что формула (11.21) имеет вид

$$\boldsymbol{w}_i = A\boldsymbol{w}_{i-1} - \frac{(\boldsymbol{w}_{i-1},\, A\boldsymbol{w}_{i-1})_A}{(\boldsymbol{w}_{i-1},\, \boldsymbol{w}_{i-1})_A}\boldsymbol{w}_{i-1} - \frac{(\boldsymbol{w}_{i-1},\, A\boldsymbol{w}_{i-2})_A)}{(\boldsymbol{w}_{i-2},\, A\boldsymbol{w}_{i-2})_A}\boldsymbol{w}_{i-2}, \qquad (11.25)$$

т. е. содержит лишь три слагаемых в правой части. Поэтому вычисления по ней проводятся быстрее, чем непосредственно по формулам (11.16).

Алгоритм Ланцоша работает следующим образом. Мы вычисляем последовательность векторов $\boldsymbol{w}_0=\boldsymbol{b}$, $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, …, пользуясь формулами (11.21), (11.23), (11.25) (в предположении, что выполнены условия леммы 11.6). Если на некотором шаге будет построен вектор $\boldsymbol{w}_i\neq\boldsymbol{0}$ такой, что $(\boldsymbol{w}_i,\, \boldsymbol{w}_i)_A=0$, то мы не сможем найти решение (11.13) этим методом. Если же будет построен вектор $\boldsymbol{w}_m=\boldsymbol{0}$, то решение $\boldsymbol{x}$ мы находим по формуле (11.19) при $r=m-1$. При этом следует сделать проверку, поскольку мы предполагаем выполнение некоторых условий.

В алгоритмах факторизации и дискретного логарифмирования матрица A системы (11.13) не является симметричной (и даже квадратной). В этом случае предлагается случайным образом выбрать диагональную матрицу D и рассмотреть систему уравнений

$$A^TD^2A\boldsymbol{x}=A^TD^2\boldsymbol{b}. \qquad (11.26)$$

Ее матрица $A^TD^2A=(DA)^TDA$ будет квадратной и симметричной, и к системе (11.26) мы затем применяем алгоритм Ланцоша. Если алгоритм закончится неудачей либо найденный вектор $\boldsymbol{x}$ не будет решением (11.13), то следует выбрать другую матрицу D.

Однородная система $A\boldsymbol{x}=\boldsymbol{0}$ может быть преобразована в неоднородную в предположении, что в ее решении $(x_1,\, \ldots,\, x_n)$ последняя координата $x_n\neq 0$. Положив тогда $x_n=1$, мы сможем перенести столбец соответствующих x_n коэффициентов в правую часть и затем применить алгоритм Ланцоша.

Дальнейшее обсуждение деталей реализации алгоритма Ланцоша можно найти в прекрасной диссертации [104].

Заметим, что последние рекорды в области дискретного логарифмирования в простых полях были достигнуты с помощью алгоритма Ланцоша (см. выше гл. 5).

§ 11.5. Алгоритм Видемана

В этом параграфе мы опишем алгоритм Видемана [281] для решения системы линейных уравнений

$$A\boldsymbol{x} = \boldsymbol{b}, \quad \boldsymbol{b} \neq \boldsymbol{0}, \tag{11.27}$$

над конечным полем $\mathbb{K} = GF(q)$. Матрица A предполагается разреженной; w обозначает число ненулевых элементов A. Мы будем считать, что A квадратная размера $n \times n$ и невырожденная; рассмотрение других случаев, а также алгоритм вычисления определителя матрицы можно найти в работе [281]. Полученные Видеманом оценки сложности алгоритмов являются наилучшими из известных, и их применение позволяет улучшать оценки сложности в других алгоритмах, использующих методы решения линейных систем. В частности, описываемый ниже алгоритм 2 является детерминированным и требует $O(n(w + n\log n \log\log n))$ операций поля. Заметим, что первое время после появления работы [281] алгоритмы Видемана считались непрактичными и полезными лишь для получения наилучших оценок сложности. Однако в последние годы алгоритмы были реализованы на компьютере и использовались, например, для разложения многочленов на множители над конечными полями, см. [139]. Появились различные модификации, в частности, блочный алгоритм Видемана, см. [96; 105; 138; 140; 177].

Вернемся к решению системы (11.27). Матрица A задает невырожденное линейное отображение (которое мы также обозначаем A) на пространстве $\mathbb{K}^n$. Рассмотрим пространство S, порожденное множеством векторов $\{A^i\boldsymbol{b} \mid i = 0, 1, 2, \ldots\}$, и положим $A_S = A|_S$ — линейное отображение S на S. Обозначим $f(z) \in \mathbb{K}[z]$ — минимальный многочлен A_S, т. е. ненулевой многочлен наименьшей степени, такой, что $f(A_S)$ — нулевое отображение S. Мы считаем $f(z)$ нормализованным таким образом, что его свободный член равен 1. Заметим, что если $g(z) \in \mathbb{K}[z]$, то $g(A_S)$ — нулевое отображение S тогда и только тогда, когда $g(A)\boldsymbol{b} = \boldsymbol{0}$. Кроме того, $f(z)$ делит многочлен $\det(zI_n - A)$, и поэтому $\deg f(z) \leqslant n$.

Обозначим $d = \deg f(z)$, $f(z) = \sum\limits_{i=0}^{d} f[i]z^i$, где $f[i] \in \mathbb{K}$ — коэффициенты $f(z)$. Если мы сможем найти $f(z)$, то найдем и решение системы (11.27): так как $f(A)\boldsymbol{b} = \boldsymbol{0}$ и $f[0] = 1$, то

$$\boldsymbol{x} = -\sum_{i=1}^{d} f[i]A^{i-1}\boldsymbol{b}. \tag{11.28}$$

Пусть $\boldsymbol{u}$ — какой-либо фиксированный вектор из $\mathbb{K}^n$; $(\cdot, \cdot)$ — стандартное билинейное отображение $\mathbb{K}^n$ в $\mathbb{K}$,

$$((v_1, \ldots, v_n), (w_1, \ldots, w_n)) = \sum_{i=1}^{n} v_i w_i.$$

Поскольку $f(A)\boldsymbol{b} = \boldsymbol{0}$, то последовательность

$$(\boldsymbol{u}, A^i\boldsymbol{b}), \quad i = 0, 1, 2, \ldots \tag{11.29}$$

удовлетворяет линейному рекуррентному соотношению, характеристический многочлен которого равен $f(z)$. Пусть $f_{\boldsymbol{u}}(z)$ — минимальный многочлен для последовательности (11.29) (т. е. характеристический многочлен самого короткого рекуррентного соотношения). Тогда $f_{\boldsymbol{u}} \mid f(z)$. Действительно, если мы разделим с остатком

$$f(z) = q(z)f_{\boldsymbol{u}}(z) + r(z), \quad \deg r(z) < \deg f_{\boldsymbol{u}}(z),$$

то из равенств

$$0 = (\boldsymbol{u}, f(A)\boldsymbol{b}) = (\boldsymbol{u}, q(A)f_{\boldsymbol{u}}(A)\boldsymbol{b}) + (\boldsymbol{u}, r(A)\boldsymbol{b}),$$
$$(\boldsymbol{u}, f_{\boldsymbol{u}}(A)A^j\boldsymbol{b}) = 0, \quad j = 0, 1, 2, \ldots,$$

и минимальности $f_{\boldsymbol{u}}(z)$ будет следовать, что $r(z) = 0$. Поскольку свободный член $f(z)$ равен 1, мы можем считать, что и свободный член $f_{\boldsymbol{u}}$ равен 1.

Минимальный многочлен $f_{\boldsymbol{u}}(z)$ для последовательности (11.29) может быть вычислен с помощью алгоритма Берлекэмпа—Месси (см. [7; 31, гл. 2; 175]) по первым $2n$ ее членам. Поэтому возможен следующий метод решения (11.27): выбрать случайный вектор $\boldsymbol{u}$, построить $f_{\boldsymbol{u}}(z)$ и в предположении, что $f(z) = f_{\boldsymbol{u}}(z)$, найти $\boldsymbol{x}$ по формуле (11.28). Согласно [281], мы этим путем с достаточно высокой вероятностью найдем решение системы (11.27).

Рассмотрим другой подход к решению (11.27). Пусть $\boldsymbol{b}_0 = \boldsymbol{b}$, $f_1(z) = f_{\boldsymbol{u}_1}(z)$ для некоторого вектора $\boldsymbol{u}_1$. Если вектор $\boldsymbol{b}_1 = f_1(A)\boldsymbol{b}_0$ равен 0, то мы находим $\boldsymbol{x}$ по формуле (11.28) (так как тогда $f_1(z) = f(z)$). Если же $\boldsymbol{b}_1 \neq \boldsymbol{0}$, то повторяем процедуру, т. е. выбираем случайный вектор $\boldsymbol{u}_2$ и строим минимальный многочлен $f_2(z) = f_{\boldsymbol{u}_2}(z)$ для последовательности $(\boldsymbol{u}_2, A^i\boldsymbol{b}_1)$. Если $\boldsymbol{b}_2 = f_2(A)\boldsymbol{b}_1 = \boldsymbol{0}$, то (как будет показано ниже) $f(z) = f_1(z)f_2(z)$ и мы находим решение $\boldsymbol{x}$ по формуле (11.28), иначе выбираем $\boldsymbol{u}_3$ и т. д.

Покажем, что если мы сделали k итераций, то $f_1(z)\dots f_k(z)$ делит $f(z)$. Действительно, выше было показано, что $f_1(z)\mid f(z)$. Далее, если мы предположим, что $f_1(z)\dots f_{k-1}(z)$ делит $f(z)$, то поскольку $f_k(z)$ — минимальный многочлен для последовательности $\{(\boldsymbol{u}_k, A^i\boldsymbol{b}_{k-1})\}_i = \{(\boldsymbol{u}_k, f_{k-1}(A)\dots f_1(A)A^j\boldsymbol{b})\}_j$, а многочлен $\dfrac{f(z)}{f_1(z)\dots f_{k-1}(z)}$ ее аннулирует, то $f_k(z)\,\Big|\,\dfrac{f(z)}{f_1(z)\dots f_{k-1}(z)}$, что и требовалось доказать.

Теперь очевидно, что если $\boldsymbol{b}_k = f_k(A)\dots f_1(A)\boldsymbol{b} = \boldsymbol{0}$, то $f(x) = f_1(x)\dots$ $\dots f_k(x)$. Следовательно, как только будет построен нулевой вектор $\boldsymbol{b}_k = f_k(A)\boldsymbol{b}_{k-1}$, мы сможем найти решение (11.27) по формуле (11.28).

Формализуем сказанное в виде алгоритма 1. Для произвольного многочлена $g(z)\in\mathbb{K}[z]$ мы будем обозначать $\hat{g}(z) = \dfrac{g(z)-g(0)}{z}$.

Алгоритм 1.

1 шаг. Присвоить $\boldsymbol{b}_0 := \boldsymbol{b}$, $k := 0$, $\boldsymbol{y}_0 := \boldsymbol{0}$, $d_0 := 0$.

2 шаг. Если $\boldsymbol{b}_k = \boldsymbol{0}$, то решение (11.27) равно $\boldsymbol{x} = -\boldsymbol{y}_k$, и алгоритм завершает работу.

3 шаг. Выбрать случайный вектор $\boldsymbol{u}_{k+1}\in\mathbb{K}^n$, $\boldsymbol{u}_{k+1}\neq\boldsymbol{0}$.

4 шаг. Вычислить первые $2(n-d_k)$ членов последовательности $\{(\boldsymbol{u}_{k+1},\, A^i\boldsymbol{b}_k)\}_{i=0,1,2,\dots}$

5 шаг. С помощью алгоритма Берлекэмпа—Месси вычислить минимальный многочлен $f_{k+1}(z)$ последовательности шага 4, нормализованный так, чтобы его свободный член равнялся единице.

6 шаг. Присвоить

$$\begin{aligned}\boldsymbol{y}_{k+1} &:= \boldsymbol{y}_k + \hat{f}_{k+1}(A)\boldsymbol{b}_k,\\ \boldsymbol{b}_{k+1} &:= \boldsymbol{b}_0 + A\boldsymbol{y}_{k+1},\\ d_{k+1} &:= d_k + \deg\, f_{k+1}(z).\end{aligned}$$

7 шаг. Присвоить $k := k+1$ и вернуться на 2 шаг.

Конец алгоритма.

