ОТ ПЕРЕВОДЧИКА

Компьютерная алгебра является одной из областей математики и информатики, особенно активно развивающейся в последние годы. Усилия специалистов в этой области направлены как на разработку новых алгоритмов, так и на создание систем компьютерной алгебры, которые все шире используются и в научных исследованиях, и в практических приложениях. Полученные результаты находят отражение не только в периодической печати, но и в монографиях, опубликованных в последние годы.

В монографии американского ученого греческого происхождения А. Акритаса наряду с фундаментальными результатами в области компьютерной алгебры рассматриваются ее прикладные аспекты, в частности теория кодирования. Автор уделяет большое внимание историческим исследованиям, касающимся рассматриваемых проблем; он отдает должное как древнегреческим математикам, так и ученым сравнительно недавнего прошлого, в частности Доджсону, более известному современному читателю под псевдонимом Льюис Кэрролл. Особо следует отметить большое количество задач, приведенных в книге, что позволяет весьма успешно использовать ее в учебном процессе. Собственно, книга появилась на свет в результате преподавания автором в течение ряда лет курса компьютерной алгебры в Канзасском университете.

Во время пребывания автора в Московском университете в 1990 г. в качестве стипендиата Фонда Фулбрайта и моего пребывания в таком же качестве в Канзасском университете в 1993 г. мы подробно обсуждали различные разделы книги. Автор предоставил для русского издания список замеченных опечаток и, что особенно ценно, обширный список дополнительных задач, за что я пользуюсь случаем выразить ему глубокую благодарность. Некоторые доказательства по сравнению с английским оригиналом изменены, сделаны более короткими или более строгими.

Пользуюсь случаем выразаить мою глубокую благодарность В.В. Борисенко, выполнившему перевод гл. 2.

Надеюсь, что данная книга будет представлять интерес для широкого круга читателей, как для разработчиков алгоритмов и систем компьютерной алгебры, так и для многочисленных пользователей, применяющих системы компьютерной алгебры в своей работе.

Е.В. Панкратьев

ПРЕДИСЛОВИЕ К РУССКОМУ ИЗДАНИЮ

Мне доставляет большое удовольствие быть свидетелем возрастающего интереса к компьютерной алгебре во всем мире и особенно в бывшем Советском Союзе. Мне посчастливилось провести весенний семестр 1990 г. в МГУ в качестве фулбрайтского стипендиата, и я встречался лично и установил деловые контакты с большинством моих советских коллег. Их энтузиазм, результаты и возможности дальнейшего вклада в рассматриваемую область впечатляют, и я надеюсь, что данная книга принесет некоторую пользу в этом деле. Я хотел бы воспользоваться случаем поблагодарить тех из моих русских коллег, кто своим конструктивным критицизмом способствовал улучшению этого издания.

Алкивиадис Г. Акритас

ПРЕДИСЛОВИЕ

Памяти моего отца

Компьютер прежде всего является устройством для обработки информации, а что представляет из себя эта информация — не столь важно. Компьютеры нужно было «научить» выполнять арифметические операции, чтобы они стали «переваривать» *числа*. Деловой мир породил текстовый процессор, который «переваривает» *слова*. А теперь существует и возможность «переваривать» *символы* — компьютерная алгебра. Имея дело в основном с точными числами и алгебраическими выражениями в их символьном представлении, системы компьютерной алгебры могут помочь ученым лучше представить себе *изнутри* различные рассматриваемые физические явления.

Компьютерная алгебра отличается от численного анализа, где упор делается на ошибки, которые могут появиться при выполнении некоторого алгоритма. Эти ошибки, усечение и округление, возникают из-за использования арифметики с плавающей точкой одинарной и двойной точности. В общем, чем меньше итоговая ошибка, тем лучше алгоритм.

Начиная с 1960 г. было разработано много программных систем, предназначенных для различного рода символьных вычислений; эффективность и возможности этих систем постоянно возрастают и в будущем можно ожидать расширения их использования. Операции над полиномами и рациональными функциями составляют основу любой системы символьных преобразований, поэтому исследования в этой области включают в себя развитие и анализ эффективных алгоритмов для разложения на множители, вычисления наибольших общих делителей и отделения вещественных корней полиномов.

Компьютерная алгебра включает в себя большое количество различных тем, а поскольку она до настоящего времени находится в стадии развития, к имеющемуся списку тем постоянно добавляются новые. В книге нашли отражение только те темы, которые автор считает «классическими», которые можно использовать как справочное пособие для исследователей и в качестве руководства при обучении численным методам, информатике или математике. Поскольку в курсах численного анализа и компьютерной алгебры существует очень мало общих тем (если таковые вообще имеются), эта книга может послужить основой курса, дополнительного или альтернативного курсу численного анализа.

При преподавании математики, с одной стороны, эта книга дает прекрасное средство обучения как теории, так и приложениям алге-

бры и эффективного соединения традиционной алгебры с информатикой. С другой стороны, студент, специализирующийся в области информатики, изучая компьютерную алгебру, использует большое число понятий, освоенных в предыдущих курсах, и осознает красоту работ некоторых гигантов-математиков предыдущих столетий (Галуа, Гензеля, Лагранжа, Штурма, Сильвестра и Винсента, если перечислить лишь немногих из них), чей фундаментальный подход к вычислениям *не может* уложиться в рамки численного анализа, но весьма напоминает то, что современные исследователи пытаются сделать, используя компьютерную алгебру.

Основные результаты, впервые опубликованные в этой книге

(1) Лучшая версия метода вычисления субрезультантных последовательностей полиномиальных остатков, разработанная автором в 1986 г. на основе статьи Сильвестра (Sylvester, 1853).
(2) Показаны значимость теоремы Бюдана и ее связь с теоремой Фурье.
(3) Самый быстрый из существующих методов отделения вещественных корней полиномиального уравнения, разработанный автором в 1978 г. на основе теоремы Винсента (Vincent, 1836).

Чего нет в этой книге

В этой книге главное — не доказательства; доказательства включены только в тех случаях, когда они помогают лучшему пониманию материала или их можно найти лишь в научных журналах. Эта книга и не о структурах данных; на ее основе может быть разными способами выполнена реализация различных алгоритмов, описанных в ней, но это оставлено преподавателю и изобретательности студентов.

Основой для данной книги послужил курс «Компьютерная алгебра» для студентов старших курсов и аспирантов, который я читал как в Канзасском университете, США, так и в Национальном техническом университете Афин, Греция. Курс был успешно воспринят студентами, специализирующимися в области информатики, математики, электротехники и вычислительной техники. Что касается предварительной подготовки — *желательно*, но не обязательно прослушать хороший курс по структурам данных, а также курс по современной и/или линейной алгебре.

Материал в данной книге разделен на три части:

(1) Часть I является вводной и состоит из гл. 1, объясняющей, что такое компьютерная алгебра.

(2) Часть II содержит основные математические результаты и базисные алгоритмы. Поскольку компьютерная алгебра имеет дело в основном с целыми числами и полиномами с целыми коэффициентами, в гл. 2 описываются основные свойства целых чисел, а в гл. 3 — полиномов.

(3) В части III мы находим приложения идей, развитых в предыдущих частях, а также более специальные разделы, а именно гл. 4 посвящена кодам, исправляющим ошибки, и криптографии, гл. 5 — вычислению полиномиальных наибольших общих делителей и последовательностей полиномиальных остатков, гл. 6 — разложению на неприводимые множители полиномов с целыми коэффициентами, а гл. 7 — отделению и аппроксимации вещественных корней полиномиальных уравнений.

Отбор преподавателем материала имеет очень большое значение, поскольку я обнаружил, что даже не приближаюсь к тому, чтобы за семестр охватить весь этот материал. Разумный подход состоит в том, чтобы детально разобрать ч. I (введение) и II (основные математические результаты и базисные алгоритмы), а затем некоторые разделы из ч. III (приложения и специальные разделы), насколько позволит время, имея в виду, что разд. 7.2 зависит от разд. 5.2. Наконец, *настоятельно* рекомендуется в таком курсе использовать систему компьютерной алгебры **maple**.

Я благодарен моему редактору Марии Тейлор и всему коллективу издательства John Wiley за наше чудесное сотрудничество.

Я благодарю Замира Бавеля, моего коллегу и друга из Канзасского университета за ценные советы, Манолиса Протонотариоса, председателя Отделения электротехники и вычислительной техники Национального технического университета Афин, за предоставленную мне возможность прочитать курс компьютерной алгебры в Греции и мою мать, взявшую на себя заботу обо всем во время моей работы над книгой в Греции.

Алкивиадис Г. Акритас
Канзасский университет

Часть I
ВВЕДЕНИЕ

Эта часть книги — о компьютерной алгебре и системах компьютерной алгебры; основные понятия алгоритмов и их сложности представлены в ней наряду с понятиями структур данных. Разъясняется различие между компьютерной алгеброй и численным анализом.

В численном анализе вещественные числа аппроксимируются числами с плавающей точкой, поскольку внутреннее устройство большинства компьютеров ориентировано на работу с числами, состоящими не более чем из 10 десятичных цифр. Это приводит к неточным вычислениям, которые выполняются очень быстро, поскольку арифметические операции реализованы на аппаратном уровне.

Компьютерная алгебра имеет дело в основном с целыми числами произвольной точности, употребляя соответствующие структуры данных; это приводит к безошибочным вычислениям, которые выполняются несколько медленнее, так как арифметические операции должны быть реализованы в программном обеспечении.

Глава 1

Что такое компьютерная алгебра?

Термин *компьютерная алгебра* (или *символьные и алгебраические вычисления*) объясняется способностью компьютеров манипулировать математическими выражениями, заданными символьно, а не численно, подобно тому, как это делается в алгебре при помощи карандаша и бумаги. Имея дело главным образом с точными числами (целыми и рациональными числами бесконечной точности) и алгебраическими выражениями в их символьном представлении, системы компьютерной алгебры могут освободить ученых от утомительной рутинной работы, связанной с численными ошибками (усечение и округление), и, таким образом, помочь им глубже понять различные изучаемые физические явления — «цель вычислений в проникновении в суть, а не в цифрах» (согласно Р. Хэммингу). Это проникновение в суть достигается иногда при вычислении значений математических выражений, но во многих случаях при использовании алгебраических средств соотношения между величинами становятся яснее.

Компьютерная алгебра применяется к широкому кругу проблем. Рассмотрим, например, теорию гравитации, где исследуются возможные варианты общей теории относительности. Для того чтобы согласовываться с экспериментальными данными, эти варинанты должны удовлетворять теоретическому критерию Биркгофа; компьютерная алгебра является подходящим средством для применения этого критерия. Другой пример можно взять из неврологии, где система уравнений моделирует распространение сигнала по нерву. При некоторых условиях эти уравнения могут порождать повторяющиеся серии сигналов, так называемые *wamn*. Для того чтобы проверить устойчивость волнового пакета, достаточно вычислить знак некоторого математического выражения. Получить само это выражение вручную — кропотливый труд, но с использованием компьютерной алгебры это становится рутинным вычислением (Pavelle et al., 1981).

1.1

Компьютерная алгебра и численный анализ

Прежде чем обсуждать точную арифметику, посмотрим более пристально на неотъемлемые недостатки численных расчетов с использованием компьютеров. Читатель должен помнить, что компьютер — это машина с конечной памятью, состоящей из слов конечной длины; обычно длина компьютерного слова составляет 16 или 32 бита, при этом максимальное целое число, которое можно разместить в слове, составляет $2^{16}-1$ или $2^{32}-1$, что соответствует пятизначным или десятизначным числам в десятичной системе счисления.

При выполнении численных расчетов на компьютере мы обычно сталкиваемся с проблемой представления *бесконечного* множества вещественных чисел в компьютере с конечной памятью и данной длиной слова. Наиболее распространенный способ решения этой проблемы в численном анализе — приближать вещественные числа, используя *конечное* множество чисел с плавающей точкой. Множество F чисел с плавающей точкой характеризуется основанием счисления β, точностью t и областью значений экспоненты $[L,U]$, где параметры β, t, L и U явным образом зависят от компьютера. Каждое число с плавающей точкой f из множества F может быть представлено в виде

$$f = \pm\left(\frac{d_1}{\beta}+\frac{d_2}{\beta^2}+\cdots+\frac{d_t}{\beta^t}\right)\beta^e,$$

где целые числа $d_i,\ i=1,2,\ldots,t$, удовлетворяют неравенствам $0 \le d_i \le \beta-1$ и $L \le e \le U$; если мы потребуем, чтобы $d_1 \neq 0$ для всех $f \in F,\ f \neq 0$, то будем иметь дело с *нормализованными* числами с плавающей точкой.