Приведем обоснование корректности алгоритма. Заметим, что $\hat{f}(z) = \dfrac{f(z)-f(0)}{z}$ соответствует правой части формулы (11.28) (без знака минус). При $k=0$ мы выбираем $\boldsymbol{u}_1$, рассматриваем $2n$ членов последовательности $\{(\boldsymbol{u}_1,\, A^i\boldsymbol{b})\}_{i=0,1,\dots}$ и находим $f_1(x)$ по алгоритму Берлекэмпа—Месси. Тогда $\boldsymbol{y}_1 = \hat{f}_1(A)\boldsymbol{b}$, $\boldsymbol{b}_1 = \boldsymbol{b}_0 + A\boldsymbol{y}_1 = \boldsymbol{b} + A\dfrac{f_1(A)-1}{A}\boldsymbol{b} = f_1(A)\boldsymbol{b}$, $d_1 = \deg f_1(z)$. Далее рассуждаем по индукции. Пусть после k проходов

алгоритма выполнены равенства

$$\boldsymbol{y}_k = \frac{f_k(A)\dots f_1(A)-1}{A}\,\boldsymbol{b}, \tag{11.30}$$

$$\boldsymbol{b}_k = f_k(A)\dots f_1(A)\,\boldsymbol{b}. \tag{11.31}$$

Тогда после $k+1$ прохода

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k + \hat{f}_{k+1}(A)\boldsymbol{b}_k =$$
$$= \frac{f_k(A)\dots f_1(A)-1}{A}\boldsymbol{b} + \frac{f_{k+1}(A)-1}{A} f_k(A)\dots f_1(A)\boldsymbol{b} = \frac{f_{k+1}(A)\dots f_1(A)-1}{A}\boldsymbol{b},$$

$$\boldsymbol{b}_{k+1} = \boldsymbol{b} + A\frac{f_{k+1}(A)\dots f_1(A)-1}{A}\boldsymbol{b} = f_{k+1}(A)\dots f_1(A)\boldsymbol{b}.$$

То есть формулы (11.30) и (11.31) сохраняются. Корректность алгоритма 1 следует теперь из сказанного выше перед его описанием.

Теперь опишем детерминированный алгоритм.

Алгоритм 2.

1 шаг. Вычислить $A^i\boldsymbol{b}$, $i = 0, 1, \dots, 2n-1$.

2 шаг. Присвоить $k := 0$, $g_0(z) := 1$.

3 шаг. Присвоить $\boldsymbol{u}_{k+1} := (0, \dots, 0, 1, 0, \dots, 0)$ (единица стоит на $(k+1)$-м месте).

4 шаг. Используя результаты 1-го шага, вычислить последовательность

$$(\boldsymbol{u}_{k+1},\ A^i\boldsymbol{b}), \quad i = 0, 1, \dots, 2n-1.$$

5 шаг. Вычислить последовательность

$$(\boldsymbol{u}_{k+1},\ g_k(A)A^i\boldsymbol{b}), \quad i = 0, \dots, 2n-1-\deg\ g_k(z)$$

(здесь можно использовать дискретное преобразование Фурье, см. [281]).

6 шаг. Найти (с помощью алгоритма Берлекэмпа—Месси) минимальный многочлен $f_{k+1}(z)$ для последовательности, полученной на 5-м шаге (свободный член $f_{k+1}(z)$ равен 1).

7 шаг. Присвоить $g_{k+1}(z) := f_{k+1}(z)g_k(z)$.

8 шаг. Присвоить $k := k+1$. Если $\deg g_k(z) < n$ и $k < n$, то идти на 3 шаг.

9 шаг. Для многочлена $f(z) = g_k(z)$ с помощью найденных на 1 шаге значений $A^i\boldsymbol{b}$ найти решение $\boldsymbol{x}$ системы (11.27) по формуле (11.28).

Конец алгоритма.

Обоснуем корректность алгоритма 2. Заметим, что фактически алгоритм 2 работает так же, как алгоритм 1, только векторы $\boldsymbol{u}_k$ выбираются

не случайно, а идет перебор единичных векторов (0, …, 0, 1, 0, …, 0). Легко видеть, что $g_k(z) = f_k(z) \dots f_1(z)$, где $f_k(z)$ — минимальный многочлен для последовательности

$$(\boldsymbol{u}_k,\ f_{k-1}(A) \dots f_1(A) A^i \boldsymbol{b}), \quad i = 0, \dots, 2n - 1 - \deg(f_{k-1}(z) \dots f_1(z)).$$

Пусть алгоритм закончил работу при некотором значении параметра k. Рассмотрим сначала случай $k < n$ и $\deg g_k(z) = n$. Так как $\deg f(z) \leqslant n$ и $g_k(z) \mid f(z)$, то $g_k(z) = f(z)$. Следовательно, на 9 шаге действительно будет найдено решение (11.27).

Теперь предположим, что $k = n$. Поскольку мы перебрали все единичные векторы $\boldsymbol{u}_1, \dots, \boldsymbol{u}_n$, то вектор $g_n(A)\boldsymbol{b}$ ортогонален $\boldsymbol{u}_1, \dots, \boldsymbol{u}_n$ (очевидно по построению). Следовательно, $g_n(A)\boldsymbol{b} = \boldsymbol{0}$. Так как $g_n(z) \mid f(z)$ и $f(z)$ — минимальный, то $g_k(z) = f(z)$. Поэтому и в данном случае алгоритм 2 работает корректно.

На этом мы закончим описание алгоритмов Видемана. Получение оценок сложности для них можно найти в работе [281].

§ 11.6. Другие методы. Заключение

Обзор методов решения систем линейных уравнений применительно к алгоритмам факторизации и дискретного логарифмирования содержится в работе [150]. Отметим, в частности метод сопряженных градиентов (см. [97; 210]), работающий, согласно [151], примерно столько же времени, сколько и алгоритм Ланцоша. Оба эти алгоритма используют сравнительно небольшое пространство памяти.

Другие методы решения линейных систем основаны на алгоритмах быстрого умножения матриц, таких, как алгоритмы Штрассена и Копперсмита—Винограда (см. обзор [2]). В работе Коновальцева [26] предложен алгоритм решения квадратной системы линейных уравнений с n неизвестными над полем $GF(q)$ за $O(n^3/\log_q n)$ арифметических операций. Бриллхарт [78] предложил алгоритм решения линейных систем над $GF(2)$, являющийся, по сути, некоторой вариацией гауссова исключения; см. также [212].

Обсуждение возможностей применения различных методов решения систем линейных уравнений в алгоритмах факторизации и дискретного логарифмирования можно найти в недавних обзорах [74] и [209]. В последние годы предпочтение все же отдается методу Ланцоша (возможно, в сочетании со структурированным гауссовым исключением для получения более плотной матрицы подсистемы).

Приложение. Сведения из теории чисел

В данном Приложении мы приводим основные определения и факты из элементарной теории чисел, наиболее часто используемые в книге. Доказательства теорем можно найти в [18]. Также мы описываем ряд усовершенствований алгоритма Евклида; их обоснование см. в [25, гл. 4.5.2; 60, гл. 4; 89, гл. 1].

Начнем с *алгоритма Евклида*. Пусть $a \in \mathbb{Z}$, $b \in \mathbb{N}$, и мы хотим найти $d = \text{НОД}(a, b)$. Разделим a на b с остатком: $a = q_0 b + r_0$, $0 \leqslant r_0 < b$. Если $r_0 > 0$, то разделим b на r_0: $b = q_1 r_0 + r_1$, $0 \leqslant r_1 < r_0$, и т. д. Получим последовательность равенств с убывающими остатками r_j:

$$r_{j-2} = q_j r_{j-1} + r_j, \quad j = 0, 1, 2 \ldots, \tag{1}$$

где $r_{-2} = a$, $r_{-1} = b$. Если r_k — последний ненулевой остаток, то $r_{k-1} = q_{k+1} r_k$. Тогда справедливо равенство $d = \text{НОД}(a, b) = r_k$. Вычисление последовательности (1) и есть алгоритм Евклида.

Теперь найдем представление $d = r_k$ в виде $d = au + bv$, где $u, v \in \mathbb{Z}$. Мы можем последовательно выражать r_k из (1), поднимаясь снизу вверх: сначала через r_{k-1} и r_{k-2} из предпоследнего равенства в (1); затем, подставив $r_{k-1} = r_{k-3} - q_{k-1} r_{k-2}$, мы выразим r_k через r_{k-2} и r_{k-3}, и так далее. В конце мы выразим r_k через a и b.

Однако более удобен *обобщенный алгоритм Евклида*. В нем мы работаем с системой равенств

$$\begin{cases} u_1 a + u_2 b = u_3, \\ v_1 a + v_2 b = v_3, \\ t_1 a + t_2 b = t_3. \end{cases} \tag{2}$$

На 1 шаге полагаем

$$(u_1, u_2, u_3) := (1, 0, a), \quad (v_1, v_2, v_3) := (0, 1, b), \quad (t_1, t_2, t_3) := (0, 0, 0).$$

Затем при очередном проходе алгоритма мы проверяем условие $v_3 = 0$. Если оно выполнено, то искомые значения d, u, v равны u_3, u_1, u_2 соответственно. Если же $v_3 \neq 0$, то производим деление с остатком:

$u_3 = qv_3 + r$. Затем присваиваем:

$$(t_1,\, t_2,\, t_3) := (v_1,\, v_2,\, v_3),$$
$$(v_1,\, v_2,\, v_3) := (u_1 - qv_1,\, u_2 - qv_2,\, u_3 - qv_3),$$
$$(u_1,\, u_2,\, u_3) := (t_1,\, t_2,\, t_3).$$

Поскольку значение $v_3 \in \mathbb{Z}_{\geqslant 0}$ монотонно убывает, то алгоритм заканчивает работу и выдает верный ответ. Заметим, что с переменными u_2, v_2, t_2 можно не работать, поскольку их значения однозначно восстанавливаются по u_1, u_3, v_1, v_3, t_1, t_3 соответственно.

Теорема 1 (Ламе). *Параметр k в формуле (1) удовлетворяет неравенству*

$$k \leqslant 5([\log_{10} b] + 1).$$

Следствие 2. *Для выполнения алгоритма Евклида и обобщенного алгоритма Евклида требуется $O(\log b)$ арифметических операций.*

Замечание 3. В алгоритме Евклида можно использовать деление с выбором наименьшего по абсолютной величине остатка: при каждом делении какого-либо целого числа a' на какое-либо натуральное число b' мы находим q', $r' \in \mathbb{Z}$ такие, что $a' = q'b' + r'$, $0 \leqslant |r'| \leqslant b'/2$. Это ускоряет работу алгоритма Евклида.

Теперь перейдем к сравнениям.

Определение 4. Пусть a, $b \in \mathbb{Z}$, $m \in \mathbb{N}$. Мы говорим, что *a сравнимо с b по модулю m* (и обозначаем это $a \equiv b \pmod m$), если $m \mid a - b$, или, эквивалентно, a и b имеют одинаковые остатки от деления на m; или, эквивалентно, $a = b + mt$, где $t \in \mathbb{Z}$.

Если m фиксировано, то $\mathbb{Z}$ разбивается на классы $\overline{0}$, $\overline{1}$, …, $\overline{m-1}$ чисел, имеющих остатки 0, 1, …, $m-1$ от деления на m. Эти классы вычетов образуют $\mathbb{Z}/m\mathbb{Z}$ — кольцо вычетов по модулю m. Для $a \in \mathbb{Z}$ символом $a \pmod m$ мы обозначаем класс вычетов по модулю m, содержащий a. Обратимые по умножению элементы $\mathbb{Z}/m\mathbb{Z}$ образуют мультипликативную группу $(\mathbb{Z}/m\mathbb{Z})^*$. Число $a \in \mathbb{Z}$ является элементом обратимого по умножению класса вычетов по модулю m тогда и только тогда, когда разрешимо уравнение

$$ax \equiv 1 \pmod m. \tag{3}$$

Уравнение (3) разрешимо тогда и только тогда, когда $(a,\, m) = 1$. Решение (3) можно найти с помощью обобщенного алгоритма Евклида: мы находим u, $v \in \mathbb{Z}$ такие, что $au + mv = 1$, и тогда $a \pmod m u \pmod m \equiv \equiv 1 \pmod m$, т. е. $u \equiv a^{-1} \pmod m$.

Полной системой вычетов по модулю m мы называем какой-либо набор целых чисел $a_1, \ldots, a_m$, попарно не сравнимых по модулю m. *Приведенной системой вычетов по модулю* m мы называем какой-либо набор целых чисел $b_1, \ldots, b_k$, где $k = |(\mathbb{Z}/m\mathbb{Z})^*|$, попарно не сравнимых по модулю m и взаимно простых с m.

Теперь определим *функцию Эйлера* $\varphi(m)$: если $m \in \mathbb{N}$, то $\varphi(m) = |(\mathbb{Z}/m\mathbb{Z})^*|$. Эквивалентно, $\varphi(m)$ равна количеству чисел в наборе $\{0, 1, \ldots, m-1\}$, взаимно простых с m. В частности, $\varphi(1) = 1$, $\varphi(p) = p - 1$, где p — простое число. Справедлива следующая формула для нахождения $\varphi(m)$: если $m = \prod_{j=1}^{k} p_j^{\alpha_j}$ — разложение m на простые множители, то $\varphi(m) = m \prod_{j=1}^{k} (1 - 1/p_j)$.

Теорема 5 (Эйлер). *Если* $(a, m) = 1$, *то*

$$a^{\varphi(m)} \equiv 1 \pmod{m}.$$

Теорема 6 (малая теорема Ферма). *Если* p *— простое число,* p *не делит* a, *то*

$$a^{p-1} \equiv 1 \pmod{p}.$$

Теорема 7 (китайская теорема об остатках). *Рассмотрим систему сравнений*

$$\begin{cases} x \equiv a_1 \pmod{m_1}, \\ \ldots\ldots\ldots\ldots\ldots \\ x \equiv a_k \pmod{m_k}, \end{cases} \tag{4}$$

где $(m_i, m_j) = 1$ *при всех* $i \neq j$. *Положим* $M = m_1 \ldots m_k$; $M_i = M/m_i$, $i = 1, \ldots, k$. *Найдем* (*с помощью обобщенного алгоритма Евклида*) *числа* $b_1, \ldots, b_k$ *такие, что* $M_i b_i \equiv 1 \pmod{m_i}$, $i = 1, \ldots, k$. *Тогда общее решение системы* (4) *в целых числах имеет вид*

$$x \equiv \sum_{i=1}^{k} M_i a_i b_i \pmod{M},$$

т. е. $x = \sum_{i=1}^{k} M_i a_i b_i + Mt$, *где* $t \in \mathbb{Z}$.

Теперь рассмотрим группу $(\mathbb{Z}/m\mathbb{Z})^*$. Она является циклической тогда и только тогда, когда $m = 1, 2, 4, p^k, 2p^k$, где p обозначает нечетное простое число, $k \in \mathbb{N}$. Число $g \in \mathbb{Z}$ такое, что класс вычетов $g \pmod{m}$ является образующим $(\mathbb{Z}/m\mathbb{Z})^*$ (в случае, когда эта группа — циклическая), называется *первообразным корнем* по модулю m.

Теорема 8. *Пусть p — простое число, $p > 2$.*

1) *Пусть $p - 1 = \prod_{j=1}^{k} q_j^{\alpha_j}$ есть разложение $p - 1$ на простые множители. Число g является первообразным корнем по модулю p тогда и только тогда, когда*

$$(g, p) = 1, \quad g^{(p-1)/q_j} \not\equiv 1 \pmod{p}, \quad j = 1, \ldots, k.$$

2) *Пусть g — первообразный корень по модулю p, число g_1 равно тому из чисел g или $g + p$, которое удовлетворяет сравнению $x^{p-1} \not\equiv 1 \pmod{p^2}$. Тогда g_1 является первообразным корнем по модулю p^2.*

3) *Если число g_2 является первообразным корнем по модулю p^2, то g_2 является первообразным корнем по модулю p^k для всех $k \in \mathbb{N}$.*

Теорема 9. *Справедливо равенство $(\mathbb{Z}/4\mathbb{Z})^* = \langle 3 \pmod 4 \rangle_2$. Далее, при $k \geqslant 3$ имеем*

$$(\mathbb{Z}/2^k\mathbb{Z})^* = \langle -1 \pmod{2^k} \rangle_2 \times \langle 5 \pmod{2^k} \rangle_{2^{k-2}}$$

есть разложение $(\mathbb{Z}/2^k\mathbb{Z})^*$ в прямое произведение циклических групп.

Если $m = 2^{\alpha_0} p_1^{\alpha_1} \ldots p_k^{\alpha_k}$ есть разложение m на простые множители, то можно указать полную систему образующих группы $(\mathbb{Z}/m\mathbb{Z})^*$, зная первообразные корни по модулям $p_i^{\alpha_i}$, $i = 1, \ldots, k$ (см. [18, гл.6]).

Если g — первообразный корень по модулю m, $a \in \mathbb{Z}$, $(a, m) = 1$, то решение уравнения $g^x \equiv a \pmod{m}$ называется индексом элемента a по основанию g или дискретным логарифмом a по основанию g и обозначается $\operatorname{ind}_g a$ или $\log_g a$. Эта величина определена по модулю $\varphi(m)$, т. е., $\operatorname{ind}_g a \in \mathbb{Z}/\varphi(m)\mathbb{Z}$.

Пусть $m \in \mathbb{Z}$. *Числовым характером по модулю m* называется отображение $\chi : \mathbb{Z} \to \mathbb{C}$ со следующими свойствами:

1) $\chi(n) = 0$ тогда и только тогда, когда $(m, n) > 1$;

2) $\chi(n + m) = \chi(n)$ для всех $n \in \mathbb{Z}$;

3) $\chi(n_1 n_2) = \chi(n_1)\chi(n_2)$ для всех $n_1, n_2 \in \mathbb{Z}$.

По сути χ является характером группы $(\mathbb{Z}/m\mathbb{Z})^*$, естественным образом продолженным на все множество $\mathbb{Z}$.

Теперь рассмотрим символы Лежандра и Якоби.

Пусть p — простое число, $p > 2$. Пусть $a \in \mathbb{Z}$, $(a, p) = 1$. Число a называется *квадратичным вычетом* по модулю p, если уравнение

$$x^2 \equiv a \pmod{p} \tag{5}$$

разрешимо; иначе оно называется *квадратичным невычетом*. *Символ Лежандра* $\left(\frac{a}{p}\right)$ (где $a \in \mathbb{Z}$) равен $+1$, если a — квадратичный вычет по модулю p; $\left(\frac{a}{p}\right) = -1$, если a — квадратичный невычет; $\left(\frac{a}{p}\right) = 0$, если $a \equiv 0 \pmod{p}$.

Символ Лежандра обладает следующими свойствами:

1) *критерий Эйлера*:

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p};$$

2) если $a \equiv b \pmod{p}$, то $\left(\frac{a}{p}\right) = \left(\frac{b}{p}\right)$;

3) $\left(\frac{1}{p}\right) = 1$, $\left(\frac{-1}{p}\right) = (-1)^{(p-1)/2}$;

4) $\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$;

5) $\left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}$;

6) *квадратичный закон взаимности Гаусса*: если p, q — нечетные простые числа, $p \neq q$, то

$$\left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}\left(\frac{p}{q}\right).$$

Если $m \in \mathbb{N}$, m — нечетное составное число и $m = \prod\limits_{j=1}^{k} p_j^{\alpha_j}$ есть разложение m на простые множители, то для $a \in \mathbb{Z}$ *символ Якоби* $\left(\frac{a}{m}\right)$ определяется равенством

$$\left(\frac{a}{m}\right) = \prod_{j=1}^{k}\left(\frac{a}{p_j}\right)^{\alpha_j}.$$

Свойства символа Якоби:

1) если $a \equiv b \pmod{m}$, то $\left(\frac{a}{m}\right) = \left(\frac{b}{m}\right)$;

2) $\left(\frac{1}{m}\right) = 1$, $\left(\frac{-1}{m}\right) = (-1)^{(m-1)/2}$;

3) $\left(\frac{ab}{m}\right) = \left(\frac{a}{m}\right)\left(\frac{b}{m}\right)$;

4) $\left(\frac{2}{m}\right) = (-1)^{(m^2-1)/8}$;

5) если m и n — натуральные взаимно простые нечетные числа, то

$$\left(\frac{m}{n}\right) = (-1)^{(m-1)(n-1)/4}\left(\frac{n}{m}\right).$$

Нетрудно написать алгоритмы для вычисления символов Лежандра и Якоби, основываясь на их свойствах. Эти алгоритмы приведены, например, в [89, гл. 1].

Пусть $m \in \mathbb{N}$, $f(x) \in \mathbb{Z}[x]$. Если мы хотим решить уравнение $f(x) \equiv 0 \pmod{m}$, то с помощью китайской теоремы об остатках мы можем свести нашу задачу к уравнению $f(x) \equiv 0 \pmod{p^\alpha}$, где p — простое число, $\alpha \in \mathbb{N}$ (это называется *редукцией*). Если мы знаем решение a уравнения $f(x) \equiv 0 \pmod{p^\alpha}$, то можем построить решение a_1 уравнения $f(x) \equiv 0 \pmod{p^{\alpha+1}}$ (это — *подъем решения*). О редукции и подъеме решения см. [18, гл. 4]. Наиболее известные методы решения уравнений $f(x) \equiv 0 \pmod{p}$ мы описали в гл. 6 нашей книги.

Теперь приведем определения и факты из теории *непрерывных* (или *цепных*) дробей, см., например, [29; 41]. Пусть $n \in \mathbb{Z}_{\geqslant 0}$, $a_0 \in \mathbb{Z}$, $a_1, \ldots, a_n \in \mathbb{N}$. *$n$-членной непрерывной дробью* $[a_0; a_1, \ldots, a_n]$ называется рациональное число

$$[a_0; a_1, \ldots, a_n] = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ldots + \cfrac{1}{a_n}}}}.$$

В частности, 0-членная непрерывная дробь $[a_0;\ ] = a_0$ — это просто целое число.

Теорема 10. *Любое рациональное число $\alpha \neq 1$ представимо в виде $\alpha = [a_0; a_1, \ldots, a_n]$ при некоторых $n \in \mathbb{Z}_{\geqslant 0}$, $a_0 \in \mathbb{Z}$, $a_1, \ldots, a_n \in \mathbb{N}$. Это представление единственно, если считать, что $a_n > 1$.*

Замечание 11. Число $\alpha = 1 = [1;\ ] = [0; 1]$ разложимо двумя способами, и в каждом разложении последний элемент равен 1. Если $\alpha \neq 1$, $\alpha = [a_0; a_1, \ldots, a_n]$, где $a_n \in \mathbb{N}$, $a_n > 1$, то α также представимо в виде $\alpha = [a_0; a_1, \ldots, a_n - 1, 1]$.

Подходящие дроби к $\alpha = [a_0; a_1, \ldots, a_n]$ имеют вид $p_k/q_k = [a_0; a_1, \ldots, a_k]$, $k = 0, 1, \ldots, n$, причем $p_k \in \mathbb{Z}$, $q_k \in \mathbb{N}$, $(p_k, q_k) = 1$. Для удобства положим $p_{-1} = 1$, $q_{-1} = 0$. Тогда справедливы соотношения

$$\begin{cases} p_0 = a_0, \quad q_0 = 1, \\ p_k = a_k p_{k-1} + p_{k-2}, \quad q_k = a_k q_{k-1} + q_{k-2} \quad \text{при} \quad k \geqslant 1, \end{cases}$$

которые позволяют быстро вычислять подходящие дроби.

Подходящие дроби к числу α обладают следующими свойствами:

1) $p_k q_{k-1} - p_{k-1} q_k = (-1)^{k-1}$, $k = 0, 1, \ldots$;

2) $p_k q_{k-2} - p_{k-2} q_k = (-1)^k a_k$, $k = 2, 3, \ldots$;

3) подходящие дроби с четными номерами возрастают, с нечетными — убывают;

4) любая подходящая дробь с четным номером меньше любой подходящей дроби с нечетным номером;

5) $1 = q_0 \leqslant q_1 < q_2 < \ldots < q_n$;

6) $\left|\alpha - \dfrac{p_k}{q_k}\right| \leqslant \dfrac{1}{q_{k+1} q_k} \leqslant \dfrac{1}{q_k^2}$;

7) $\left|\alpha - \dfrac{p_k}{q_k}\right| \geqslant \dfrac{a_{k+2}}{q_{k+2} q_k}$.

Если $a_0 \in \mathbb{Z}$, $a_1, a_2, \ldots \in \mathbb{N}$, то *бесконечной непрерывной дробью* $[a_0; a_1, a_2, \ldots]$ называется предел

$$\alpha = [a_0; a_1, a_2, \ldots] = \lim_{n \to \infty} [a_0; a_1, \ldots, a_n]. \tag{6}$$

Дроби $p_n/q_n = [a_0; a_1, \ldots, a_n]$ называются *подходящими дробями* к бесконечной непрерывной дроби. Они обладают теми же свойствами, что и подходящие дроби к конечной непрерывной дроби; в частности, легко видеть, что предел (6) существует.