Следует отметить, что при использовании чисел с плавающей точкой (или целых чисел, помещающихся в одном компьютерном слове) арифметические операции $+$, $*$ и т.д. выполняются очень быстро. Это происходит потому, что эти операции выполняются не программным обеспечением, а электронными схемами компьютера; мы говорим, что арифметические операции $+$, $*$ и т.д. реализованы *аппаратно*.

Внимательный читатель будет теперь иметь представление, какого рода проблемы возникают из такого приближения вещественных чисел числами с плавающей точкой. Прежде всего множество F не является непрерывным или даже бесконечным множеством. В множестве F существует в точности $2(\beta-1)\beta^{t-1}(U-L+1)+1$ нормализованных чисел с плавающей точкой (включая нуль); более того,

эти числа распределены равномерно не на всей области значений, а только между последовательными степенями β. В качестве примера рассмотрим 33-точечное множество F при $\beta = 2$, $t = 3$, $L = -1$ и $U = 2$, которое показано на рис. 1.1.1.

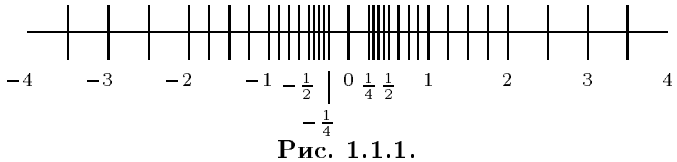


Рис. 1.1.1.

Система чисел с плавающей точкой при $\beta = 2$, $t = 3$, $L = -1$ и $U = 2$. (Forsythe, G.E., M.A. Malcolm, C.B. Moler. Computer methods for mathematical computations, 1977, p. 12. Воспроизведено с разрешения издательства Prentice-Hall, Inc., Englewood Cliffs, New Jersey.)

Из сказанного выше следует, что сумма (или произведение) данных чисел f_1 и f_2 из множества F может *не* принадлежать F и должна быть приближена ближайшим числом с плавающей точкой. Разность между истинным и приближенным значением суммы (или произведения) является ошибкой *округления*. Следует также отметить, что операции сложения и умножения в F не являются ассоциативными и закон дистрибутивности тоже не выполняется. Рассмотрим, например, в нашем игрушечном 33-точечном множестве F выражение $5/4 + (3/8 + 3/8) = 2$, где числа $5/4$, $3/8$ и 2 принадлежат F. Однако в этом выражении $(5/4 + 3/8) + 3/8 \neq 2$, поскольку сумма $(5/4 + 3/8)$ не принадлежит F и должна быть приближена либо числом $3/2$, либо числом $7/4$. Ошибки округления встречаются не только при использовании чисел с плавающей точкой; они могут возникнуть и при работе с целыми числами, например в случае, когда мы хотим вычислить произведение двух s-значных чисел в компьютере, который не может обрабатывать числа, содержащие больше s цифр.

Таким образом, мы видим, что в численном анализе нужно тщательно оценивать ошибки округления (и вычислять их границы), возникающие при работе любого алгоритма, а не фокусировать все внимание на самом алгоритме и его эффективности. Это вместе с тем

фактом, что храниться в памяти и обрабатываться могут только численные значения математических выражений, указывает на необходимость программных систем, способных обрабатывать выражения в символьном виде и производить безошибочные вычисления (именно таким образом «родилась» компьютерная алгебра). Как мы сейчас (в разд. 1.2) убедимся, эти системы избегают чисел с плавающей точкой и работают с целыми числами произвольной точности, используя соответствующие структуры данных. В разд. 1.3 мы укажем некоторые из многочисленных имеющихся систем и кратко опишем возможности двух из них.

1.2

Компьютерная алгебра: точная целочисленная и полиномиальная арифметики

Как отмечалось выше, для того чтобы иметь возможность представлять целые числа произвольной точности и выполнять точные арифметические операции, нам необходимо ввести (если их еще нет) соответствующие структуры данных (или воспользоваться модулярной арифметикой, как мы увидим в гл. 2) и избегать чисел с плавающей точкой; это и делается в компьютерной алгебре. В этом разделе мы вводим вычислительную модель для целочисленного и полиномиального представлений и анализируем некоторые алгоритмы «с карандашом и бумагой» выполнения над ними арифметических операций. Очевидно, что для представления целых чисел и полиномов можно пользоваться массивами, но они не являются «динамическими» структурами данных; мы будем обсуждать только их представление в виде списков (Horowitz et al., 1976).

Списки и базисные операции над списками. Прежде всего рекурсивно определим *список* над произвольным множеством S как конечную последовательность $(a_1, a_2, \ldots, a_n)$, $n \geq 0$, в которой каждый элемент a_i является либо элементом множества S, либо списком над S; пустой список представляется символом 0 и соответствует $n = 0$. Когда мы пишем $a = (a_1, a_2, \ldots, a_n)$, мы интерпретируем это двояко: (1) a рассматривается как указатель на начало списка и (2) a представляет весь список, так что, когда мы пишем $a \oplus x$, где $\oplus$ — одна из бинарных операций над скалярами, мы имеем в виду, что операция $\oplus$ должна быть произведена над каждым элементом списка a и скаляром x. (Из контекста всегда будет ясно, что в данный

момент имеется в виду.)

Если дан список $a = (a_1, a_2, \ldots, a_n)$, где a рассматривается как указатель на начало списка, то на нем можно определить различные операции. Для нас представляют интерес следующие: длина$(a) = n$; первый$(a) = a_1$; последний$(a) = a_n$; хвост$(a) = (a_2, a_3, \ldots, a_n)$; развернутый$(a) = (a_n, \ldots, a_1)$; присоединить $b_1, \ldots, b_k$ к a, $k \geq 1$, что дает в результате список $(b_1, \ldots, b_k, a_1, \ldots, a_n)$; отделить $b_1, \ldots, b_k$ от a, $k \leq n$, что приводит к тому, что b_i становятся равными a_i, $1 \leq i \leq k$, и $a = (a_{k+1}, \ldots, a_n)$. Если $a = 0$, пустой список, то мы определяем результат операции «присоединить a_1 к a» как $a = (a_1)$. В дальнейшем списки и списковые элементы списка можно будет легко отличить от элементов множества S.

Списочное представление целых чисел. Для того чтобы иметь возможность хранить в памяти компьютера целые числа произвольной точности и выполнять над ними точные арифметические операции, мы должны представлять их в виде списков. На этом пути, однако, мы утрачиваем возможность выполнять над этими целыми числами аппаратно реализованные операторы $+$, $*$ и т.д.; вместо этого, как будет показано ниже, мы должны разработать специальное программное обеспечение, выполняющее эту работу.

Мы различаем два типа целых чисел: те, которые представляются списками, называются целыми числами *кратной точности* или *длинными* целыми числами, остальные называются целыми числами *одинарной точности* или *короткими* целыми числами. Целые числа первого типа представляются в виде $i = (i_0, i_1, \ldots, i_n)$, $n \geq 1$, где i_j удовлетворяют неравенству $|i_j| \leq \beta - 1$, служат коэффициентами при β^j в выражении $i = \sum_{0 \leq j \leq n} i_j \beta^j$ и являются все неотрицательными или неположительными в соответствии с тем $i > 0$ или $i < 0$; $\beta - 1 = 2^\mu - 1$ — наибольшее значение, хранимое в компьютерном слове.

Каждое i_j хранится в отдельном компьютерном слове и, за исключением i_n, занимает μ бит. Список может быть образован одним из двух способов, а именно, i_n — наиболее значимая β-цифра — может находиться либо в начале, либо в конце списка. Большей частью мы будем следовать второму подходу, в котором порядок обратен естественному представлению целого числа, но такое представление выбирается потому, что большинство арифметических операций выполняется, начиная с цифр младших разрядов; однако при изучении деления длинных целых чисел наиболее значимая β-цифра будет в начале списка. Знак$(i) = \pm 1$, зависит от того будет $i_n > 0$ или $i_n < 0$, т.е. знак числа i «хранится» только в i_n.

Например, предположим, что $\beta = 10^3$, т.е. компьютерное слово может содержать только три десятичные цифры, и что нам нужно запомнить число $i = +23456789$. Мы можем разместить его в памяти, как показано на рис. 1.2.1.

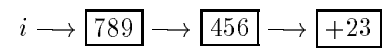


Рис. 1.2.1.

Внутреннее представление целого числа $i = +23456789$ в компьютере, слово которого может содержать только три десятичные цифры.

Стрелки на рис. 1.2.1 указывают, что *звенья*, или *ячейки* (состоящие из одного или нескольких машинных слов) ссылаются на следующие. Ко всей структуре адресуются с помощью переменной i. Аналогично, рациональное число n/d представляется списком $r = (\mathbf{n}, \mathbf{d})$, где $\mathbf{n}$ и $\mathbf{d}$ – целые числа n и d, представленные списками.

Структуры данных. Первое, о чем необходимо позаботиться при работе со ссылочными структурами, — как построить звено, т.е. сколько компьютерных слов используется для одного звена, сколько полей данных мы собираемся иметь в звене и каков должен быть размер этих полей. На рис. 1.2.1 эти характеристики звена опущены, однако разумный выбор может состоять в использовании двух слов для звена (или ячейки), разделенного на три поля, функции которых будут разъяснены ниже: поле типа T, поле элемента E и поле ссылки S, как это показано на рис. 1.2.2.

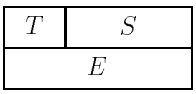


Рис. 1.2.2.

Внутреннее представление *ячейки*, или *звена*.

Рассмотрим список общего вида $(x_1, x_2, \ldots, x_n)$. Компьютерное представление этого списка состоит из n ячеек, связанных через их поля ссылок, вместе с предполагаемыми уже данными представлениями каждого из значений x_i, являющихся в свою очередь списками. Поле ссылки i-й ячейки содержит адрес $(i+1)$-й ячейки $(1 \le i < n)$, а поле ссылки n-й ячейки содержит 0. (Разумеется, предполагается, что 0 не является адресом никакой ячейки.) Поле типа i-й ячейки содержит нуль или единицу в зависимости от того, является x_i атомом или списком. Если x_i является атомом, то поле элемента i-й ячейки

содержит x_i; если x_i — список, то поле элемента i-й ячейки содержит координаты некоторого представления списка x_i. Координаты списка $(x_1, x_2, \ldots, x_n)$ — это адрес первой ячейки его представления. В соответствии с этим внутреннее представление целого числа i, изображенного на рис. 1.2.1, является таким, как показано на рис. 1.2.3.

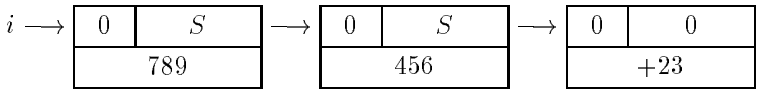


Рис. 1.2.3.

Полное внутреннее списочное представление целого числа на рис. 1.2.1.

Раз такое решение принято, то во время вычислений все неиспользуемые звенья связаны вместе в произвольном порядке и образуют *список свободного места*. Когда требуется новое звено, первое звено списка свободного места отщепляется и присоединяется к соответствующему списку данных. Когда список данных становится не нужным в дальнейших вычислениях, его звенья присоединяются к голове списка свободного места. В упражнениях по программированию 1 и 2 к этому параграфу (в конце главы, перед библиографией) читателю поручается написать некоторые из этих процедур. Как для новичка, так и для опытного программиста представляет интерес техника, позволяющая на языках высокого уровня проверить значение данного поля и/или обновить его, не разрушая имеющейся в данном слове информации. Мы разъясним это на следующем примере, использующем десятичную систему счисления.

Пример. Предположим, что число $n = 123456789$ хранится в машинном слове, разделенном на *три* поля, содержащих 1, 3 и 5 десятичных цифр соответственно, т.е. это число можно представлять себе как 1-234-56789, хотя компьютер воспринимает это девятизначное число как единое целое. Для того чтобы изменить значение центрального трехсимвольного поля с 234, например, на 432, мы должны сделать следующее:

Разделить число $n = 123456789$ на 10^5, получив частное $q_1 = 1234$ и остаток $r_1 = 56789$.

Затем разделить $q_1 = 1234$ на 10^3, получив $q_2 = 1$ и $r_2 = 234$ (в этот момент значение r_2 несущественно).

Затем умножить $q_2 = 1$ на 1000 и прибавить число 432 (которое должно заменить число 234), чтобы получить промежуточный результат $n_p = 1432$.

Наконец, умножить $n_p = 1432$ на 10^5 и прибавить $r_1 = 56789$, чтобы получить число 143256789 в требуемом измененном виде.

Чтобы досконально понять эту технику, читателю требуется самостоятельно выполнить несколько примеров.

Алгоритмы и их сложность. Алгоритм — это метод решения некоторого класса задач. Ресурсы, используемые алгоритмом для решения одной такой задачи, называются *сложностью* алгоритма и измеряются в соответствующих единицах (время счета или используемая память). В данной книге мы интересуемся только *временно́й* сложностью вычислений и будем измерять ее, выражая время вычислений в виде функции от некоторой меры (определенной ниже) количества данных, подаваемых на вход.

Определение 1.2.1. Пусть f и g — вещественнозначные функции, определенные на множестве S. Мы будем говорить, что (1) f *доминируется* функцией g, и писать $f = O(g)$, если существуют положительное вещественное число c_1 и элемент $x_1 \in S$, такие, что $|f(x)| \le c_1|g(x)|$ для всех $x \in S$, $x > x_1$; (2) f *доминирует* g, и писать $f = \Omega(g)$, если g доминируется f, и (3) функции f и g *кодоминантны*, и писать $f \sim g$, если $f = O(g)$ и $f = \Omega(g)$.

Некоторые свойства доминирования и кодоминантности, которые мы будем использовать в дальнейшем, легко следуют из определения.

Определение 1.2.2. *β-длиной* целого числа i мы будем называть число β-цифр в его представлении и будем обозначать ее $L_\beta(i)$. Если $\lceil x \rceil$ — *потолок*, наименьшее целое большее или равное x, а $\lfloor x \rfloor$ — *пол* — наибольшее целое число меньшее или равное x, то

$$L_\beta(i) = \begin{cases} 1, & \text{если } i = 0 \\ \lceil \log_\beta(|i|+1) \rceil = \lfloor \log_\beta |i| \rfloor + 1, & \text{если } i \neq 0. \end{cases}$$

Пример. Целое число $i = +23456789$ на рис. 1.2.1 при $\beta = 10^3$ имеет β-длину $L_\beta(i) = 3$, т.е. длина целого числа — это количество компьютерных ячеек, требующееся для его представления. В дальнейшем индекс β мы будем опускать, поскольку для любого другого основания γ имеем $L_\beta \sim L_\gamma$ (если рассматривать L_β и L_γ как функции, определенные на множестве целых чисел).

Определение 1.2.3. Пусть **A** — алгоритм и S — множество допустимых значений входа для **A**. Целое число $t_{\mathbf{A}}(n)$ для $n \in S$ — число базисных операций, выполняемых алгоритмом **A** при значении входа n, — называется *функцией времени вычислений*, ассоциированной с **A** и определенной на S. К базисным операциям относятся такие, как сложение и умножение одинарной точности, замены, безусловные передачи управления и вызовы подпрограмм.

Формально полиномы определяются в гл. 3; в сформулированной ниже теореме мы предполагаем, что читатель имеет о них интуитивное представление.

Теорема 1.2.4. Если $p(n) = p_m n^m + \cdots + p_1 n + p_0$ — полином степени m, то $p(n) = O(n^m)$.

Доказательство. Полагая $n \geq 1$, имеем

$$|p(n)| \leq |p_m| n^m + \cdots + |p_1| n + |p_0| \leq \left(|p_m| + \cdots + \frac{|p_1|}{n^{m-1}} + \frac{|p_0|}{n^m} \right) n^m$$
$$\leq (|p_m| + \cdots + |p_1| + |p_0|) n^m.$$

Для доказательства теоремы достаточно теперь положить $c = |p_m| + \cdots + |p_1| + |p_0|$. □

В качестве применения теоремы 1.2.4 рассмотрим алгоритм, содержащий k инструкций (или шагов), каждая из которых выполняется за время $c_i n^{m_i}$, $1 \leq i \leq k$. Тогда весь алгоритм выполняется за время $O(n^m)$, где m — максимум чисел m_i, $1 \leq i \leq k$.

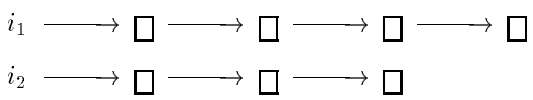
Если функция времени вычислений алгоритма **A** имеет вид $t_{\mathbf{A}}(n) = O(n^3)$ или $t_{\mathbf{A}}(n) = O(n^{13})$ и т.д., где n — размер входа (т.е. если функция времени вычислений является самое большее полиномиальной функцией от величины входных данных), то **A** называется алгоритмом, *полиномиальным по времени*. Существуют также *экспоненциальные по времени* алгоритмы, функции времени вычислений которых являются показательными функциями от количества входных данных, т.е. функциями вида $t_{\mathbf{A}}(n) = O(2^n)$. Очевидно, что

$$O(\log n) < O(n) < O(n \log n) < O(n^2) < O(2^n),$$

и общее правило состоит в том, что вычисление является «легким», если мы имеем дело с полиномиальным по времени алгоритмом, и «сложным», если мы имеем дело с экспоненциальным по времени алгоритмом.

В этой книге мы будем в основном рассматривать границу для *худшего случая*, которая представляет собой максимальное время счета, необходимое для выполнения алгоритма. Другой тип границы — это граница *в среднем*, т.е. просто среднее время счета, получающееся, если время выполнения алгоритма усреднить по всем возможным входным данным.

Классические алгоритмы целочисленной арифметики и их сложность. Рассмотрим теперь классические алгоритмы выполнения арифметических операций с длинными целыми числами и их сложность. Как уже отмечалось, нам нужно разработать программное обеспечение для арифметики целых чисел, поскольку компьютер не может применять аппаратно реализованные операторы $+$, $*$ и т.д. к длинным целым числам; поэтому мы говорим, что теперь арифметические операторы $+$, $*$ и т.д. реализованы *программно*, и очевидно, что они работают медленнее, чем их аппаратно реализованные аналоги. Ниже, при определении функции времени вычислений алгоритма, читатель должен иметь в виду алгоритмы целочисленной арифметики, которые проходят в средней школе; звено, или ячейка, тогда аналогичны десятичной цифре, и нам просто требуется подсчитывать одноразрядные сложения и/или умножения. Различные операции будут выполняться над двумя длинными целыми i_1 и i_2, представленными списками.



Предположим сначала, что мы хотим вычислить сумму чисел i_1 и i_2. С программистской точки зрения мы можем воспользоваться двумя подходами:

1. Написать процедуру, назвав ее **ISUM**, для сложения целых чисел (*i*nteger *sum*mation), на вход которой подаются i_1 и i_2, а на выходе получается значение их суммы s.
2. «Перегрузить» оператор $+$, т.е. когда встречается оператор $+$, проверяется тип переменных, являющихся его аргументами, и если обнаруживаются длинные целые числа, то существует ветвление на процедуру **ISUM**. (Такой подход мы называем «дружественным к пользователю», поскольку в данном случае пользователю не надо запоминать имена всех процедур, которые ему могут потребоваться.)

Эти два подхода применимы к любым другим операциям с целыми числами и в дальнейшем упоминаться явно не будут.

При рассмотрении сложения выход $s = i_1 + i_2$ также является списком, который получается при одновременном сканировании (продвижении вдоль) списков i_1 и i_2 и сложении малых целых чисел соответствующих звеньев (используя аппаратно реализованный +), учитывая, конечно, переносы. (Напомним, что в соответствии с нашими соглашениями о списочном представлении целого числа для сложения и умножения наименее значимые звенья появляются первыми.) Функция времени вычислений имеет вид $t_{\mathbf{ISUM}}(i_1, i_2) = O\{\max[L(i_1), L(i_2)]\}$; это легко следует из того, что мы считаем число сложений одинарной точности и что их имеется не более чем $\max[L(i_1), L(i_2)]$ (эта же величина ограничивает число возможных переносов) (см. определение 1.2.3). Кроме того, число ячеек во вновь созданном списке не превосходит $\max[L(i_1), L(i_2)] + 1$.

В качестве упражнения мы оставляем читателю проверку того, что, используя школьные алгоритмы целочисленного умножения для вычисления произведения $i_1 \cdot i_2$ (двух упомянутых выше длинных целых чисел), получим $t_{\mathbf{IMULT}}(i_1, i_2) = O[L(i_1), L(i_2)]$.

Деление значительно сложнее. При делении i_1 на i_2 мы в действительности ищем целые числа q и r, обладающие свойством делимости с остатком $i_1 = i_2 q + r$, $0 \le r < i_2$. Небольшое размышление над школьным процессом деления длинных чисел показывает, что для его выполнения нам достаточно уметь неоднократно делить $(k+1)$-значное число $m = (m_0, m_1, \ldots, m_k)$ на k-значный делитель $n = (n_1, n_2, \ldots, n_k)$, $n \le m < bn$, где b — основание системы счисления (в компьютерных приложениях $b = \beta$, где β уже использовалось выше). Обычно в компьютерных приложениях b — это 2^{32} или какая-нибудь другая степень 2; в этих рассуждениях мы сначала рассматриваем наиболее значимые цифры, так что $m = m_0 b^k + \cdots + m_k$ и $n = n_1 b^{k-1} + \cdots + n_k$. Например, если нам нужно разделить 1234 на 23, то мы сначала делим 123 (наше начальное m) на 23 (наше n), получаем 5 и 8 в остатке; затем делим 84 (наше новое m) на 23, получаем 3 и 15 в остатке. Очевидно, что эта же идея работает и в общем случае. Наиболее очевидный подход к данной задаче состоит в том, чтобы угадывать частное q по наиболее значимым цифрам чисел m и n; полученное таким путем частное называется *пробным частным* и обозначается q_t. Стандартный процесс угадывания состоит в делении на n_1, наиболее значимую цифру числа n, двузначного числа $m_0 b + m_1$; в качестве q_t берем получающееся частное. Таким образом, определяем

$$q_t = \begin{cases} b-1, & \text{если } n_1 = m_0, \\ \lfloor \frac{m_0 b + m_1}{n_1} \rfloor, & \text{если } n_1 > m_0, \end{cases}$$

где в любом случае $q_t \le b-1$ и $q_t n_1 \le m_0 b + m_1$. (Почему в этих рассуждениях не может быть $n_1 < m_0$?)

Пример. Обозначим истинное частное через q. Тогда при $b = 10$ им имеем:

Если $n = 69$ и $m = 600$, то $n_1 = m_0$, а следовательно, $q_t = 9$. В этом случае $q = 8$.

Если $n = 69$ и $m = 480$, то $q_t = 48/6 = 8$. В этом случае $q = 6$.

Если $n = 29$ и $m = 200$, то $n_1 = m_0$, а следовательно, $q_t = 9$. В этом случае $q = 6$.

Для каждого рассмотренного выше случая выполняются неравенства $n \le m < bn$, не позволяющие нам рассматривать случаи типа $n = 59$ и $m = 600$ или даже $n = 60$ и $m = 600$. Ограничивает ли это общность? (Для ответа на этот вопрос см. теорему 1.2.5 и следующие за ней комментарии.) Кроме того, отметим, что во всех рассмотренных случаях q_t слишком велико, однако при $n = 69$ угаданное значение не так плохо, как при $n = 29$. Почему это так, объясняется следующей теоремой.

Теорема 1.2.5. Пусть b — основание системы счисления, и рассмотрим числа $m = m_0 b^k + \cdots + m_k$ и $n = n_1 b^{k-1} + \cdots + n_k$, $n \le m < bn$. Если мы обозначим через q_t и q (оба — целые числа) пробное частное и частное соответственно при делении m на n, то $q_t \ge q$; более того, если $n_1 \ge b/2$, то $q_t - 2 \le q \le q_t$; это значит, что q_t равно либо q, либо $q+1$, либо $q+2$.