Теорема 12. *Любое действительное иррациональное число α единственным образом представимо в виде бесконечной непрерывной дроби $\alpha = [a_0; a_1, a_2 \ldots]$.*

Замечание 13. Разложение числа $\alpha \in \mathbb{R}$ в непрерывную дробь можно получить с помощью следующего алгоритма. Пусть $\alpha = [\alpha] + \{\alpha\}$. Положим $a_0 = [\alpha]$, $r_1 = 1/\{\alpha\}$, $\alpha = a_0 + 1/r_1$. Затем $r_1 = [r_1] + \dfrac{1}{1/\{r_1\}} = a_1 + 1/r_2$ и так далее. После $(k+1)$-го вычисления целой части и переворачивания дробной части получим

$$\alpha = a_0 + \cfrac{1}{a_1 + \cfrac{1}{\ldots + \cfrac{1}{a_{k-1} + \cfrac{1}{r_k}}}}.$$

Если $\alpha = a/b \in \mathbb{Q}$, то этот процесс эквивалентен алгоритму Евклида для чисел a и b. Если же $\alpha \in \mathbb{R} \setminus \mathbb{Q}$, то этот процесс будет бесконечным.

Для некоторых иррациональных чисел разложение в непрерывную дробь периодично: дробь $\alpha = [a_0; a_1, \ldots, a_{k-1}, \overline{a_k, \ldots, a_{k+T}}]$ — *периодическая*, если $a_k, \ldots, a_{k+T}$ периодически повторяются. Например, $\sqrt{5} = [2; \overline{4}] = [2; 4, 4, 4, \ldots]$.

Теорема 14 (Эйлер—Лагранж). *Число* $\alpha \in \mathbb{R} \setminus \mathbb{Q}$ *раскладывается в периодическую непрерывную дробь тогда и только тогда, когда* α *является квадратичной иррациональностью, т. е. корнем уравнения* $ax^2 + bx + c = 0$, *где* $a \in \mathbb{N}$, $b, c \in \mathbb{Z}$.

Теперь мы опишем несколько модификаций алгоритма Евклида и обобщенного алгоритма Евклида. В этих алгоритмах мы стремимся к экономии операций деления, поскольку деление целых чисел многократной точности отнимает много времени. Деление же на 2 по сути не является делением, а всего лишь сдвигом двоичной записи числа в машинном представлении.

Бинарный алгоритм.

На входе алгоритма заданы числа $a, b \in \mathbb{N}$, $a \geqslant b$. На выходе $d = \text{НОД}(a, b)$.

1 шаг. Присвоить r остаток от деления a на b. Затем $a := b$, $b := r$.

2 шаг. Если $b = 0$, то выдать a и закончить работу. Иначе $k := 0$, и затем, пока a и b оба четные, выполнять:

$$\begin{cases} k := k + 1, \\ a := a/2, \\ b := b/2. \end{cases} \tag{7}$$

(После этого 2^k идет в НОД(a, b), оставшаяся часть НОД будет нечетной.)

3 шаг. Сейчас по крайней мере одно из чисел a и b нечетно. Если a четное, то делать присвоения $a := a/2$ до тех пор, пока a не станет нечетным. То же для b.

4 шаг. Сейчас a и b оба нечетны. Присвоить $t := (a - b)/2$. Если $t = 0$, то выдать $2^k a$ и закончить работу.

5 шаг. До тех пор, пока t четное, делать $t := t/2$.

6 шаг. Сейчас t нечетное, $t \neq 0$. Если $t > 0$, то $a := t$. Если $t < 0$, то $b := -t$. Идти на 4 шаг.

Конец алгоритма.

Замечание 15. Анализ сложности алгоритма см. в [25, гл. 4.5.2].

Алгоритм Лемера.

На входе $u, v \in \mathbb{N}$, $u \geqslant v$ (u, v — целые числа многократной точности). На выходе $d = \text{НОД}(u, v)$. Вспомогательные переменные: t, w — многократной точности, $\hat{u}, \hat{v}, A, B, C, D, T, q$ — однократной точности, p-значные. То есть длина записи числа, не превосходящая p, считается однократной точностью; запись числа — в системе счисления с основанием $b \in \mathbb{N}$.

1 шаг. (Начальная установка.)

1) Если v — однократной точности (в частности, $v=0$), то применяем алгоритм Евклида к u и v и находим d, после чего заканчиваем работу.

2) Если запись u в системе счисления с основанием b занимает k_1 порций по p разрядов, а запись v — k_2 порций по p разрядов, и $k_1 > k_2$, то выполняем одно деление многократной точности с остатком $u = tv + w$, и затем присваиваем $u := v$, $v := w$. (Эвристически после этого u и v будут примерно одинаковой величины.)

3) В $\hat{u}$ заносим число, записываемое p старшими цифрами u; то же для $\hat{v}$ и v. Затем

$$A := 1, \quad B := 0, \quad C := 0, \quad D := 1.$$

2 шаг. (Проверка окончания.) Сейчас $\hat{u} \neq 0$, $\hat{v} \neq 0$, $\hat{u} \geqslant \hat{v}$. Если $\hat{u} \neq 0$, но $\hat{u} + A = 0$ или $\hat{u} + B = 0$ (это возможно, поскольку данные переменные — однократной точности; например, если $\hat{u} = b^p - 1$, $A = 1$, то $\hat{u} + A = 0$), то перейти на 4 шаг. Аналогично, если $\hat{v} \neq 0$, но $\hat{v} + C = 0$ или $\hat{v} + D = 0$, то также перейти на 4 шаг. Иначе находим $q := \left[\dfrac{\hat{u} + A}{\hat{v} + C}\right]$. Если $q \neq \left[\dfrac{\hat{u} + B}{\hat{v} + D}\right]$, то перейти на 4 шаг.

3 шаг. Выполняем вычисления:

$$\begin{cases} T := A - qC, \\ A := C, \\ C := T, \end{cases} \qquad \begin{cases} T := B - qD, \\ B := D, \\ D := T, \end{cases} \qquad \begin{cases} T := \hat{u} - q\hat{v}, \\ \hat{u} := \hat{v}, \\ \hat{v} := T. \end{cases}$$

Если $\hat{v} = 0$, перейти на 4 шаг. Иначе перейти на 2 шаг.

4 шаг. Если $B = 0$, то произвести деление многократной точности с остатком u на v: $u = wv + t$, $0 \leqslant t \leqslant v - 1$. Затем $u := v$, $v := t$, и перейти на 1-й шаг. Если же $B \neq 0$, то вычислить с помощью арифметики многократной точности

$$t := Au + Bv, \quad w := Cu + Dv,$$

затем присвоить $u := t$, $v := w$. Перейти на 1 шаг.

Конец алгоритма.

Обобщенный бинарный алгоритм.

На входе алгоритма заданы $a, b \in \mathbb{N}$, $a \geqslant b$. На выходе $d = \text{НОД}(a, b) \in \mathbb{N}$ и числа $u, v \in \mathbb{Z}$ такие, что $au + bv = d$. Используются вспомогательные переменные многократной точности v_1, v_3, t_1, t_3 и две булевы переменные f_1, f_2.

1 шаг. (Одноразовое уменьшение размера.) Если $a < b$, то поменять местами a и b и присвоить $f_1 := 1$. Иначе $f_1 := 0$. Если $b = 0$, то выдать $(u, v, d) = (1, 0, a)$ при $f_1 = 0$, и $(u, v, d) = (0, 1, a)$ при $f_1 = 1$; закончить работу. Иначе поделить с остатком $a = bq + r$, $0 \leqslant r < b$, и присвоить $a := b$, $b := r$.

2 шаг. (Вычисление степеней двойки.) Если $b = 0$, то выдать $(u, v, d) = (0, 1, a)$ при $f_1 = 0$, $(u, v, d) = (1, 0, a)$ при $f_1 = 1$ и закончить работу. Иначе $k := 0$, и до тех пор, пока a и b оба четные, выполнять:

$$\begin{cases} k := k + 1, \\ a := a/2, \\ b := b/2. \end{cases}$$

3 шаг. (Инициализация.) Если b четно, то поменять местами a и b и присвоить $f_2 := 1$. Иначе $f_2 := 0$. Далее,

$$u := 1, \quad d := a, \quad v_1 := 0, \quad v_3 := b.$$

Если a нечетно, то $t_1 := 0$, $t_3 := -b$, и перейти на 5 шаг. Иначе $t_1 := (b + 1)/2$, $t_3 := a/2$.

4 шаг (Удаление лишних двоек.) Если t_3 четно, то присвоить

$$\begin{cases} t_3 := t_3/2, \quad t_1 := t_1/2, & \text{если } t_1 \text{ четно}, \\ t_3 := t_3/2, \quad t_1 := (t_1 + b)/2, & \text{если } t_1 \text{ нечетно}, \end{cases}$$

и вернуться на 4 шаг.

5 шаг. (Петля.) Если $t_3 > 0$, то $u := t_1$, $d := t_3$; иначе $v_1 := b - t_1$, $v_3 := -t_3$.

6 шаг. (Вычитание.) Присвоить

$$t_1 := u - v_1, \quad t_3 := d - v_3.$$

Если $t_1 < 0$, то $t_1 := t_1 + b$. Если $t_3 \neq 0$, то перейти на 4-й шаг.

7 шаг. (Окончание.) Присвоить $v := (d - au)/b$, $d := 2^k d$. Если $f_2 = 1$, то поменять u и v местами. Присвоить $u := u - vq$. Выдать (u, v, d) при $f_1 = 1$, (v, u, d) при $f_1 = 0$.

Конец алгоритма.

Мы закончим наше Приложение описанием двух методов возведения в степень. Пусть G — какая-либо мультипликативная группа, $g \in G$, $n \in \mathbb{N}$. Мы хотим найти $h = g^n \in G$. Пусть $n = b_k 2^k + b_{k-1} 2^{k-1} + \ldots$ $\ldots + b_1 2 + b_0$ есть двоичное представление n, т. е. $b_i \in \{0; 1\}$, $b_k = 1$.

1 способ. Сначала находим элементы $h_j = g^{2^j}$, $j = 0, 1, \ldots, k$; при этом $h_0 = g$, $h_{j+1} := h_j \cdot h_j$. Затем находим

$$h = \prod_{j:\ b_j = 1} h_j.$$

Для нахождения h этим способом требуется не более чем $2k + 1 = O(\log n)$ операций умножения в группе G.

2 способ. Находим последовательно

$$\hat{h}_j = g^{b_k 2^{j-1} + b_{k-1} 2^{j-2} + \ldots + b_{k-j+1}},\ j = 1, 2, \ldots, k+1.$$

В начале $\hat{h}_1 = g = g^{b_k}$. Далее, если мы уже нашли $\hat{h}_j$, то $\hat{h}_{j+1} := \hat{h}_j \cdot \hat{h}_j$, если $b_{k-j} = 0$; $\hat{h}_{j+1} := \hat{h}_j \cdot \hat{h}_j \cdot g$, если $b_{k-j} = 1$. В конце $\hat{h}_{k+1} = h = g^n$. Здесь также нужно $O(\log n)$ операций умножения в группе G.

Преимущество второго способа перед первым заключается в следующем. Пусть мы хотим найти $5^{10000} \in \mathbb{Z}$. Степени 5^{2^j} быстро растут. В первом способе мы возводим в квадрат и затем перемножаем большие числа. Во втором способе при нахождении $\hat{h}_{j+1}$ мы перемножаем большие числа $\hat{h}_j \cdot \hat{h}_j$, но домножаем на $g = 5$ (при $b_{k-j} = 1$), т. е. на маленькое число. В этом заключается существенная экономия.

Описанные способы называются *бинарными методами возведения в степень*.

Литература

[1] *Акритас А.* Основы компьютерной алгебры с приложениями. М.: Мир, 1994.

[2] *Алексеев В.Б.* Сложность умножения матриц. Обзор // Кибернетич. сборн. 1988. Вып. 25. С. 189—236.

[3] *Алферов А.П.*, *Зубов А.Ю.*, *Кузьмин А.С.*, *Черёмушкин А.В.* Основы криптографии. М.: Гелиос АРВ, 2002. 2-е изд.

[4] *Анохин М.И.*, *Варновский Н.П.*, *Сидельников В.М.*, *Ященко В.В.* Криптография в банковском деле. М.: МИФИ, 1997.

[5] *Ахо А.*, *Хопкрофт Дж.*, *Ульман Дж.* Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.

[6] *Березин И.С.*, *Жидков Н.П.* Методы вычислений. Т. 1. М.: Наука, 1966.