Доказательство. В течение доказательства мы будем помнить, что $m = m_0 b^k + m_1 b^{k-1} + m_2 b^{k-2} + \cdots + m_k$ и $n = n_1 b^{k-1} + n_2 b^{k-2} + \cdots + n_k$, $n \le m < bn$.

Из $qn \le m$, используя неравенство $m_2 b^{k-2} + \cdots + m_k < b^{k-1}$, мы получаем, что $qn_1 b^{k-1} < (m_0 b + m_1 + 1) b^{k-1}$, следовательно, $qn_1 \le m_0 b + m_1$, где $q \le b-1$. По определению, однако, q_t равно либо $b-1$, либо, если $n_1 > m_0$, наибольшему кратному числа n_1, которое $\le m_0 b + m_1$. Ясно, что $q_t \ge q$.

Для доказательства второй части теоремы предположим, что $n_1 \ge b/2$; достаточно в этом случае показать, что $(q_t - 2)n \le m$. Пользуясь неравенством $n_2 b^{k-2} + \cdots + n_k < b^{k-1}$, получаем

$$\begin{aligned}(q_t - 2)n &< (q_t - 2)(n_1 + 1)b^{k-1} = [q_t n_1 + (q_t - 2 - 2n_1)]b^{k-1} \\ &\le (m_0 b + m_1)b^{k-1} + (q_t - 2 - 2n_1)b^{k-1}\end{aligned}$$

по определению q_t. Поскольку $n_1 \ge b/2$ и $q_t \le b-1$, имеем $q_t - 2 - 2n_1 < 0$, и правая часть этого соотношения $\le (m_0 b + m_1)b^{k-1} \le (m_0 b + m_1)b^{k-1} + m_2 b^{k-2} + \cdots + m_k = m$. Таким образом, вторая часть теоремы доказана. $\square$

Чтобы добиться выполнения условия, что старшая цифра делителя $\geq b/2$, нам нужно *нормализовать* его, т.е. домножить m и n на 2^e, где 2^e – наибольшая степень 2, для которой $2^e \cdot n < b^{k+1}$. Затем делим $2^e \cdot m$ на $2^e \cdot n$. Для демонстрации рассмотрим последний случай предыдущего примера, в котором $b = 10$, $n = 29$ и $m = 200$. Вычислим наибольшее e, такое, что $2^e \cdot 29 < 1000$; получаем $e = 5$ и нормализованные значения n и m равны 928 ($= 32 \cdot 29$) и 6400 ($= 32 \cdot 200$) соответственно. Нормализация не влияет на частное, однако нам следует разделить остаток на 2^e.

Отметим, что в теореме 1.2.5 значение основания b несущественно. К тому же мы легко можем при необходимости подправить q_t на 1 или 2, чтобы получить правильное частное на каждом шаге длинного деления. Более того, как мы увидим в следующем примере, можно изменить нашу стратегию угадывания и использовать большее количество старших цифр как в m, так и в n.

Пример. Разделим 272828282 на 3242. Получим

2728282 | **32**42		$q_t = \lfloor 272/32 \rfloor = 8$
25936 | 84154		
13468		$q_t = \lfloor 134/32 \rfloor = 4$
12968		
5002		$q_t = \lfloor 50/32 \rfloor = 1$
3242		
17608		$q_t = \lfloor 176/32 \rfloor = 5$
16210		
13982		$q_t = \lfloor 139/32 \rfloor = 4$
12968		
1014		остаток

Заметим, что мы использовали две или три старшие цифры делимого.

Такой метод угадывания всегда даст нам либо истинную цифру частного, либо цифру, большую ее на единицу. Доказательство этого факта, а также детали реализации мы оставляем читателю в качестве упражнения по программированию. [*Указание.* См. упр. 19–21 в книге (Knuth, 1981, p. 235–238, 246); см. также полный текст программ на Паскале длинной целочисленной арифметики в книге (Flanders, 1984, p. 342–357)].

Читателю в качестве упражнения мы оставляем доказательство того, что, используя предыдущий алгоритм с карандашом и бумагой,

получаем $t_{\mathbf{IDIV}}(i_1, i_2) = O[L(i_2)\{L(i_1) - L(i_2) + 1\}]$; т.е. время деления i_1 на i_2 $(i_1 \geq i_2)$, сводящегося к вычислению q и r, обладающих свойством делимости, по существу совпадает с временем вычисления произведения $i_2 \cdot q$; см. также упр. по программированию 4 для данного раздела.

Списочное представление полиномов. Обратим теперь наше внимание на полиномы с целыми коэффициентами. Существует несколько способов представлять полином от одной переменной $p(x)$ степени n [и уравнение $p(x) = 0$] в компьютере; мы будем представлять его упорядоченным списком $p = (x, c_r, e_r, c_{r-1}, e_{r-1}, \ldots, c_1, e_1)$, $r \geq 1$, где присутствуют только целые коэффициенты $c_i \neq 0$, представленные списками $c_i = (c_{i1}, c_{i2}, \ldots, c_{im_i})$, $m_i \geq 1$; показатели e_i располагаются в порядке убывания $e_r > e_{r-1} > \cdots > e_1$. Степень полинома $p(x)$ равна $n = e_r$, и мы считаем, что знак полинома $p(x)$ совпадает со знаком c_r. [Другой способ представления полинома $p(x)$ степени $n \geq 0$ состоит в использовании списка $p = (x, n, c_n, c_{n-1}, \ldots, c_0)$; в этом случае включаются нулевые коэффициенты. Мы будем пользоваться первым представлением.]

Как и в случае списочного представления целых чисел, наши ячейки (или звенья) снова будут состоять из двух компьютерных слов с теми же самыми тремя полями. Например, полином $p(x) = x^3 - 7$ можно представить так, как показано на рис. 1.2.4.

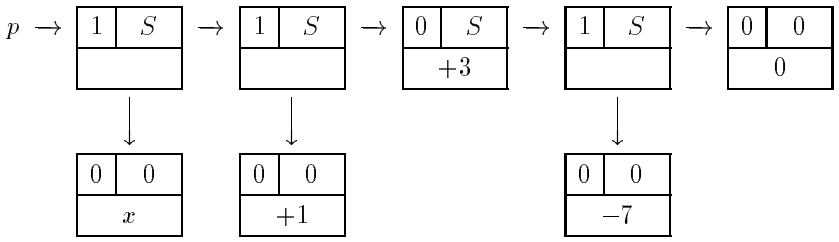


Рис. 1.2.4.

Внутреннее списочное представление полинома от одной переменной $x^3 - 7$; x обозначает численное значение, используемое компьютером для внутреннего представления переменной x. Отметим, что коэффициенты могут иметь произвольную длину, в то время как показатели степеней — целые числа одинарной точности.

Пустой список представляет полином $p(x) = 0$. Полиномы от

многих переменных над целыми числами могут быть представлены в рекурсивной канонической форме, т.е. полином от v переменных $x_1, x_2, \ldots, x_v$ рассматривается как полином от одной переменной x_v с коэффициентами c_i, являющимися полиномами от $v-1$ переменных $x_1, x_2, \ldots, x_{v-1}$.

Читателю в качестве упражнения оставляется задача изобразить внутреннее представление полиномов от многих переменных.

Классические алгоритмы полиномиальной арифметики и их сложность. Границы времени вычислений для операций над полиномами обычно даются в виде функций от степеней и длин норм полиномов.

Определение 1.2.6. Пусть $p(x) = \sum_{0\le i\le n} c_i x^i$ — полиномиальное уравнение от одной переменной с целыми коэффициентами. (Если коэффициенты являются рациональными числами, то мы прежде всего превратим их в целые, домножив каждый из них на наименьшее общее кратное знаменателей.) *Максимальная норма* (или норма с нижним индексом ∞) полинома $p(x)$ — это $|p(x)|_\infty = \max_{0\le i\le n}(|c_i|)$, *суммарная норма* (или норма с нижним индексом 1) — это $|p(x)|_1 = \sum_{0\le i\le n}|c_i|$, и *евклидова норма* — это $|p(x)|_2 = (\sum_{0\le i\le n}|c_i|^2)^{1/2}$.

Традиционно при временно́м анализе алгоритмов, имеющих на входе полиномы, используются только две первые нормы, в то время как евклидова норма используется в доказательстве теорем общего вида. Из определения следует, что $|p(x)|_\infty \le |p(x)|_1 \le (n+1)|p(x)|_\infty$, где n — степень полинома $p(x)$; поэтому $L[|p(x)|_\infty] \sim L[|p(x)|_1]$.

Пусть $p_1(x) = \sum_{0\le i\le m} c_i x^i$ и $p_2(x) = \sum_{0\le i\le n} d_i x^i$ — два полинома с целыми коэффициентами степеней m и n соответственно. Мы хотим получить ограничения в виде функций степеней m и n и длин $L[|p_1(x)|_\infty]$, $L[|p_2(x)|_\infty]$ времен, требуемых для вычисления $p_1(x) \pm p_2(x)$, $p_1(x)\cdot p_2(x)$ и $q(x)$, $r(x)$, обладающих свойством делимости $p_1(x) = p_2(x)q(x) + r(x)$, где степень полинома $r(x) < n$ (предполагая, конечно, что $m \ge n$).

Давайте вычислим, пользуясь описанным выше представлением полиномов, функцию времени вычислений для программы **PSUM**, выполняющей сложение полиномов (*p*olynomial *sum*mation); эта программа берет в качестве входа полиномы $p_1(x)$ и $p_2(x)$ и возвращает их сумму $p_1(x) + p_2(x)$ в виде нового списка. [Более точно, $2[\max(m,n)+1]$ — это длина нового списка, представляющего $p_1(x)+p_2(x)$.] Прежде всего мы видим, что должно быть выполнено не более $\max(m,n)+1$ сложений коэффициентов. Затем напомним, что

для любых двух длинных целых чисел i_1 и i_2 имеем $t_{\mathbf{ISUM}}(i_1, i_2) = O\{\max[L(i_1), L(i_2)]\}$. В нашем случае в худшей возможной ситуации коэффициенты полинома $p_1(x)$ все будут равны $|p_1(x)|_\infty$ [максимальному коэффициенту полинома $p_1(x)$], в то время как коэффициенты полинома $p_2(x)$ все будут равны $|p_2(x)|_\infty$ [максимальному коэффициенту полинома $p_2(x)$]; таким образом, одно сложение коэффициентов выполняется за время $O\{\max[L|p_1(x)|_\infty, L|p_2(x)|_\infty]\}$. Поскольку имеется не более $\max(m, n) + 1$ сложений коэффициентов, легко видеть, что функция времени вычислений этой программы равна

$$t_{\mathbf{PSUM}}[p_1(x), p_2(x)] = O([\max(m, n) + 1] \max\{L[|p_1(x)|_\infty], L[|p_2(x)|_\infty]\}).$$

В качестве упражнения читателю мы оставляем доказательство того, что время, необходимое для вычисления произведения $p_1(x) \cdot p_2(x)$, равно

$$t_{\mathbf{PMULT}}[p_1(x), p_2(x)] = O\{(m + 1)(n + 1)L[|p_1(x)|_\infty]L[|p_2(x)|_\infty]\}.$$

Более того, если $p_1(x)$ — делитель, а $p_2(x)$ — частное, то последнее выражение ограничивает время, необходимое для выполнения обычного алгоритма деления полиномов с целыми коэффициентами. (Алгоритмы полиномиального деления будут представлены в следующих главах.)

1.3

Системы компьютерной алгебры

Имеется несколько доступных систем компьютерной алгебры, большинство из которых разработано в США; в СССР имеется система, называемая **аналитик**, реализованная аппаратным образом. Прекрасным источником информации об этих системах являются периодически проводимые конференции, симпозиумы и т.д. по символьным и алгебраическим преобразованиям, SYMSAC — в США и EUROCAL или EUROSAM — в Европе[1)]. Труды этих конференций содержат как обзоры текущего состояния, так и направления дальнейшего развития; см. например, (Petricle (ed), 1971). Кроме того, ежеквартальный бюллетень публикует группа SIGSAM (*S*pecial *I*nterest *G*roup on

[1)] Начиная с 1988 г. вместо этих конференций под эгидой SIGSAM ежегодно проводится Единая международная конференция, получившая название ISSAC. — *Прим. перев.*

*S*ymbolic and *A*lgebraic *M*anipulation) ассоциации ACM (*A*ssociation for *C*omputing *M*achinary). Также ежеквартально выходит *Journal of Symbolic Computation*.