[7] *Берлекэмп Э.* Алгебраическая теория кодирования. М.: Мир, 1971.

[8] *Боревич З.И.*, *Шафаревич И.Р.* Теория чисел. М.: Наука, 1985.

[9] *Ван дер Варден Б.Л.* Алгебра. М.: Наука, 1976.

[10] *Василенко О.Н.* Современные способы проверки простоты чисел. Обзор // Кибернетич. сборн. 1988. Вып. 25. С. 162—188.

[11] *Василенко О.Н.* Некоторые алгоритмы построения больших простых чисел // Вестн. Моск. ун-та. Сер. 1. Матем. Механ. 1997. № 5. С. 62—64.

[12] *Василенко О.Н.* О некоторых свойствах чисел Ферма // Вестн. Моск. ун-та. Сер. 1. Матем. Механ. 1998. № 5. С. 56—58.

[13] *Василенко О.Н.* Об алгоритме Миллера—Рабина // Вестн. Моск. ун-та. Сер. 1. Матем. Механ. 2000. № 2. С. 41—42.

[14] *Василенко О.Н.* О дискретном логарифмировании в некоторых группах // Вестн. Моск. ун-та. Сер. 1. Матем. Механ. 2000. № 5. С. 53—55.

[15] *Василенко О.Н.* О дискретном логарифмировании по составному модулю // IV Международная конференция «Современные проблемы теории чисел и ее приложения». Тула, 10—15 сентября, 2001 / Тезисы докладов. С. 35—36.

[16] *Василенко О.Н.* Об одном применении тригонометрических сумм для проверки простоты чисел // Вестн. Моск. ун-та. Сер. 1. Матем. Механ. 2001. № 5. С. 49—51.

[17] *Василенко О. Н.* Применение круговых полей в криптосистемах RSA // IV Международная конференция «Современные проблемы теории чисел и ее приложения». Тула, 10-15 сентября, 2001 / Тезисы докладов. С 36—37.

[18] *Виноградов И. М.* Основы теории чисел. М.: Наука, 1972.

[19] *Гантмахер Ф. Р.* Теория матриц. М., 1954.

[20] *Гашков С. Б.* Упрощенное обоснование вероятностного теста Миллера-Рабина для проверки простоты чисел // Дискретная математика. 1998. Т. 10 (4). С. 35—38.

[21] *Григорьев Д. Ю.* Разложение многочленов над конечным полем и решение систем алгебраических уравнений // Зап. науч. семин. ЛОМИ АН СССР. 1984. № 137. С. 20–79.

[22] *Дэвенпорт Дж.*, *Сирэ И.*, *Турнье Э.* Компьютерная алгебра. М.: Мир, 1991.

[23] *Карацуба А. А.*, *Офман Ю. П.* Умножение многозначных чисел на автоматах // ДАН СССР. 1961. Т. 145 (2). С. 293—294.

[24] *Касселс Дж.* Введение в геометрию чисел. М.: Мир, 1965.

[25] *Кнут Д.* Искусство программирования. Т. 2. Получисленные алгоритмы. Вильямс: М.—СПб.—Киев, 2000. 3-е издание.

[26] *Коновальцев И. В.* Об одном алгоритме решения систем линеных уравнений в конечных полях // Проблемы кибернетики. 1967. Вып. 19. С. 269—274.

[27] *Кострикин А. И.* Введение в алгебру. М.: Наука, 1977.

[28] *Лемеш А. Н.* О λ-низких числах // Тезисы докл. 12 междунар. конф. «Проблемы теоретической кибернетики». Часть II. Ниж. Новг., 1999. С. 135.

[29] *Ленг С.* Введение в теорию диофантовых приближений. М.: Мир, 1970.

[30] *Ленг С.* Эллиптические функции. М.: Наука, 1984.

[31] *Лидл Р.*, *Нидеррайтер Г.* Конечные поля. Т. 1, 2. М.: Мир, 1988.

[32] *Матюхин Д. В.* Об асимптотической сложности дискретного логарифмирования в поле $GF(p)$ // Дискретная математика. 2002. Т. 15 (1). С. 28—49.

[33] *Матюхин Д. В.*, *Мурашов Н. Н.* Модификация метода решета числового поля для дискретного логарифмирования в поле $GF(p)$ // Обозр. прикл. и промышл. матем. 2000. Т. 7 (2). С. 387—389,

[34] *Нечаев В. И.* Сложность дискретного логарифма // Научные труды МГПУ. 1994. С. 46–49.

[35] *Нечаев В.И.* К вопросу о сложности детерминированного алгоритма для дискретного логарифма // Мат. заметки. 1994. Т. 55 (2). С. 91—101.
[36] *Нечаев В.И.* Элементы криптографии. М.: Высшая школа, 1999.
[37] *Ноден П.*, *Китте К.* Алгебраическая алгоритмика. М.: Мир, 1999.
[38] *Панкратьев Е.В.* Компьютерная алгебра. Факторизация многочленов. М.: Изд-во МГУ, 1988.
[39] *Тоом А.Л.* О сложности схемы из функциональных элементов, реализующей умножение целых чисел // ДАН СССР. 1963. Т. 150 (3). С. 496—498.
[40] *Уильямс Х.* Проверка чисел на простоту с помощью вычислительных машин // Кибернетич. сборн. 1986. Вып. 23. С. 51—99.
[41] *Хинчин А.Я.* Цепные дроби. М.: Наука, 1978.
[42] *Чебышев П.Л.* Полное собрание сочинений. Т. 1. Теория чисел. Изд-во АН СССР, 1946.
[43] *Чистов А.Л.* Алгоритм полиномиальной сложности для разложения многочленов и нахождения компонент многообразия в субэкспоненциальное время // Зап. науч. семин. ЛОМИ АН СССР. 1984. № 137. С. 124—188.
[44] *Adleman L.* A subexponential algorithm for the discrete logarithm problem with applications to cryptography // Proc. 20th Ann. IEEE Symp. Found. Comput. Sci. 1979. P. 55—60.
[45] *Adleman L.* The function field sieve // Proceedings of ANTS-I. 1994. (Lect. Notes in Comput. Sci.; V. 877). P. 108—121.
[46] *Adleman L.*, *Pomerance C.*, *Rumely R.S.* On distinguishing prime numbers from composite numbers // Ann. Math. 1983. V. 117. P. 173—206.
[47] *Adleman L.*, *Huang M.-D.A.* Primality testing and abelian varietes over finite fields. 1992. (Lect. Notes in Math.; V. 1512).
[48] *Adleman L.*, *McCurley K.* Open problems in number theoretic complexity // Proceedings of ANTS-I. 1994. (Lect. Notes in Comput. Sci.; V. 877). P. 291—322.
[49] *Adleman L.M.*, *Manders K.*, *Miller G.L.* On taking roots in finite fields // Proc. 18th Ann. Symp. Found. Comput. Sci. 1977. P. 175—178.
[50] *Agrawal M.*, *Kayal N.*, *Saxena N.* PRIMES is in P. Preprint, August 2002.
[51] *Alford W.R.*, *Granville A.*, *Pomerance C.* There are infinitely many Carmichael numbers // Ann. Math. 1994. V. 140. P. 703—722.

[52] *Alford W.R.*, *Granville A.*, *Pomerance C.* On the difficulty of finding reliable witnesses (invited talk) // Proceedings of ANTS-I. 1994. (Lect. Notes in Comput. Sci.; V. 877). P. 1—16.
[53] *Alt H.* Square rooting is as difficult as multiplication // Computing. 1979. V. 21. P. 221—232.
[54] *Ankeny N.C.* The least quadratic non-residue // Ann. Math. 1952. V. 55. P. 65—72.
[55] *Apostol T.M.* Introduction to analytic number theory. Springer-Verlag, 1997.
[56] *Atkin A.O.L.*, *Morain F.* Elliptic curves and primality proving // Math. Comp. 1993. V. 61 (203). P. 29—67.
[57] *Atkin A.O.L.*, *Morain F.* Finding suitable curves for elliptic method of factoring // Math. Comp. 1993. V. 60 (201). P. 399—405.
[58] *Atkins D.*, *Graff M.*, *Lenstra A.K.*, *Leyland P.C.* The magic words are squeamish ossifrage // Advances in cryptology — ASIACRYPT'94 (Wollongong, 1994). 1995. (Lecture Notes in Computer Science; V. 917). P. 263—277.
[59] *Bach E.*, *Shallit J.* Factoring with cyclotomic polynomials // Math. Comp. 1989. V. 52 (185). P. 201—219.
[60] *Bach E.*, *Shallit J.* Algorithmic number theory. V. 1. MIT Press, 1996.
[61] *Baker R.C.*, *Harman G.* The Brun—Titchmarsh theorem on average // Proc. Conf. in Honour of Heini Halberstam. V. 1. 1996. P. 39—103.
[62] *Ben-Or M.* Probabilistic algorithms in finite fields // Proc. 22nd Ann. IEEE Symp. Found. Comput. Sci. 1981. P. 394—398.
[63] *Berlekamp E.R.* Factoring polynomials over finite fields // Bell System Tech. J. 1967. V. 46. P. 1853—1859.
[64] *Bernstein D.J.* Detecting perfect powers in essentially linear time // Math. Comp. 1998. V. 67 (223). P. 1253—1283.
[65] *Blake I.F.*, *Seroussi G.*, *Smart N.P.* Elliptic curves in cryptography. Cambridge University Press, 1999.
[66] *Boender H.*, *te Riele H.J.J.* Factoring integers with large prime variations of the quadratic sieve / CWI Report NM-R9513. 1995.
[67] *Borodin A.B.*, *Munro I.* The computational complexity of algebraic and numeric problems. N. Y.: American Elsevier, 1975.
[68] *Bosma W.*, *van der Hulst M.P.* Faster primality testing (extended abstract) // Advances in Cryptology — EuroCrypt'89 / Jean-Jacques Quisquater and Joos Vandewalle, editors. Berlin: Springer-Verlag, 1989. (Lect. Notes in Comput. Sci.; V. 434). P. 652—656.

[69] *Bosselaerts A.*, *Govaerts R.*, *Vandewalle J.* Comparison of three modular reduction functions // Advances in Cryptology — Crypto'93 / Douglas R. Stinson, editor. Berlin: Springer-Verlag, 1993. (Lect. Notes in Comput. Sci.; V. 773). P. 175—186.

[70] *Brassard G.*, *Monet S.*, *Zuffelato D.* Algorithmes pour l'arithmetique des tres grands entiers // Techniques and Science Informatique. 1986. V. 5. P. 89—102.

[71] *Brent R.P.* An improved Monte Carlo factorization algorithm // BIT. 1980. V. 20. P. 176—184.

[72] *Brent R.P.* Some integer factorization algorithms using elliptic curves // Austral. Comput. Sci. Comm. 1986. V. 8. P. 149—163.

[73] *Brent R.P.* Factorization of the tenth Fermat number // Math. Comp. 1999. V. 68. P. 429—451.

[74] *Brent R.P.* Some parallel algorithms for integer factorisation // Lect. Notes in Comput. Sci. 1999. V. 1685. P. 1—22.

[75] *Brent R.P.*, *Pollard J.M.* Factorization of the eighth Fermat number // Math. Comp. 1981. V. 36. P. 627—630.

[76] *Brentjes A.J.* Multidimensional continued fraction algorithms. Amsterdam, 1981. (Mathematical Centre Tracts; V. 145).

[77] *Bressoud D.M.* Factorization and primality testing. Springer-Verlag, 1989.

[78] *Brillhart J.* A note on finding depencensies over $GF(2)$ // Utilitas Math. 1989. V. 36. P. 211—213.

[79] *Brillhart J.*, *Morrison M.A.* A method of factoring and the factorization of F_7 // Math. Comp. 1975. V. 29. P. 183—205.

[80] *Brillhart J.*, *Tonascia J.*, *Weinberger P.* On the Fermat quotient // Computers in number theory. London, N. Y.: Acad. Press, 1971. P. 213—222.

[81] *Buchberger B.*, *Winkler F.* Gröbner bases and applications. Cambridge Univ. Press, 1998. (London Math. Soc. Lecture Notes Series; V. 251).

[82] *Buchmann J.*, *Jacobson M.J.*, *Teske E.* On some computational problems in finite abelian groups // Math. Comp. 1997. V. 66 (220). P. 1663—1687.

[83] *Buchmann J.*, *Weber D.* Discrete logarithms: Recent progress // Proc. Internat Conf. on Coding Theory, Cryptography and Related Areas, Guanajuato. Springer-Verlag, 2000. P. 42—56.