Стандартным тестом для любой системы компьютерной алгебры является вычисление Делоне движения Луны. Нахождение точного положения Луны в любой данный момент чрезвычайно важно для навигации и астрономии и удивительно сложно; это вычисление было начато в 1847 г. французским астрономом Делоне и заняло у него 10 *лет, плюс* еще 10 лет заняла проверка. Полученная формула занимает целую книгу. В 1970 г. три исследователя лаборатории Боинг в Сиэтле проверили работу Делоне на компьютере, что заняло 20 *часов*. Они обнаружили только три незначительные ошибки, что кажется почти невероятным.

Системы компьютерной алгебры демонстрируют великую изощренность и разнообразие проектов и могут быть разбиты на две основные группы в соответствии с их развитием.

Системы первой группы можно рассматривать как программы *специального назначения*, т.е. они разработаны для решения конкретных задач в различных областях, таких, как математика, теоретическая физика и химия. Все эти системы работают с относительно высокой скоростью, поскольку программа специального назначения может быть настроена на ожидаемый тип входных данных. В следующем списке приводятся примеры таких систем: **camal** — британская система для небесной механики и общей теории относительности; **schoonship** — для физики высоких энергий; **altran, sac-1** и **sac-2** — для полиномиальной арифметики.

Из чисто сентиментальных побуждений мы кратко опишем возможности системы **sac-1** (*s*ymbolic and *a*lgebraic *c*alculation–версия *1*), являющейся высоко переносимой системой.

Sac-1 — переносимая система на базе Фортрана для выполнения разнообразных заданий на многих различных алгебраических структурах. В действительности это — иерархия подсистем или модулей, состоящих из некоторого числа подпрограмм, выполняющих последовательность заданий. Каждый модуль непосредственно зависит от одного или нескольких (а неявно может зависеть от многих) предшествующих модулей системы, кроме стоящей особняком системы обработки списков. Следующий список указывает взаимозависимость различных подсистем (т.е. из приведенного ниже списка видно, например, что полиномиальная система зависит как от системы обработки списков, так и от системы целочисленной арифметики): (1)

обработка списков; (2) целочисленная арифметика; (3) полиномиальная арифметика; (4) модулярная арифметика; (5) наибольший общий делитель и результант; (6) линейная алгебра, разложение на множители полиномов, рациональные функции, гауссовы полиномы; (7) интегрирование рациональных функций, вещественные нули; (8) вещественные алгебраические числа, комплексные нули.

Чтобы воспользоваться системой **sac-1**, нужно написать программу на Фортране, которая выполняет обычные фортрановские вызовы подпрограмм и функций для обращения к требуемым процедурам системы **sac-1**; следовательно, пользователю, знакомому с Фортраном, не придется учить нового синтаксиса. Имена алгоритмов в **sac-1** указывают на их функции; например, **PSUM** означает сложение полиномов (*p*olynomial *sum*mation), в то время как **ISUM** — сложение целых чисел (*i*nteger *sum*mation).

Системы из второй группы являются программами *общего назначения*, снабжающими пользователя как можно более широкими математическими возможностями. Большая часть из них доступна через различные компьютерные сети. Основные представители этой группы — **macsyma, reduce, schratchpad, maple** (канадская система).

Например, **reduce** — система на базе языка Лисп. Ниже перечислены некоторые из ее возможностей: (1) разложения и упорядочение полиномов и рациональных функций; (2) символьное дифференцирование; (3) символьное интегрирование; (4) подстановки и поиск по образцу; (5) вычисление наибольшего общего делителя двух полиномов; (6) автоматическое и контролируемое пользователем упрощение выражений; (7) полный язык для символьных вычислений, на котором написана сама программа **reduce**.

Новейшей и, по-видимому, лучшей для использования системой компьютерной алгебры является **maple**. Она была разработана в 1980-х гг. и вобрала в себя лучшие черты других систем, разработанных в конце 1960-х гг. Для пользователя имеется язык высокого уровня с современным синтаксисом, более подходящий для описания алгебраических алгоритмов.

Системы компьютерной алгебры общего назначения были разработаны также и для микрокомпьютеров, но обычно они более медленные и менее понятливые, чем их сородичи, спроектированные для больших ЭВМ; наиболее распространенной из таких систем является μ-**math**.

В прошлом системы компьютерной алгебры в общем не получали широкого распространения, во-первых, из-за медлительности, поскольку основные арифметические операции должны были быть в явном виде запрограммированы, а не реализованы на уровне аппаратного обеспечения, а во-вторых, из-за быстрого истощения простран-

ства памяти, отводимого для хранения символьных выражений, в силу роста результирующих и промежуточных вычисляемых выражений. Сегодня, однако, с развитием недорогих компьютеров системы компьютерной алгебры стали значительно более доступными для обучения и исследований. Во время написания этой книги (начало 1987 г.) фирма Hewlett-Packard поставляла карманный калькулятор HP-28c для компьютерной алгебры стоимостью приблизительно 200 долларов.

Упражнения

Раздел 1.2

1. **a.** Предполагая существование алгоритма **En**, возводящего целое число в степень n многократным умножением, используйте обозначение O (определение 1.2.1), чтобы оценить время вычисления функции $t_{\mathbf{En}}(k, n)$.

b. Если алгоритм **F** вычисляет $n!$, умножая сначала 2 на 3, затем результат на 4, затем новый результат на 5 и так до n, то чему равно время вычисления функции $t_{\mathbf{F}}(n)$?

(*Указание.* В обеих частях упражнения воспользуйтесь тем, что число цифр в произведении двух чисел либо равно *сумме* числа цифр в сомножителях, либо на 1 больше этой суммы.)

2. Для суммы первых n кубов имеет место формула

$$\sum_{1 \le i \le n} i^3 = [n(n+1)/2]^2.$$

Пользуясь O-обозначениями, оцените в терминах простой функции от n

a. число базисных (битовых) операций, необходимых для выполнения вычислений в левой части этого равенства;

b. число базисных (битовых) операций, необходимых для выполнения вычислений в правой части этого равенства.

3. Предположим, что в нашем распоряжении имеется чрезвычайно длинный список, содержащий все простые числа по n

включительно. Оцените число битовых операций, необходимых для вычисления произведения всех простых чисел, меньших n. (*Указание.* Воспользуйтесь знаменитой теоремой о простых числах, включенной в разд. 2.3.1. Согласно этой теореме,

$$\lim_{n\to\infty} \varpi(n)/(n/\ln n) = 1,$$

где $\varpi(n)$ представляет число простых чисел, меньших, чем n; для больших n частное $n/\ln n$ аппроксимирует $\varpi(n)$.)

Назначение двух следующих упражнений состоит в том, чтобы познакомить читателя с доступными системами компьютерной алгебры.

4. Вычислим общую формулу для значения суммы первых n целых чисел. Заметим, что она имеет вид

$$1 + 2 + 3 + \cdots + n = a \cdot n^2 + b \cdot n + c$$

и нам нужно найти численные значения констант a, b и c. Для этого последовательно подставим вместо n числа 1, 2 и 3 и решим получившуюся систему трех уравнений.

5. Повторите намеченную выше процедуру, чтобы вычислить общую формулу для значения суммы квадратов первых n целых чисел. (*Указание.* Общая формула имеет вид $1^2 + 2^2 + 3^2 + \cdots + n^2 = a \cdot n^3 + b \cdot n^2 + c \cdot n + d$.)

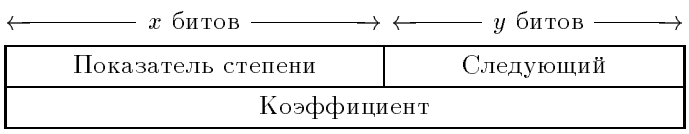
Упражнения по программированию

Раздел 1.2

Эта серия упражнений предназначена только для опытных программистов.

1. Предположим, что вы проектируете ссылочную систему распределения памяти для представления и преобразования полиномов от одной переменной с целыми коэффициентами и что вы решили устроить звено, используя два последовательных машинных слова, одно из которых отводится под показатель степени и поле ссылки, а второе — под поле коэффици-

ента:



Напишите процедуры на языке высокого уровня (без использования логических операций или структурированных записей), которые будут помещать значения в любое поле и читать значения из любого поля, не разрушая содержимое других полей.

2. Установив формат звеньев, напишите процедуры, которые будут (a) инициализировать доступное пространство, т.е. образовывать односвязный список (связанный через поле ссылки) и возвращать указатель, указывающий на первое звено в этом списке (для нашего проекта достаточно 300 звеньев); (b) брать звено из доступного пространства для использования в вычислениях; (c) возвращать звено в доступное пространство, если оно более не нужно; (d) определять длину, т.е. число звеньев, любого списка.

3. Реализуйте классические алгоритмы выполнения арифметических операций с длинными целыми числами.

4. Покажите, что если $i_1 = (a_0, \ldots, a_{n-1})$ и $i_2 = (b_0, \ldots, b_{m-1})$ — два длинных целых числа, таких, что $L(i_1) = n$, $L(i_2) = m$, то $t_{\mathbf{IDIV}}(i_1, i_2) = O[m(n - m + 1)]$. [*Указание.* Покажите, что $L(q) \le n - m + 1$, так что $q = q_{n-m}, \ldots, q_0$ и для определения каждого q_i требуется не более m умножений.]

5. Предположим, что *внешнее* представление полинома с целыми коэффициентами от одной переменной имеет ту же форму, что и его *внутреннее* представление, описанную в этом разделе, т.е. $(x, c_r, e_r, c_{r-1}, e_{r-1}, \ldots, c_1, e_1)$, $r \ge 1$, где члены c_i представляют коэффициенты, а e_i — показатели степеней. Пользуясь процедурами (a)–(d) упр. 2 и предполагая наличие дружественных к пользователю сложения и умножения целых чисел, напишите новые процедуры, которые будут (a) читать входной полином и преобразовывать его во внутреннее ссылочное представление (возвращается указатель на начало этого списка); (b) записывать выходной полином; (c) вычислять сумму $p_1(x) + p_2(x)$ для любых двух заданных полиномов и (d) вычислять произведение $p_1(x) \cdot p_2(x)$ для любых двух заданных полиномов.

6. Пусть $p(x_1, x_2, \ldots, x_v) = \sum_{0 \le j \le n} p_j(x_1, x_2, \ldots, x_{v-1}) \cdot (x_v)^j$ — полином от v переменных с целыми коэффициентами; через

$\vartheta_i[p(x)]$ обозначим его степень по x_i. Ниже мы будем обозначать через $p(\mathbf{x})$ полином $p(x_1, x_2, \ldots, x_v)$ при любом значении v. Индукцией по v определим две нормы $|p(\mathbf{x})|_\infty$ и $|p(\mathbf{x})|_1$ следующим образом: если $v = 0$, то $|p(\mathbf{x})|_\infty = |p(\mathbf{x})|_1 = |p(\mathbf{x})|$, поскольку $|p(\mathbf{x})|$ — целое число. Для $v > 0$ положим $|p(\mathbf{x})|_\infty = \max_{0\le j\le n} |p_j(\mathbf{x})|_\infty$ (наибольший коэффициент) и $|p(\mathbf{x})|_1 = \sum_{0\le j\le n} |p_j(\mathbf{x})|_1$ (сумма коэффициентов).

a. Покажите, что $L[|p(\mathbf{x})|_\infty] \sim L[|p(\mathbf{x})|_1]$.

Составьте алгоритм для работы с карандашом и бумагой, который для двух полиномов $p_1(\mathbf{x})$ и $p_2(\mathbf{x})$ от многих переменных с целыми коэффициентами будет

b. вычислять их сумму $p_1(\mathbf{x}) + p_2(\mathbf{x})$ за время

$$O[\max\{L[|p_1(\mathbf{x})|_\infty], L[|p_2(\mathbf{x})|_\infty]\} \cdot \prod_{1\le i\le v} (\max\{\vartheta_i[p_1(\mathbf{x})], \vartheta_i[p_2(\mathbf{x})]\} + 1)];$$

c. вычислять их произведение $p_1(\mathbf{x}) \cdot p_2(\mathbf{x})$ за время

$$O\big(L[|p_1(\mathbf{x})|_\infty] \cdot L[|p_2(\mathbf{x})|_\infty] \cdot \prod_{1\le i\le v} \{\vartheta_i[p_1(\mathbf{x})] + 1\}\{\vartheta_i[p_2(\mathbf{x})] + 1\}\big).$$

Литература

Flanders H. *Scientific Pascal.* Reston, VA, 1984.