[84] *Buell D.A.* Binary quadratic forms: classical theory and modern computations. Springer-Verlag, 1989.

[85] *Cantor D.G.*, *Zassenhaus H.* A new algorithm for factoring polynomials over finite fields // Math. Comp. 1981. V. 36. P. 587—592.

[86] *Caron T.R.*, *Silverman R.D.* Parallel implementation of the quadratic sieve. J. Supercomputing. 1988. V. 1. P. 273—290.

[87] *Cavallar S.*, *Dodson B.*, *Lenstra A.K.*, *Leyland P.C.*, *Lioen W.M.*, *Montgomery P.L.*, *Murphy B.*, *te Riele H.J.J.*, *Zimmerman P.* Factorization of RSA-140 using the number field sieve / CWI Report MAS-R9925, September 1999.

[88] *Cavallar S.*, *Lioen W.M.*, *te Riele H.J.J.*, *Dodson B.*, *Lenstra A.K.*, *Montgomery P.L.*, *Murphy B.* et al. Factorization of 512-bit RSA-modulus / CWI Report MAS-R0007, February 2000.

[89] *Cohen H.* A course in computational algebraic number theory. Springer-Verlag, 1993.

[90] *Cohen H.*, *Lenstra H.W.* Primality testing and Jacobi sums // Math. Comp. 1984. V. 42 (165). P. 297—330.

[91] *Comba P.G.* Experiments in fast multiplication of integers / Technical Report G320-2158, IBM, Cambridge Scientific Center, February 1989.

[92] *Comba P.G.* Exponentiation cryptosystems on IBM PC // IBM Systems J. 1990. V. 29 (4). P. 29—37.

[93] *Contini S.* Factoring integers with the self initializing quadratic sieve / Master's thesis. Univ. Georgia, 1997.

[94] *Cook S.A.* On the minimum computation time of functions / Doctoral thesis. Harvard University, Cambridge, Mass., 1966.

[95] *Coppersmith D.* Fast evaluation of discrete logarithms in fields of characteristic two. IEEE Trans // Inform. Theory. 1984. V. 30 (4). P. 587—594.

[96] *Coppersmith D.* Solving homogeneous linear equations over $GF(2)$ via block Wiedemann algorithm // Math. Comp. 1994. V. 62 (205). P. 333—350.

[97] *Coppersmith D.*, *Odlyzko A.*, *Schroeppel R.* Discrete logarithms in $GF(p)$ // Algorithmica. 1986. V. 1. P. 1—15.

[98] *Coppersmith D.*, *Winograd S.* On the asymptotic complexity of matrix multiplication // SIAM J. Comput. 1982. V. 11. P. 472—492.

[99] *Couvreur C.*, *Quisquater J.J.* An introduction to fast generation of large primes // Philips J. Res. 1982. V. 37. P. 231—264. Errata in: 1983. V. 38. P. 77.

[100] *Cox D.*, *Little J.*, *O'Shea D.* Ideals, verietes and algorithms. N. Y.: Springer-Verlag, 1992. (Undergraduate Texts in Mathematics).
[101] *Crandall R.*, *Pomerance C.* Prime numbers: a computational perspective. Springer-Verlag, 2001.
[102] *de Weger B.* Algorithms for Diophantine equations / Dissertation. Centrum voor Wiskunde en Informatica, Amsterdam, 1988.
[103] *Denny T.*, *Müller V.* On the reduction of composed relations from the number field sieve // Proceedings of ANTS-II. 1996. (Lect. Notes in Comput. Sci.; V. 1122). P. 75—90.
[104] *Denny T.F.* Lösen großer dünnbesetzter Gleichungssysteme über endlichen Primkörpern / Dissertation. Universität des Saarlandes, Saarbrücken, 1997.
[105] *Diaz A.*, *Hitz M.*, *Kaltofen E.*, *Lobo A.* Process scheduling in DSC and the large sparce linear systems challenge // Lect. Notes in Comput. Sci. 1993. V. 722. P. 66—80.
[106] *Dixon B.*, *Lenstra A.K.* Massively parallel elliptic curve factoring // Lect. Notes in Comput. Sci. V. 658. 1993. P. 183—193.
[107] *Dodson B.*, *Lenstra A.K.* NFS with four large primes: an explosive experiment // Advances in Cryptology — Crypto'95. 1995. (Lect. Notes in Comput. Sci.; V. 963). P. 372—385.
[108] *ElGamal T.* A subexponential-time algorithm for computing discrete logarithms over $GF(p^2)$ // IEEE Trans. Inform. Theory. 1985. V. 31. P. 473—481.
[109] *ElGamal T.* On computing logarithm over finite fields // Advances in cryptology — CRYPTO'85 (Santa Barbara, Calif., 1985). 1986. (Lect. Notes in Comput. Sci.; V. 218). P. 396—402.
[110] *Elkenbracht-Huizing M.* An implementation of the number field sieve // Experimental Mathematics. 1996. V. 5. P. 231—253.
[111] *Elkenbracht-Huizing M.* A multiple polynomial general number field sieve // Proceedings of ANTS-II. 1996. (Lect. Notes in Comput. Sci.; V. 1122). P. 99—114.
[112] *Elkenbracht-Huizing M.* Factoring integers with the number field sieve / PhD thesis. Leiden Univ., 1997.
[113] *Elkies N.D.* Elliptic and modular curves over finite fields and related computaional issues // Computational perspectives in number theory: Proc. of a Conf. in Honor of A.O.L. Atkin / J.T. Teitelbaum and D.A. Buell, editors. 1998. (Amer. Math. Soc. Inf. Press; V. 7). P. 21—76.
[114] *Ernvall R.*, *Mëtsankÿla T.* On the p-divisibility of Fermat quotients // Math. Comp. 1997. V. 66 (219). P. 1353—1365.

[115] *Fagin B.S.* Large integers multiplication on massively parallel processors // Proc. Frontiers'90: Third Symp. on the Frontiers of Massively Parallel Computation. IEEE Press, 1990. P. 38—42.

[116] *Ferguson H.R.P.* A short proof of the existence of vector Eucledian algorithm // Proc. Amer. Math. Soc. 1986. V. 97. P. 8—10.

[117] *Ferguson H.R.P.* A non-inductive $GL(n, 2)$ algorithm that constructs integral linear relations for n $\mathbb{Z}$-linearly dependent real numbers // J. Algorithms. 1987. V. 8 (1). P. 131—142.

[118] *Ferguson H.R.P.*, *Bailey D.H.*, *Arno S.* Analysis of integer relation finding algorithm // Math. Comp. 1999. V. 68 (225). P. 351—369.

[119] *Ferguson H.R.P.*, *Forcade R.W.* Generalization of the Eucledian algorithm for real numbers to all dimensions higher than two // Bull. Amer. Math. Soc. (N. S.). 1979. V. 1. P. 912—914.

[120] *Ferguson H.R.P.*, *Forcade R.W.* Multidimensional Eucledian algorithms // J. Reine Angew. Math. 1982. V. 334. P. 171—181.

[121] *Fincke U.*, *Pohst M.* Improved methods for calculating vectors of short length in a lattice, including acomplexity analysis // Math. Comp. 1985. V. 44. P. 463—471.

[122] *Fleischmann P.* Connections between the algorithms of Berlekamp and Niederraiter for factoring polynomials over $\mathbb{F}_q$ // Linear Algebra and Applications. 1993. V. 192. P. 101—108.

[123] *Fouvry E.* Theoreme de Brun-Titchmarsh: application an theoreme de Fermat // Invent. Math. 1985. V. 79. P. 383—407.

[124] *Garner H.* The residue number system // IRE Transactions on Electronic Computers. 1959. V. 8. P. 140—147.

[125] *Gianni P.*, *Mora T.* Algebraic solution of systems of polynomial equations using Gröbner bases // Applied algebra, algebraic algorithms and error-correcting codes (Menorca, 1987). 1989. (Lect. Notes in Comput. Sci.; V. 356). P. 247—257.

[126] *Goldwasser S.*, *Kilian J.* Almost all primes can be quickly certified // Proc. 18-th Ann. ACM Symp. on Theory of Computing. 1986. P. 316—329.

[127] *Gordon D.* Discrete logarithms in $GF(p)$ using the number field sieve // SIAM J. Disc. Math. 1993. V. 6. P. 124—138.

[128] *Gordon D.M.*, *McCurley K.S.* Massively parallel computation of discrete logarithms // Advances in Cryptology — Crypto'92 / Ernest F. Brickell, editor. Berlin: Springer-Verlag, 1993. (Lect. Notes in Comput. Sci.; V. 740). P. 312—323.

[129] *Hastad J.*, *Just B.*, *Lagarias J.C.*, *Schnorr C.P.* Polynomial time algorithms for finding integer relations among real numbers // SIAM J. Comput. 1989. V. 18. P. 859—881.

[130] *Hellman M.E.*, *Reyneri J.M.* Fast computation of discrete logarithms in $GF(q)$ // Advances in Cryptology — CRYPTO'82. N. Y.: Plenum Press, 1983. P. 3—13.

[131] *Herlestam T.*, *Johannesson R.* On computing logarithms over $GF(2^p)$ // BIT. 1981. V. 21. P. 326—336.

[132] *Hong S.M.*, *Oh S.Y.*, *Yoon H.* New modular multiplication algorithms for fast modular exponentiation // Lect. Notes in Comput. Sci. 1996. V. 1070. P. 166—177.

[133] *Izu T.*, *Kogure J.*, *Noro M.*, *Yokoyama K.* Efficient implementation of Schoof's algorithm // Advances in cryptology — ASIACRYPT'98 (Beijing). 1998. (Lect. Notes in Comput. Sci.; V. 1514). P. 66—79.

[134] *Joux A.*, *Lercier R.* Improvements to the general number field sieve for discrete logarithms in prime fields. Preprint. `http://www.medicis.polytechnique.fr/~lercier`.

[135] *Joux A.*, *Lercier R.* Discrete logarithms in $GF(p)$ / e-mail to the NMBRTHRY mailing list, January 2001. `http://listserv.nodak.edu/archives/nmbrthry.html`.

[136] *Joux A.*, *Lercier R.* Discrete logarithms in $GF(p)$ / e-mail to the NMBRTHRY mailing list, April 2001. `http://listserv.nodak.edu/archives/nmbrthry.html`.

[137] *Kaltofen E.* Polynomial factorization 1987—1991 // LATIN'92 (São Paulo, 1992). 1992. (Lect. Notes in Comput. Sci.; V. 583). P. 294—313.

[138] *Kaltofen E.* Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparce linear systems // Applied algebra, algebraic algorithms and error-correcting codes (San Juan, PR, 1993). 1993. (Lect. Notes in Comput. Sci.; V. 673). P. 195—212.

[139] *Kaltofen E.*, *Lobo A.* Factoring high-degree polynomials by the black box Berlekamp algorithm // Proceedings of ISSAC'94. ACM Press, 1994. P. 90—98.

[140] *Kaltofen E.*, *Sanders B.D.* On Wiedemann's method of solving sparce linear systems // Applied algebra, algebraic algorithms and error-correcting codes (New Orleans, LA, 1991). 1991. (Lect. Notes in Comput. Sci.; V. 539). P. 29—38.

[141] *Kaltofen E.*, *Shoup V.* Fast polynomial factorization over high algebraic extensions of finite fields // Proceedings of ISSAC'97. ACM Press, 1997. P. 184—188.

[142] *Kaltofen E., Shoup V.* Subquadratic-time factoring of polynomials over finite fields // Math. Comp. 1998. V. 67 (223). P. 1179—1197.

[143] *Kannan R., Lenstra A.K., Lovasz L.* Polynomial factorization and nonrandomness of bits of algebraic and some transcendental numbers // Math. Comp. 1988. V. 50 (181). P. 235—250.

[144] *Koblitz N.* A course in number theory and cryptography. Springer-Verlag, 1987.

[145] *Koblitz N.* Elliptic curve cryptosystems // Math. Comp. 1987. V. 48. P. 203—209.

[146] *Koblitz N.* Algebraic aspects of cryptography. Springer-Verlag, 1998.

[147] *Konyagin S., Shparlinski I.* Linear complexity of discrete logarithm. Preprint, December 2000.

[148] *Konyagin S.V., Pomerance C.* On primes recognizable in deterministic polynomial time // Algorithms and combinatorics. Springer-Verlag, 1997. (The mathematics of Paul Erdös; V. 1). P. 176—198.

[149] *LaMacchia B.* Basis reduction algorithms and subset sum problems / Thesis. MIT Artifical Intelligence Lab., 1991.

[150] *LaMacchia B., Odlyzko A.* Solving large sparse linear systems over finite fields // Advances in Cryptology — CRYPTO'90. 1990. (Lecture Notes in Computer Science; V. 537). P. 109—133.

[151] *LaMacchia B.A., Odlyzko A.M.* Computation of discrete logarithm in prime fields // Des. Codes Cryptogr. 1991. V. 1. P. 47—62.

[152] *Lambert R.* Computational aspects of discrete logarithms / PhD thesis. Univ. of Waterloo, Dept. Electrical Comp. Eng., 1996.

[153] *Lanczos C.* Solution of systems of linear equations by minimized iterations // J. Res. Nat. Bur. Standards. 1952. V. 49. P. 33—53.

[154] *Lay G.-J., Zimmer H.G.* Constructing elliptic curves with given group order over large finite fields // Algorithmic number theory (Ithaca, NY, 1994). 1994. (Lect. Notes in Comput. Sci.; V. 877). P. 250—263.

[155] *Lazard D.* Resolution des systems d'equations algebraiques // Theor. Comput. Sci. 1981. V. 15. P. 77—110.

[156] *Lazard D.* Ideal basis and primary decomposition: case of two variables // J. Symb. Comput. 1985. V. 1. P. 261—270.

[157] *Lazard D.* Solving zero-dimensional algebraic systems // J. Symb. Comput. 1992. V. 13. P. 117—131.

[158] *Lehman R.S.* Factoring large integers // Math. Comp. 1974. V. 28. P. 637—646.

[159] *Lenstra A.K., Lenstra H.W.*, editors. The development of the number field sieve. 1993. (Lecture Notes in Mathematics; V. 1554).

[160] *Lenstra A.K., Lenstra H.W., Lovász L.* Factoring polynomials with rational coefficients // Math. Ann. 1982. V. 261. P. 515—534.

[161] *Lenstra A.K., Lenstra H.W., Manasse M.S., Pollard J.M.* The number field sieve // Proc. 22nd ACM Symposium on Theory of Computing. 1990. P. 564—572.

[162] *Lenstra A.K., Lenstra H.W., Manasse M.S., Pollard J.M.* The factorization of the ninth Fermat number // Math. Comp. 1993. V. 61 (203). P. 319—349.

[163] *Lenstra A.K., Manasse M.S.* Factoring with two large primes // Math. Comp. 1994. V. 63. P. 785—798.

[164] *Lenstra H.W.* Primality testing algorithms (after Adleman, Rumely and Williams) // Bourbaki Seminar. V. 1980/81. 1981. (Lect. Notes in Math.; V. 901). P. 243—257.

[165] *Lenstra H.W.* Divisors in residue classes // Math. Comp. 1984. V. 42 (165). P. 331—340.

[166] *Lenstra H.W.* Elliptic curves and number-theoretic algorithms // International Congress of Mathematicians. 1986. P. 99—120.

[167] *Lenstra H.W.* Factoring integers with elliptic curves // Ann. Math. 1987. V. 126. P. 649—673.

[168] *Lenstra H.W.* Finding isomorphisms between finite fields // Math. Comp. 1991. V. 56 (193). P. 329—347.

[169] *Lenstra H.W., Pomerance C.* A rigorous time bound for factoring integers // J. Amer. Math. Soc. 1992. V. 5 (3). P. 483—516.

[170] *Lerch M.* Zur Theorie des Fermatischen Quotienten $\frac{a^{p-1}-1}{p} = q(a)$ // Math. Ann. 1905. V. 60. P. 471—490.

[171] *Lercier R.* Algorithmique des courbes dans les corps finis / These. L'Ecole Polytechnique, Laboratoire D'Informatique, CNRC, Paris, 1997.

[172] *Li T.Y.* Solving polynomial systems // Math. Intelligencer. 1987. V. 9. P. 33—39.

[173] *Lovorn Bender R.* Rigorous, subexponential algorithms for discrete logarithms in $GF(p^2)$ // SIAM J. Discrete Math., to appear.

[174] *Lovorn Bender R.*, *Pomerance C.* Rigorous discrete logarithm computations in finite fields via smooth polynomials // Computational perspectives in number theory (Chicago, 1995). Amer. Math. Soc., 1998. (AMS/IS Stud. Adv. Math.; V. 7). P. 221—232.

[175] *Massey J.L.* Shift-register synthesis and BCH decoding // IEEE Trans. Inform. Theory. 1969. V. 15. P. 122—127.

[176] *McCurley K.S.* The discrete logarithm problem // Cryptology and computational number theory (Boulder, CO, 1989). Amer. Math. Soc., 1990. (Proc. of Symp. Appl. Math.; V. 42). P. 49—74.

[177] *McCurley K.S.* Odds and ends from cryptology and computational number theory // Cryptology and computational number theory (Boulder, CO, 1989). Amer. Math. Soc., 1990. (Proc. of Symp. Appl. Math.; V. 42). P. 145—166.

[178] *McKee J.* Speeding Fermat's factoring method // Math. Comp. 1999. V. 68 (228). P. 1729—1737.

[179] *Menezes A.* Elliptic curve public key cryptosystems. Kluwer Acad. Publ., 1993.

[180] *Menezes A.*, *Van Oorschot P.C.*, *Vanstone S.A.* Handbook of applied cryptography. CRC Press, 1997.

[181] *Menezes A.*, *Qu M.*, *Vanstone S.* IEEE P1363 Standard, Part 4: Elliptic curve systems, 1995.

[182] *Menezes A.J.*, *Vanstone S.A.*, *Zuccherato R.J.* Counting points on elliptic curves over F_{2^m} // Math. Comp. 1993. V. 60 (201). P. 407—420.

[183] *Mignotte M.* An inequality about factors of polynomials // Math. Comp. 1974. V. 28. P. 1153—1157.

[184] *Mihailescu P.* Cyclotomic primality proving — recent developments // Proceedings of ANTS-III. 1998. (Lect. Notes in Comput. Sci.; V. 1423). P. 95—110.

[185] *Mihailescu P.* Cyclotomy of rings and primality testing / PhD thesis. Swiss Federal Institute of Technology, Zürich, 1998.

[186] *Mihailescu P.* Fast generation of provable primes using search in arithmetic progressions. Preprint, 1998.

[187] *Miller G.L.* Riemann's hypothesis and tests for primality // J. Comput. and Syst. Sci. 1976. V. 13. P. 300—317. [Перевод: Кибернетический сборник. 1986. Вып. 23. С. 31—50.]

[188] *Miller V.* Use of elliptic curves in cryptography // Advances in cryptology — CRYPTO'85 (Santa Barbara, Calif., 1985). 1986. (Lecture Notes in Comput. Sci.; V. 218). P. 417—426.

[189] *Miyaji A.* Elliptic curves over F_p suitable for cryptosystems // Advances in cryptology — AUSCRYPT'92 (Gold Coast, 1992). 1993. (Lect. Notes in Comput. Sci.; V. 718). P. 479—471.

[190] *Monier L.* Evaluation and comparision of two efficient probabilistic primality testing algorithms // Theor. Comput. Sci. 1980. V. 12. P. 97—108.

[191] *Montgomery P.L.* Modular multiplication without trial division // Math. Comp. 1985. V. 44 (170). P. 519—521.

[192] *Montgomery P.L.* Speeding the Pollard and elliptic curve methods of factorization // Math. Comp. 1987. V. 48 (177). P. 243—264.

[193] *Montgomery P.L.* A block Lanczos algorithm for finding dependencies over $GF(2)$ // Advances in Cryptology — EuroCrypt'95 / Louis C. Guillou and Jean-Jacques Quisquater, editors. Berlin: Springer-Verlag, 1995. (Lect. Notes in Comput. Sci.; V. 921). P. 106—120.

[194] *Montgomery P.L., Silverman R.D.* A FFT-extension to the $p-1$ factoring algorithm // Math. Comp. 1990. V. 54 (190). P. 839—854.

[195] *Morain F.* Atkin's test: news from the front. Preprint.

[196] *Morain F.* Elliptic curves, primality proving and some titanic primes. Preprint.

[197] *Morain F.* Solving equations of small degree modulo large primes. Preprint.

[198] *Morain F.* Distributed primality proving and the primality of $(2^{3539}+1)/3$. Preprint, 1990.

[199] *Morain F.* Primality proving using elliptic curves: an update // Proceedings of ANTS-III. 1998. (Lect. Notes in Comput. Sci.; V. 1423). P. 111—127.

[200] *Morain F., Olivos J.* Speeding up the computations on an elliptic curve using addition-subtraction chains // Inform. Theor. et Appl. 1990. V. 24. P. 531—544.

[201] *Mullen G.L., White D.* A polynomial representation for logarithms in $GF(q)$ // Acta Arithm. 1986. V. 47. P. 255—261.

[202] *Müller V.* Ein Algorithmus zur Bestimmung der Punktzahl elliptischer Kurvenüber endlichen Körpern der Characteristic grössen drei / PhD thesis, Universität des Saarlandes, 1995.

[203] *Murphy B.A.* Modelling the yield of number field sieve polynomials // Proceedings of ANTS-III. 1998. (Lect. Notes in Comput. Sci.; V. 1423). P. 137—150.

[204] *Murphy B.A.* Polynomial selection for the number field sieve integer factorisation algorithm / PhD thesis. Australian National University, July 1999.

[205] *Murphy B.A.*, *Brent R.P.* On quadratic polynomials for the number field sieve // Austral. Comput. Sci. Comm. 1998. V. 20. P. 199—213.

[206] *Niederreiter H.* A new efficient factorization algorithm for polynomials over small finite fields // Appl. Algebra Engrg. Comm. Comput. 1993. V. 4. P. 81—87.

[207] *Niederreiter H.*, *Göttfert R.* Factorization of polynomials over finite fields and characteristic sequences // J. Symbolic Computation. 1993. V. 16 (5). P. 401—412.

[208] *Odlyzko A.* Discrete logarithms and smooth polynomials // Contemp. Math. 1994. V. 168. P. 269—278.

[209] *Odlyzko A.* Discrete logarithms: the past and the future // Designs, Codes and Cryptography. 2000. V. 19. P. 129—145.

[210] *Odlyzko A.M.* Discrete logarithms in finite fields and their cryptographic significance // Advances in Cryptology: Proceedings of EuroCrypt'84 / Thomas Beth, Norbert Cot, and Ingemar Ingemarsson, editors. Berlin: Springer-Verlag, 1984. (Lect. Notes in Comput. Sci.; V. 209). P. 224—316.

[211] *Odlyzko A.M.* The future of integer factorization // CryptoBytes. 1995. V. 1 (2). P. 5—12.

[212] *Parkinson D.*, *Wunderlich M.* A compact algorithm for Gaussian elimination over $GF(2)$ implemented on highly parallel computers // Parallel Computing. 1984. V. 1. P. 65—73.

[213] *Peralta R.* Implementation of the hypercube multiple polynomial sieve. Preprint.

[214] *Plaisted D.A.* Fast verification, testing and generation of large primes // Theoret. Comp. Sci. 1979. V. 9. P. 1—16. Errata in: 1981. V. 14. P. 345.

[215] *Pohlig S.*, *Hellman M.* An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance // IEEE Trans. Inform. Theory. 1978. V. 24. P. 106—110.

[216] *Pohst M.* A modification of the LLL-reduction algorithm // J. Symb. Comp. 1987. V. 4. P. 123—128.

[217] *Pohst M.*, *Zassenhaus H.* Algorithmic algebraic number theory. Cambridge University Press, 1989.

[218] *Pollard J.M.* Theorems on factorization and primality testing // Proc. Cambridge Phil. Soc. 1974. V. 76. P. 521—528.

[219] *Pollard J. M.* A Monte Carlo method for factorization // BIT. 1975. V. 15. P. 331—334.

[220] *Pollard J. M.* Monte Carlo methods for index computation (mod p) // Math. Comp. 1978. V. 32 (143). P. 918—924.

[221] *Pomerance C.* Analysis and comparision of some integer factoring algorithms // Computational methods in number theory. V. 1 / H. W. Lenstra and R. Tijdeman, editors. Amsterdam, 1982. P. 89—139.

[222] *Pomerance C.* The quadratic sieve factoring algorithm // Advances in cryptology (Paris, 1984). 1985. (Lecture Notes in Comput. Sci.; V. 209). P. 169—183.

[223] *Pomerance C.* Fast, rigorous factorization and discrete logarithms // Discrete Algorithms and complexity / A. Nozaki D. S. Johnson, T. Nishizeki and H. S. Wilf, editors. Orlando: Acad. Press, 1987. P. 119—144.

[224] *Pomerance C.* Very short primality proofs // Math. Comp. 1987. V. 48 (177). P. 315—322.