Forsythe G.E., Malcolm M.A., Moler C.B. *Computer methods for mathematical computations.* Prence-Hall, Englewood Cliffs, NJ, 1977.

Horowitz E., Sahni S. *Fundamentals of data structures.* Computer Science Press, Rockville, MD, 1976.

Knuth D. *The art of computer programming.* Vol. 2: *Seminumerical algorithms.* Addison-Wesley, Reading, MA, 1981. [Имеется перевод предыдущего издания: Кнут Д. *Искусство программирования для ЭВМ.* Т. 2. – М.: Мир. 1977.]

Pavelle R., Rothstein M., Fitch J. Computer algebra. *Scientific American.* 136–152, December 1981.

Petricle S.R. (ed.) *Proceedings of the 2nd symposium on symbolic and algebraic manipulation*. Association for Computing Machinary, New York, NY, 1971.

Часть II

МАТЕМАТИЧЕСКИЕ ОСНОВАНИЯ И ОСНОВНЫЕ АЛГОРИТМЫ

Важнейшее значение для любой вычислительной системы имеет лежащая в ее основании алгебраическая структура. Потребовалось время, чтобы математики осознали, что при любых вычислениях важна сама операция, а не только вычисляемые объекты. Хотя многие примеры алгебраических систем были хорошо известны в девятнадцатом столетии, абстрактная алгебра оформилась как самостоятельная наука только к двадцатым годам нашего века, позволив рассмотреть с единой точки зрения и упростить многие теории, которые казались раньше не связанными между собой. В этой части книги вводятся главные алгебраические системы — группы, кольца, поля — и рассматриваются основные алгоритмы для вычислений в этих системах.

В гл. 2 мы рассматриваем основные свойства целых чисел и алгоритмы для работы с ними, в гл. 3 то же самое делается для полиномов. Материал существен для понимания ч. III настоящей книги.

Глава 2

Целые числа

Старейшая область математики — теория чисел, изучение свойств целых (положительных) чисел — оказывается весьма полезной в современном исследовании алгоритмов. Многие известные задачи теории чисел при рассмотрении с алгоритмической точки зрения превращаются в глубокие и притягательные открытые проблемы, причем нерешенной является не проблема вычисления, а проблема быстрейшего вычисления. Кроме того, целые числа очень часто встречаются в различных областях математики, и, изучая алгебру, мы снова сталкиваемся с понятиями, значение которых впервые было замечено в связи с целыми числами; те же понятия позднее оказались полезными в гораздо более общих ситуациях. В этой главе рассматриваются те свойства целых чисел и те алгоритмы, которые нам понадобятся в дальнейшем.

2.1

Основные понятия

Материал этого раздела, без сомнения, знаком читателю. Изложение будет достаточно неформальным (Sims, 1984).

2.1.1. Множества

Алгебраическая система — это множество объектов вместе с набором операций, позволяющих комбинировать эти объекты. Изучаемые нами объекты и операции над ними можно эффективно описывать на языке теории множеств; см. п. 1 раздела «Исторические замечания и литература».

Мы можем рассматривать множество просто как набор некоторых специфических элементов. Если A — множество и x — его элемент, то мы будем писать $x \in A$ и читать это как «x принадлежит A» или «x — элемент из A»; $x \notin A$ читается как «x не принадлежит A». Множество A является *подмножеством* множества B, или A *содержится* в B (мы пишем $A \subset B$), если каждый элемент из A является элементом из B; другими словами, $x \in A$ влечет за собой $x \in B$. Два множества A и B *равны*, $A = B$, если каждый элемент одного

из них является элементом другого и обратно, иначе говоря, $A \subset B$ и $B \subset A$. Множество A — *собственное* подмножество множества B, если $A \subset B$ и $A \neq B$ (не равно). Имеется специальное множество, которое называется *пустым* и обозначается $\emptyset$, не содержащее ни одного элемента, т.е. $x \notin \emptyset$ для любого x. Пустое множество является подмножеством каждого множества, т.е. $\emptyset \subset A$ для любого A.

В нашей неформальной теории множеств мы будем считать, что множество определено, как только указаны его элементы. Мы будем использовать обозначение $\{\, x : P(x) \,\}$ для множества элементов x со свойством $P(x)$. Иногда элементы множества будут просто перечисляться между фигурными скобками $\{\ldots\}$. Множество, состоящее из одного элемента, $\{x\}$, называется *синглетоном*.

Для часто встречающихся числовых множеств имеются специальные обозначения. *Натуральные числа* образуют множество $\mathbb{N} = \{\, 0, 1, 2, \ldots \}$, *целые числа* — множество $\mathbb{Z} = \{\, \ldots, -2, -1, 0, 1, 2, \ldots \}$, а *рациональные числа* составляют множество $\mathbb{Q} = \{\, x/y : x, y \in \mathbb{Z},\ y \neq 0 \,\}$. Через $\mathbb{R}$ мы будем обозначать *вещественные числа; комплексные числа* образуют множество $\mathbb{C} = \{\, x + iy : x, y \in \mathbb{R} \,\}$, где $i^2 = -1$. Мы имеем цепочку включений $\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R} \subset \mathbb{C}$; совокупности ненулевых элементов соответствующих множеств обозначаются через $\mathbb{Z}^*$, $\mathbb{Q}^*$, $\mathbb{R}^*$, $\mathbb{C}^*$. Вещественные числа разделяются на положительные, отрицательные и нуль; $\mathbb{Z}^+$, $\mathbb{Q}^+$, $\mathbb{R}^+$ обозначают совокупности положительных элементов соответствующих множеств.

Над множествами можно выполнять некоторые операции и получать новые множества. *Объединение* $A \cup B$ множеств A и B — это множество $\{\, x : x \in A$ или $x \in B \,\}$. Объединение состоит из элементов, принадлежащих либо A, либо B, либо обоим этим множествам. *Пересечение* $A \cap B$ — множество $\{\, x : x \in A$ и $x \in B \,\}$. Пересечение состоит из общих элементов множеств A и B. *Разностью* называется множество $A - B = \{\, x : x \in A, x \notin B \,\}$. Декартово (или прямое) *произведение* — множество $A \times B = \{\, (x, y) : x \in A, y \in B \,\}$ — названо так в честь французского математика Р. Декарта (1596–1650). Элементами этого произведения являются пары (x, y). Наконец, имея множество A, мы можем образовать множество всех его подмножеств $\wp(A) = \{\, B : B \subset A \,\}$. Отметим, что A и $\emptyset$ всегда принадлежат $\wp(A)$.

Множество I называется *множеством индексов* для семейства множеств $F = \{\, A_i \,\}$, если для каждого $i \in I$ задано множество A_i из семейства F. Если I — множество из n элементов, мы часто будем обозначать семейство из n различных множеств через $F = \{\, A_1, \ldots, A_n \,\}$. Как и в случае двух множеств, можно рассматривать объединение и пересечение семейства множеств, $\cup_{i \in I} A_i = \{\, x :$

$x \in A_i$ для некоторого $i\,\}$ и $\cap_{i\in I} A_i = \{\, x : x \in A_i$ для всех $i\,\}$. Для семейства из n множеств декартово произведение определяется как

$$\prod_{1\le i\le n} A_i = A_1 \times \cdots \times A_n = \{\, (x_1, \ldots, x_n) : x_i \in A_i \,\}.$$

Элементами произведения являются строки длины n. Если $A = A_1 = \cdots = A_n$, то n-кратное произведение обозначается кратко через A^n.

Определение 2.1.1. *Разбиением* множества S называется множество ϖ его подмножеств, такое, что:

- **a.** Если $A \in \varpi$, то $A \neq \emptyset$.
- **b.** Если $A \in \varpi$ и $B \in \varpi$, то либо $A = B$, либо $A \cap B = \emptyset$.
- **c.** Каждый элемент множества S принадлежит некоторому элементу множества ϖ.

Иначе говоря, разбиение множества S — это семейство его непустых подмножеств, таких, что каждый элемент из S принадлежит в точности одному подмножеству из этого семейства. Элементы разбиения называются блоками. Например, $\{\, \{1,2,3\}, \{4,5,6\} \,\}$ — разбиение множества $\{\, 1,2,3,4,5,6 \,\}$ на два блока. Подмножество R множества S называется множеством *представителей* для разбиения ϖ, если R содержит по одному элементу из каждого блока ϖ.

Приведем предложение, утверждающее, что пересечение дистрибутивно относительно объединения. В упражнениях содержатся другие результаты относительно объединений и пересечений.

Предложение 2.1.2. Для множеств A, B, C выполняется равенство $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

Доказательство. Докажем половину утверждения, а именно включение $A \cap (B \cup C) \subset (A \cap B) \cup (A \cap C)$. Пусть $x \in A \cap (B \cup C)$. Тогда $x \in A$ *и* ($x \in B$ *или* $x \in C$). Если $x \in B$, то $x \in A \cap B$. Если $x \in C$, то $x \in A \cap C$. В любом случае x принадлежит объединению $(A \cap B) \cup (A \cap C)$. Доказательство обратного включения оставим читателю в качестве упражнения. □

2.1.2. Отношения эквивалентности

Бинарным отношением R на непустом множестве A называется подмножество декартова произведения $A \times A$, т.е. $R \subset A \times A$. Например, отношение «меньше, чем», $x < y$, на множестве $\mathbb{Z}$ задается подмножеством $\mathbb{Z} \times \mathbb{Z} \supset LT = \{\, (x,y) : y - x$ положительно$\}$. В

более общем случае n-арным отношением R на непустом множестве A называется подмножество декартовой степени A^n, т. е. $R \subset A^n$. Отношение $R^{-1} = \{\,(y,x) : (x,y) \in R\,\}$ называется *обратным* (*инверсным*) к R. Очевидно, что $(R^{-1})^{-1} = R$. Например, обратным к $>$ является $<$.

Определение 2.1.3. *Отношением эквивалентности* E на множестве A называется бинарное отношение $E \subset A \times A$, удовлетворяющее следующим трем условиям:

a. $(x,x) \in E$ для всех $x \in A$ (рефлексивность).

b. $(x,y) \in E$ влечет за собой $(y,x) \in E$ (симметричность).

c. $(x,y) \in E$ и $(y,z) \in E$ влечет за собой $(x,z) \in E$ (транзитивность).

Будем писать $x \equiv_E y$ (или просто $x \equiv y$, когда понятно, о каком E идет речь), если $(x,y) \in E$; это читается так: «x эквивалентно y». Для каждого элемента $x \in A$ образуем множество эквивалентных ему элементов $\mathbf{x} = \{\,y \in A : y \equiv x\,\}$, которое называется *классом эквивалентности* элемента x. В силу рефлексивности $x \in \mathbf{x}$. Элемент $y \in \mathbf{x}$ называется *представителем* этого класса эквивалентности. Если $y \in \mathbf{x}$, то из симметричности и транзитивности следует, что $\mathbf{y} = \mathbf{x}$, поэтому любые представители некоторого класса эквивалентности определяют один и тот же класс эквивалентности. Образуем множество всех возможных (различных) классов эквивалентности, $A/\equiv\, = \{\,\mathbf{x} : x \in A\,\}$. Оно называется *фактормножеством* множества A по этому отношению эквивалентности и, как мы увидим ниже, является разбиением множества A.

Пример. На $\mathbb{Z} \times \mathbb{Z}^*$ определим отношение эквивалентности $(x,y) \equiv (x',y')$, если $xy' = x'y$. Например, $(-3,-9) \equiv (2,6)$ и $(-1,2) \equiv (3,-6)$. Проверьте, что это действительно отношение эквивалентности. Мы утверждаем, что фактормножество $\mathbb{Z} \times \mathbb{Z}^*/\equiv$ может быть отождествлено с множеством рациональных чисел $\mathbb{Q}$. Действительно, элементы из $\mathbb{Q}$ представляются дробями x/y, $x \in \mathbb{Z}$, $y \in \mathbb{Z}^*$, и $x/y = x'/y'$, когда $xy' = x'y$. Таким образом, элементы из $\mathbb{Q}$ — это в точности классы эквивалентности.