[225] *Pomerance C.* Factoring // Proc. of Symp. Appl. Math. 1990. V. 42. P. 24—47.

[226] *Pomerance C.* The number field sieve // Proc. of Symp. Appl. Math. 1994. V. 48. P. 465—480.

[227] *Pomerance C.* A tale of two sieves // Notices Amer. Math. Soc. 1996. V. 43. P. 1473—1485.

[228] *Pomerance C.*, *Selfridge J. L.*, *Wagstaff S. S.* The pseudoprimes to $2.5 \cdot 10^9$ // Math. Comp. 1980. V. 36 (151). P. 1003—1026.

[229] *Pomerance C.*, *Smith J. W.*, *Tuler R.* A pipeline architecture for factoring large integers with the quadratic sieve algorithm // SIAM J. Comput. 1988. V. 17 (2). P. 387—403.

[230] *Rabin M.* Probabilistic algorithms for testing primality // J. Number Theory. 1980. V. 12. P. 128—138.

[231] *Ribenboim P.* The book of prime number records. Springer-Verlag, 1988.

[232] *Ribenboim P.* The new book of prime number records. Springer-Verlag, 1996.

[233] *Riesel H.* Prime numbers and computer methods for factorization. Birkhauser, 1985. (Progr. in Math.; V. 57).

[234] *Riesel H.* Some soluble cases of the discrete logarithm problem // BIT. 1988. V. 28 (4). P. 839—851.

[235] *Satoh T.*, *Araki K.* Fermat quotients and polynomial time discrete log algorithm for anomalous elliptic curves // Comment. Math. Univ. Sancti Pauli. 1998. V. 47. P. 81—92.
[236] *Schirokauer O.* Discrete logarithms and local units. Phil. Trans. R. Soc. Lond. A. 1993. V. 345. P. 409—423.
[237] *Schirokauer O.* Using number fields to compute discrete logarithms in finite fields // Math. Comp. 2000. V. 69. P. 1267—1283.
[238] *Schirokauer O.*, *Weber D.*, *Denny T.* Discrete logarithms: the effectiveness of the index calculus method // Proceedings of ANTS-II. 1996. (Lect. Notes in Comput. Sci.; V. 1122). P. 337—362.
[239] *Schnorr C.P.* A more efficient algorithm for lattice basis reduction // J. Algorithms. 1988. V. 9. P. 47—62.
[240] *Schnorr C.P.*, *Euchner M.* Lattice basis reduction: improved practical algorithms and solving subset sum problems // Fundamentals of computation theory (Gosen, 1991). 1991. (Lect. Notes in Comput. Sci.; V. 529). P. 68—85.
[241] *Schnorr C.P.*, *Lenstra H.W.* A Monte-Carlo factoring algorithm with linear storage // Math. Comp. 1984. V. 43. P. 289—312.
[242] *Schönhage A.* The fundamental theorem of algebra in terms of computational complexity / Preliminary report, 1982. Math. Institute Univ. Tübingen.
[243] *Schönhage A.*, *Grotfeld A.F.W.*, *Vetter E.* Fast algorithms: a multitape Turing mashine implementation. Mannheim: BI-Wissenschaftsverlag, 1994.
[244] *Schoof R.* Elliptic curves over finite fields and the computation of square roots mod$p$ // Math. Comp. 1985. V. 44. P. 483—494.
[245] *Schoof R.* Counting points on elliptic curves over finite fields // J. Theorie des Nombres des Bordeaux. 1995. V. 7. P. 219—254.
[246] *Sedgewick R.*, *Szymanski T.G.*, *Yao A.C.* The complexity of finding cycles in periodic functions // SIAM J. Comput. 1982. V. 11 (2). P. 376—390.
[247] *Semaev I.A.* An algorithm for evaluation of discrete logarithms in some nonprime finite fields // Math. Comp. 1998. V. 67. P. 1679—1689.
[248] *Shanks D.* Class number, a theory of factorization and genera // Proc. Symp. Pure Math. V. 20. Providence, R. I.: AMS, 1971. P. 415—440.
[249] *Shoup V.* The deterministic complexity of factoring polynomials over finite fields // Inform. Process. Lett. 1990. V. 33 (5). P. 261—267.

[250] *Shoup V.* New algorithm for finding irreducible polynomials over finite fields // Math. Comp. 1990. V. 54. P. 435—447.

[251] *Shoup V.* Searching for primitive roots in finite fields // Math. Comp. 1992. V. 58 (197). P. 369—380.

[252] *Shoup V.* Fast construction of irreducible polynomials over finite fields // J. Symbolic Comput. 1994. V. 17 (5). P. 371—391.

[253] *Shoup V.* A new polynomial factorization algorithm and its implementation // J. Symbolic Comput. 1995. V. 20. P. 364—397.

[254] *Shoup V.* Lower bounds for discrete logarithms and related problems // Advances in Cryptology — EuroCrypt'97 / Walter Fumy, editor. Berlin: Springer-Verlag, 1997. (Lect. Notes in Comput. Sci.; V. 1233). P. 256—266.

[255] *Shparlinski I.E.* Number theoretic methods in cryptography: Complexity lower bounds. Birkhäuser, 1999.

[256] *Silverman J.H.* The arithmetic of elliptic curves, Springer-Verlag, 1986. (Graduate Texts in Mathematics; V. 106).

[257] *Silverman J.H.* Advanced topics in the arithmetic of elliptic curves. Springer-Verlag, 1994. (Graduate Texts in Mathematics; V. 151).

[258] *Silverman R.D.* The multiple polynomial quadratic sieve // Math. Comp. 1987. V. 48 (177). P. 329—339.

[259] *Silverman R.D.* Fast generation of random strong RSA primes. Preprint. RSA Laboratories, 1997.

[260] *Silverman R.D.*, *Wagstaff S.S.* A practical analysis of the elliptic curve factoring algorithm // Math. Comp. 1993. V. 61. P. 445—462.

[261] *Solinas J.A.* An improved algorithm for arithmetic on a family of elliptic curves // Advances in Cryptology — Crypto'97 / Burt Kaliski, editor. Berlin: Springer-Verlag, 1997. (Lecture Notes in Computer Science; V. 1294). P. 357—371.

[262] *Solovay R.*, *Strassen V.* A fast Monte-carlo test for primality // SIAM J. Comput. 1977. V. 6. P. 84—85. Errata in: 1978. V. 7. P. 117.

[263] *Stewart I.*, *Tall D.* Algebraic number theory. London—N. Y.: Chapman and Hall, 1986.

[264] *Strassen V.* Einige Resultate über Berechnungskomplexität // Jahresber. Deutsch. Math.-Verein. 1976/77. V. 78. P. 1—8.

[265] *te Riele H.* 227-digit SNFS factorization. `ftp://ftp.cwi.nl/pub/herman/SNFSgiants/SNFS-227`, January 2002.

[266] *te Riele H.J.J.*, *Lioen W.*, *Winter D.* Factoring with the quadratic sieve on large vector computers // Belgian J. Comp. Appl. Math. 1989. V. 27. P. 267—278.

[267] *Teske E.* Speeding up Pollard's rho method for computing discrete logarithms // Proceedings of ANTS-III. 1998. (Lect. Notes in Comput. Sci.; V. 1423). P. 541—554.

[268] *Teske E.* Square root algorithms for the discrete logarithm problem (a survey). Preprint, January 2001.

[269] *Thomé E.* Computation of discrete logarithms in $GF(2^{607})$ // Advances in Cryptology — AsiaCrypt'2001. 2001. (Lect. Notes in Comput. Sci.; V. 2248). P. 107—124.

[270] *Thomé E.* Discrete logarithms in $GF(2^{607})$. e-mail to the NMBRTHRY mailing list, February 2002.
`http://listserv.nodak.edu/archives/nmbrthry.html`.

[271] *Turk J.W.M.* Fast arithmetic operations on numbers and polynomials // Computational methods in number theory. V. 2 / H. W. Lenstra and R. Tijdeman, editors. Amsterdam, 1982. P. 43—54.

[272] *von zur Gathen J.*, *Shoup V.* Computing Frobenius maps and factoring polynomials // Comput. Complexity. 1992. V. 2. P. 187—224.

[273] *Voorhoeve M.* Factorization algorithms of exponential order // Computational methods in number theory. V. 1 / H. W. Lenstra and R. Tijdeman, editors. Amsterdam, 1982. P. 79—88.

[274] *Weber D.* An implementation of the general number field sieve to compute discrete logarithms mod p // Advances in Cryptology — EuroCrypt'95 / Louis C. Guillou and Jean-Jacques Quisquater, editors. Berlin: Springer-Verlag, 1995. (Lecture Notes in Computer Science; V. 921). P. 95—105.

[275] *Weber D.* Computing discrete logarithms with the general number field sieve // Proceedings of ANTS-II. 1996. (Lect. Notes in Comput. Sci.; V. 1122). P. 391—404.

[276] *Weber D.* On the computation of discrete logarithms in finite prime fields / PhD thesis. Univ. des Saarlandes, Saarbrücken, 1997.

[277] *Weber D.* Computing discrete logarithms with quadratic number rings // Advances in Cryptology — EUROCRYPT'98. Springer-Verlag, 1998. (Lect. Notes in Comput. Sci.; V. 1403). P. 171—183.

[278] *Weber D.*, *Denny T.* The solution of McCurley's discrete log challenge // Advances in Cryptology — CRYPTO'98. Springer-Verlag, 1998. (Lect. Notes in Comput. Sci.; V. 1462). P. 458—471.

[279] *Weber K.* An experiment in high-precision arithmetic on shared memory multiprocessors // ACM SIGSAM Bull. 1990. V. 24 (2). P. 22—44.

[280] *Western A.E., Miller J.C.P.* Tables of indices and primitive roots. Cambridge University Press, 1968. (Royal Society Mathematical Tables; V. 9).

[281] *Wiedemann D.H.* Solving sparce linear equations over finite fields // IEEE Trans. Inform. Theory. 1986. V. 32 (1). P. 54—62.

[282] *Williams H.C.* Some algorithms for solving $x^q \equiv N \pmod{p}$ // Proc. 3rd South East Conf. on Combinatorics, Graph Theory and Computing. 1972. P. 451—462.

[283] *Williams H.C.* A $p+1$ method of factoring // Math. Comp. 1982. V. 39 (159). P. 225—234.

[284] *Williams H.C.* Factoring on a computer // Math. Intell. 1984. V. 6 (3). P. 29—36.

[285] *Williams H.C., Wunderlich M.C.* On the parallel generation of the residues for the continued fraction factoring algorithm // Math. Comp. 1987. V. 48 (177). P. 405—423.

[286] *Wu H.* Efficient computations in finite fields with cryptographic significance / PhD thesis. Univ. of Waterloo, Waterloo, Ontario, Canada, 1998.

[287] *Wu H.* Montgomery multiplier and squarer in $GF(2^n)$ / Technical report, Univ. of Waterloo, The Centre for applied cryptographic research, May 2000.

[288] *Wu H.* On computation of polynomial modular reduction / Technical report, Univ. of Waterloo, The Centre for applied cryptographic research, June 2000.

[289] *Wu H.* On modular reduction / Technical report, Univ. of Waterloo, The Centre for applied cryptographic research, June 2000.

[290] *Zayer J.* Factorisieren mit dem Number Field Sieve / PhD thesis, Universität der Saarlandes, 1995.

[291] *Zierler N.* A conversion algorithm for logarithms on $GF(2^n)$ // J. Pure and Appl. Algebra. 1974. V. 4. P. 353—356.

[292] *Zuras D.* On squaring and multiplying large integers // Proceedings of 11th IEEE Symp. Comp. Arith. IEEE Press, 1993. P. 260—271.

Предметный указатель

Олег Николаевич Василенко

Теоретико-числовые алгоритмы в криптографии

Редактор Т. Л. Коробкова

Издательство Московского Центра
непрерывного математического образования

Лицензия ИД № 01335 от 24.03.2000 г.

Подписано в печать 07.10.2003 г. Формат $60 \times 90\ 1/16$. Бумага офсетная № 1.
Печать офсетная. Усл. печ. л. 20.5. Тираж 1000 экз. Заказ № .

МЦНМО
119002, Москва, Большой Власьевский пер., 11

Отпечатано во ФГУП «Производственно-издательский комбинат ВИНИТИ».
140010, г. Люберцы Московской обл., Октябрьский пр-т, 403. Тел. 554 21 86

Книги издательства МЦНМО можно приобрести в магазине «Математическая книга»,
Большой Власьевский пер., д. 11. Тел. 241 72 85. E-mail: biblio@mccme.ru