Покажем, что между отношениями эквивалентности на множестве A и разбиениями этого множества существует тесная связь.

Предложение 2.1.4. Если E — отношение эквивалентности на множестве A, то фактормножество $A/E = \{\,\mathbf{a} : a \in A\,\}$ является разбиением множества A. Обратно, если ϖ — разбиение множества A,

то его можно представить как фактормножество A/E для некоторого отношения эквивалентности E на множестве A.

Доказательство. Для доказательства первой части предложения нужно показать, что различные классы эквивалентности не пересекаются и что их объединение совпадает с A.

Так как E рефлексивно, для всякого $a \in A$ выполняется $a \in \mathbf{a}$. Отсюда следует, что каждый класс $\mathbf{a}$ непуст и что их объединение совпадает с A.

Докажем, что различные классы не пересекаются. Предположим, что $x \in \mathbf{a}$ и $x \in \mathbf{b}$ для некоторого $x \in A$; покажем, что $\mathbf{a} = \mathbf{b}$. По определению $\mathbf{a}$ и $\mathbf{b}$ имеем $(x, a) \in E$ и $(x, b) \in E$; в силу симметричности $(a, x) \in E$ и $(x, b) \in E$, что по транзитивности влечет за собой $(a, b) \in E$ и $\mathbf{a} = \mathbf{b}$.

Для доказательства второй части предложения определим отношение E_ϖ на A, такое, что $(a, b) \in E_\varpi$ тогда и только тогда, когда a и b лежат в одном и том же блоке разбиения ϖ. (Покажите, что это отношение эквивалентности.) Мы покажем, что $A/E_\varpi = \varpi$; ниже ϖ_a обозначает единственный блок ϖ, содержащий элемент $a \in A$.

Докажем включение $\varpi \subset A/E_\varpi$. Пусть ϖ_a непусто; покажем, что $\varpi_a = \mathbf{a} \in A/E_\varpi$. Рассмотрим $x \in \mathbf{a}$; по определению $\mathbf{a}$ имеем $(x, a) \in E_\varpi$, откуда получаем $x \in \varpi_a$ по определению E_ϖ, что доказывает включение $\mathbf{a} \subset \varpi_a$. Для доказательства обратного включения рассмотрим $x \in \varpi_a$; тогда $(x, a) \in E_\varpi$ по определению E_ϖ, откуда получаем $x \in \mathbf{a}$ по определению $\mathbf{a}$, что доказывает включение $\varpi_a \subset \mathbf{a}$. Таким образом, каждый блок ϖ_a разбиения ϖ совпадает с некоторым классом эквивалентности $\mathbf{a}$, принадлежащим A/E_ϖ.

Доказательство обратного включения $A/E_\varpi \subset \varpi$ предоставляется читателю. □

На практике значение отношений эквивалентности связано с идеей *игнорирования подробностей*; а именно, когда мы имеем дело с классами эквивалентности, мы как бы не замечаем различий между эквивалентными элементами множества A.

Пример. Зафиксируем целое число $m > 1$. Зададим такое отношение эквивалентности на $\mathbb{Z}$: $b \equiv_m a$, если $b - a = mq$ для некоторого $q \in \mathbb{Z}$; иначе говоря, $b \equiv_m a$, если их разность есть целое кратное m. Несколько примеров: $-4 \equiv_5 16$, $91 \equiv_7 0$ и $1087 \equiv_2 1$. Класс эквивалентности элемента a — это множество $\mathbf{a} = \{\, a + mq : q \in \mathbb{Z} \,\}$, которое мы также будем обозначать $\mathbf{a} = a + m\mathbb{Z}$. Если $m = 5$, то $\mathbf{0} = \{\ldots, -10, -5, 0, 5, 10, \ldots\}$, $\mathbf{1} = \{\ldots, -9, -4, 1, 6, 11, \ldots\}$ и так далее;

таким образом, мы разбили $\mathbb{Z}$ на непересекающиеся подмножества $\mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}$, и фактормножество есть $\mathbb{Z}/\equiv_5 = \{\, \mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4} \,\}$. Заметим, что, хотя каждый из классов эквивалентности содержит *бесконечно много* элементов, множество классов эквивалентности содержит всего *пять* элементов. В общем случае $\mathbb{Z}/\equiv_m = \{\, \mathbf{0}, \ldots, \mathbf{m-1} \,\}$ содержит m элементов.

2.1.3. Функции и алгебраические системы

Мы начнем со следующего определения.

Определение 2.1.5. Пусть A и B — непустые множества. *Функцией* из множества A в множество B или *отображением* множества A в множество B называется правило, ставящее в соответствие каждому элементу $x \in A$ единственный элемент $y \in B$.

С теоретико-множественной точки зрения мы можем рассматривать функцию f из A в B как подмножество произведения $A \times B$, т.е. $f \subset A \times B$, со следующим свойством: для всякого $x \in A$ существует ровно один элемент $y \in B$, такой, что $(x, y) \in f$. Для $x \in A$ $f(x)$, *образ* x или *значение* f в x — это единственный элемент $y \in B$, соответствующий элементу x. Мы используем для функции f из A в B обозначение $f\colon A \to B$ или $A \xrightarrow{f} B$. Множество $f(A) = \{\, f(x) : x \in A \,\}$ (подмножество множества B) называется *образом* A относительно функции f. Для $y \in B$ множество $f^{-1}(y) = \{\, x \in A : f(x) = y \,\}$ (подмножество множества A) называется *прообразом* элемента y. Прообраз y — множество всех элементов из A, образ которых равен y. (*Замечание.* Как мы увидим ниже, f^{-1} является подмножеством в $A \times B$, но не функцией.) В случае когда множества обладают алгебраической структурой (когда мы имеем дело с группами, кольцами, полями — эти понятия будут определены ниже), функции, сохраняющие алгебраическую структуру (бинарные операции), называются *гомоморфизмами* (homo = та же, morphism = форма). Более точно, отображение $f\colon A \to B$, где A и B — множества с алгебраической структурой, называется гомоморфизмом, если

i. $f(a + b) = f(a) + f(b)$,

ii. $f(a \cdot b) = f(a) \cdot f(b)$,

iii. $f(\mathbf{1}_A) = \mathbf{1}_B$.

Здесь $\mathbf{1}_A$ и $\mathbf{1}_B$ — мультипликативные единицы (см. определение 2.1.13) в A и B. (Отметим, что условие **iii** в общем случае не выполняется, но будет справедливо везде в этой книге.)

Функция (отображение) $f\colon A \to B$ называется *сюръективной* (*отображением на*), если $f(A) = B$ (иначе говоря, каждый элемент в B является образом некоторого элемента из A). Функция f называется *взаимно однозначной* или *инъективной*, если $x \neq x'$ влечет за собой $f(x) \neq f(x')$ (иначе говоря, образы различных элементов из A различны в B). Инъективная и сюръективная функция называется *биективной* или просто *биекцией*. Биективный гомоморфизм называется *изоморфизмом*, а множества A и B — *изоморфными*; это обозначается через $A \cong B$.

Примеры функций. Пусть A, B обозначают множества.

1. Для всякого множества A тождественная функция $id\colon A \to A$ задается формулой $id(x) = x$ для любых $x \in A$. Очевидно, что id биективна.
2. Проекция произведения множеств на первый сомножитель $p_1\colon A \times B \to A$ задается формулой $p_1[(x, y)] = x$. Она сюръективна. Чему равно $p_1^{-1}(x)$ для $x \in A$? Аналогично определяется проекция на второй сомножитель B.
3. Если на A задано отношение эквивалентности $\equiv$, то имеется каноническая (или естественная) функция $s\colon A \to A/\equiv$, которая отображает каждый элемент x в его класс эквивалентности, $s(x) = \mathbf{x}$, в фактормножестве $A/\equiv$. Мы будем использовать одно и то же обозначение $\mathbf{x}$ для класса эквивалентности как в случае, когда мы рассматриваем его как подмножество в A, так и в случае, когда мы рассматриваем его как элемент фактормножества $A/\equiv$; обычно из контекста понятно, что имеется в виду. Например, в случае $s_5\colon \mathbb{Z} \to \mathbb{Z}/\equiv_5$ прообраз класса эквивалентности, $\mathbf{4} = \{\ldots, -6, -1, 4, 9, \ldots\}$, рассматриваемого как *элемент* фактормножества, — это множество $s_5^{-1}(\mathbf{4}) = \mathbf{4}$, которое в последнем случае рассматривается как *подмножество* в $\mathbb{Z}$.
4. Функция следования $\mathrm{succ}\colon \mathbb{N} \to \mathbb{N}$ задается формулой $\mathrm{succ}(n) = n + 1$. Она инъективна. (Почему?)
5. Функция сложения $+\colon \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$, где $+(x, y) = x + y$ — пример бинарной операции. Чему равно $+^{-1}(0)$? Является ли функция $+$ инъективной либо сюръективной?

Как сказано выше, сложение — пример бинарной операции. Бинарная операция $\oplus$ на множестве S *может* обладать следующими двумя важными свойствами. Если для любых x, y в S выполняется равенство $x \oplus y = y \oplus x$, то $\oplus$ называется *коммутативной*.

Если для всякой тройки элементов x, y, z в S выполняется равенство $x \oplus (y \oplus z) = (x \oplus y) \oplus z$, то $\oplus$ называется *ассоциативной*. Например, на $\mathbb{Z}$ операции $+$ и $\cdot$ коммутативны и ассоциативны, а операция $-$ не обладает ни одним из этих свойств.

Дадим точное определение конечного множества. Будем называть множество S *конечным*, если всякая инъективная функция $f: S \to S$ сюръективна. Как мы увидим, понятия сюръективности и инъективности совпадают для функций, отображающих конечное множество S в S (теорема 2.1.7); это наиболее важное свойство конечных множеств. Функция следования $\mathrm{succ}: \mathbb{N} \to \mathbb{N}$, заданная равенством $\mathrm{succ}(n) = n + 1$, инъективна, но не сюръективна; поэтому, согласно нашему определению, множество $\mathbb{N}$ не конечно. Справедлива следующая теорема.

Теорема 2.1.6. Множество S конечно тогда и только тогда, когда существуют единственное $n \in \mathbb{N}$ и биекция множества $\{ 1, 2, \ldots, n \}$ на S.

Доказательство. Теорема интуитивно очевидна. □

Теорема 2.1.7. Пусть S — конечное множество и $f: S \to S$ — сюръективная функция. Тогда f инъективна.

Доказательство. Так как f сюръективна, для каждого элемента $s \in S$ мы можем выбрать $t \in S$, такой, что $f(t) = s$; определим функцию $g: S \to S$ равенством $g(s) = t$. Если $g(s_1) = g(s_2)$, то $s_1 = f[g(s_1)] = f[g(s_2)] = s_2$, и потому g инъективна; следовательно, поскольку S конечно, g должна быть сюръективной. Если бы f не была инъективной, то существовали бы $s_1 \neq s_2$ в S, такие, что $f(s_1) = f(s_2)$. Однако, поскольку g сюръективна, существуют t_1 и t_2 в S, такие, что $g(t_1) = s_1$ и $g(t_2) = s_2$. Имеем $t_1 = f[g(t_1)] = f(s_1) = f(s_2) = f[g(t_2)] = t_2$, что противоречит соотношению $g(t_1) \neq g(t_2)$. □

Две функции $f, g: A \to B$ *равны*, $f = g$, если $f(x) = g(x)$ для всех $x \in A$. Если $f: A \to B$ и $g: B \to C$ — две функции, то $f(x) \in B$ для всякого $x \in A$ и, следовательно, $g[f(x)]$ имеет смысл. Мы определим *композицию* функций f и g как функцию $g \circ f: A \to C$, заданную равенством $g \circ f(x) = g[f(x)]$ для любых $x \in A$. Композиция $g \circ f$ действует так: сначала применяется f, затем g.

Пример. Сложение по модулю 5 — это функция $+_5: \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}/\equiv_5$, заданная равенством $+_5(x, y) = s_5(x + y)$. Она является композици-

ей функции сложения $+$ и канонического факторотображения s_5 в фактормножество из предыдущего примера: $+_5 = s_5 \circ +$.

В случае когда композиция функций имеет смысл, выполняется закон ассоциативности.

Предложение 2.1.8 (закон ассоциативности для композиции). Для функций $f: A \to B$, $g: B \to C$ и $h: C \to D$ справедливо равенство $h \circ (g \circ f) = (h \circ g) \circ f$.

Доказательство. Для $x \in A$ имеем $h \circ (g \circ f)(x) = h[g \circ f(x)] = h\{g[f(x)]\} = h \circ g[f(x)] = (h \circ g) \circ f(x)$. □

Если $f: A \to B$ — биекция, то для каждого $y \in B$ прообраз $f^{-1}(y)$ состоит ровно из одного элемента $\{ x \}$, такого, что $f(x) = y$. Определим *обратную функцию* $f^{-1}: B \to A$ как функцию, которая каждому y ставит в соответствие единственный элемент x, такой, что $f(x) = y$, где $y \in B$. Применяя операцию композиции к f и f^{-1}, получаем функции $f^{-1} \circ f: A \to A$ и $f \circ f^{-1}: B \to B$; нетрудно проверить, что $f^{-1} \circ f(x) = x$ и $f \circ f^{-1}(y) = y$. Таким образом, $f^{-1} \circ f = id_A$ и $f \circ f^{-1} = id_B$, где через id_A и id_B обозначены тождественные функции на соответствующих множествах.

Суммируем некоторые свойства функций, которые сохраняются при композиции.

Предложение 2.1.9. Пусть $f: A \to B$ и $g: B \to C$; тогда

- **a.** Если f, g сюръективны, то $g \circ f$ сюръективна.
- **b.** Если f, g инъективны, то $g \circ f$ инъективна.
- **c.** Если f, g биективны, то $g \circ f$ биективна и $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$; обратная функция к композиции равняется композиции обратных функций в обратном порядке.

Доказательство. Докажем только часть **a**, оставив остальное читателю. Пусть $c \in C$; поскольку g отображает B на C, найдется элемент $b \in B$, такой, что $g(b) = c$. Далее, так как f также сюръективна, найдется элемент $a \in A$, такой, что $f(a) = b$. Поэтому $g \circ f(a) = g[f(a)] = g(b) = c$, и $g \circ f$ сюръективна. □

Рассмотрим функцию $f: A \to B$. Определим отношение эквивалентности на множестве A следующим образом: два элемента $x, x' \in A$ эквивалентны, $x \equiv x'$, если $f(x) = f(x')$. Проверьте, что это отношение эквивалентности. Зададим теперь функцию $i: A/\equiv \to B$ на множестве классов эквивалентности равенством $i(\mathbf{x}) = f(x)$ для класса эквивалентности $\mathbf{x}$. Мы определили образ класса $\mathbf{x}$ под действием

i через представитель x; нужно проверить, что если мы выберем другой представитель $x' \in \mathbf{x}$, так что $\mathbf{x} = \mathbf{x}'$, то получим такой же образ. По нашему определению i имеем $i(\mathbf{x}') = f(x')$. Но $x' \equiv x$ означает, что $f(x) = f(x')$, поэтому i определена корректно. По построению i инъективна. Пусть $s: A \to A/\equiv$ — каноническое отображение на фактормножество. В использованных обозначениях мы доказали следующую теорему.

Теорема 2.1.10 (теорема о факторизации для функций). Пусть $f: A \to B$ — произвольная функция. Тогда диаграмма

$$\begin{array}{c} A \xrightarrow{f} B \\ s \searrow \quad \nearrow i \\ A/\equiv \end{array}$$

представляет f в виде композиции сюръекции s и инъекции i, а именно $f = i \circ s$.

Пример. Рассмотрим функцию $f: \mathbb{Z} \to \mathbb{N}$, $f(x) = x^2$, $x \in \mathbb{Z}$. Она представляется в виде $f = i \circ s$, где сюръекция — это функция $s: \mathbb{Z} \to \mathbb{Z}/\equiv_{()^2} = \{0\} \cup \{\, \mathbf{x} = \{x, -x\} : x \in \mathbb{Z}^* \,\}$, а инъекция — функция $i: \mathbb{Z}/\equiv_{()^2} \to \mathbb{N}$.

В теореме 2.1.10 утверждается, что f пропускается через фактормножество $A/\equiv$. Когда множества имеют алгебраическую структуру (например, структуру группы или кольца), фактор-множество также имеет ту же структуру; можно использовать теорему о факторизации, чтобы показать, что всякий гомоморфизм представляется в виде композиции сюръективного и инъективного гомоморфизмов. Непосредственно получаем

Следствие 2.1.11. Функция $i: A/\equiv \to f(A)$, где $f(A)$ — образ множества A, лежащий в B, является биекцией на $f(A)$.

Рассмотрим один важный пример, который приведет нас к понятию группы, первой алгебраической системы, которую мы хотим представить.

Пусть A — некоторое множество. Обозначим множество биекций из A в A через $Bij(A) = \{\, A \xrightarrow{f} A,\ f$ — биекция $\}$. По предложению 2.1.8 композиция функций ассоциативна; по предложению 2.1.9 композиция двух биекций также является биекцией. Далее, тождественная функция id принадлежит $Bij(A)$, и если f принадлежит $Bij(A)$, то и обратная функция f^{-1} принадлежит $Bij(A)$. Собирая вместе эти результаты, мы получаем следующую теорему.

Теорема 2.1.12 (определение группы). Пусть A — произвольное непустое множество. Рассмотрим множество биекций на A, $Bij(A) = \{ A \xrightarrow{f} A,\ f$ — биекция $\}$. Тогда:

a. $f,\ g \in Bij(A)$ влечет за собой $f \circ g \in Bij(A)$.

b. $h \circ (f \circ g) = (h \circ f) \circ g$ для любых $f,\ g,\ h \in Bij(A)$.

c. $id \circ f = f \circ id = f$ для $f \in Bij(A)$.

d. Если $f \in Bij(A)$, то $f^{-1} \in Bij(A)$ и $f^{-1} \circ f = f \circ f^{-1} = id$.

Пункт **a** теоремы 2.1.12 утверждает, что $Bij(A)$ замкнуто относительно бинарной операции $\circ$, пункт **b** — что $\circ$ ассоциативна, пункт **c** — что тождественная функция является единичным элементом в $Bij(A)$ и пункт **d** — что каждый элемент из $Bij(A)$ имеет обратный в $Bij(A)$. В общем случае множество, замкнутое относительно ассоциативной бинарной операции, имеющее единичный элемент и в котором все элементы обратимы, называется *группой*. Таким образом, $Bij(A)$ — группа относительно операции $\circ$. Обычно если операция в группе называется сложением (умножением), то группа называется *аддитивной* (*мультипликативной*). Пусть $\tilde{n} = \{ 1, \ldots, n \}$ — множество из n элементов; тогда $S_n = Bij(\tilde{n})$ обозначает множество всех перестановок на n элементах. S_n называется *симметрической группой степени* n. Она содержит $n!$ элементов. Очевидно, что *ненулевые* рациональные числа с операцией умножения образуют группу с 1 в качестве групповой единицы и $1/x$ в качестве обратного элемента к произвольному ненулевому элементу x; все множество рациональных чисел *не* образует группу.

Рассмотрим теперь алгебраические системы с двумя бинарными операциями — сложением и умножением.

Определение 2.1.13. *Кольцом* $(R, +, \cdot)$ называется алгебраическая система, удовлетворяющая следующим условиям:

a. R является коммутативной группой относительно операции $+$ (коммутативность означает, что $x+y = y+x$ для любых $x, y \in R$). Единичный элемент относительно операции сложения называется *нулевым* элементом и обозначается 0_R или просто 0.

b. R не является группой по умножению, так как (некоторые) элементы могут не иметь обратных, однако оно замкнуто относительно умножения, умножение ассоциативно и существует единичный элемент относительно умножения, обозначаемый 1_R или просто 1.

c. Умножение дистрибутивно относительно сложения: $x(y+z) = xy + xz$ и $(y+z)x = yx + zx$ для любых $x, y, z \in R$.

Множества $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ являются кольцами относительно обычных операций. Некоторые примеры множеств, не являющихся кольцами: множество натуральных чисел $\mathbb{N}$, множество положительных вещественных чисел $\mathbb{R}^+$ с обычными операциями $+$ и $\cdot$, множество $\mathbb{Z} - \{5\}$ всех целых чисел, кроме 5.

Множество $\mathbb{Z}-\{5\}$ не является кольцом относительно обычных операций сложения и умножения, потому что для элементов $a, b \in \mathbb{Z}-\{5\}$ сумма $a+b$ не обязательно принадлежит $\mathbb{Z}-\{5\}$; например, $2+3=5$. В этом случае мы говорим, что $\mathbb{Z}-\{5\}$ *не замкнуто относительно сложения*, и подразумеваем, что $\mathbb{Z}-\{5\}$ — подмножество большего множества $\mathbb{Z}$, в котором сумма и произведение элементов всегда имеют смысл. Аналогично определяется *незамкнутость относительно умножения*.

Кольцо называется *коммутативным*, если умножение в нем коммутативно, $ab = ba$. Пусть R — коммутативное кольцо, а 0, 1 обозначают единичные элементы для операций сложения и умножения. Возможны две ситуации. В первой $0 \neq 1 + 1 + \cdots + 1$ (n раз) для любого $n > 0$. В этом случае мы будем говорить, что R имеет *характеристику* 0; примерами являются $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ и $\mathbb{C}$. В другой ситуации существует $n > 0$, такое, что $0 = 1 + 1 + \cdots + 1$ (n раз), и этот случай будет рассмотрен ниже.

Уделим теперь внимание двум очень важным типам коммутативных колец.

Определение 2.1.14. Элемент $a \neq 0$ коммутативного кольца R называется *делителем нуля*, если $ab = 0$ для некоторого $b \neq 0$ (b также делитель нуля). Элемент $u \neq 0$ из R называется *обратимым*, если существует обратный к нему элемент, т.е. $uv = 1$ для некоторого $v \in R$ $\quad (v = u^{-1})$.

Пример. Рассмотрим алгебраическую систему $(\mathbb{Z}/\equiv_m, +, \cdot)$; проверьте, что это коммутативное кольцо. Как мы еще раз увидим ниже, при работе с целыми числами по модулю m мы определяем произведение двух классов эквивалентности из $\mathbb{Z}/\equiv_m$ как класс эквивалентности, содержащий произведение представителей. Таким образом, если $m = 8$ и $\mathbf{2}, \mathbf{4} \in \mathbb{Z}/\equiv_8$, то $\mathbf{2} \cdot \mathbf{4} = \mathbf{0}$; значит, $\mathbf{2}$ и $\mathbf{4}$ являются делителями нуля. Заметим, что при $m = 5$ подобный пример невозможен. Действительно, каждый ненулевой элемент из $\mathbb{Z}/\equiv_5$ обратим; обратные к элементам $\mathbf{2}$, $\mathbf{3}$ и $\mathbf{4}$ — это элементы $\mathbf{3}$, $\mathbf{2}$ и $\mathbf{4}$ соответственно.

Из предыдущего определения несложно вывести, что элемент кольца не может быть одновременно обратимым и делителем нуля, и мы приходим к определению двух важных классов нетривиальных коммутативных колец, где *нетривиальность* означает, что кольцо имеет более одного элемента.

Определение 2.1.15. *Областью целостности* называется нетривиальное коммутативное кольцо без делителей нуля.

Классический пример области целостности (из которого возникло название) — кольцо целых чисел $\mathbb{Z}$; в нем обратимы только элементы 1 и -1.

Определение 2.1.16. *Полем* называется нетривиальное коммутативное кольцо, в котором каждый ненулевой элемент обратим, или, эквивалентно, *поле* $(F, +, \cdot)$ — это алгебраическая система, удовлетворяющая следующим условиям:

- **a.** F — коммутативная группа относительно операции $+$.
- **b.** Множество ненулевых элементов из F образует коммутативную группу относительно операции $\cdot$.
- **c.** Умножение дистрибутивно относительно сложения.

Отметим, что поле является областью целостности. Несложно проверить, что множество вещественных чисел $\mathbb{R}$ с обычными операциями сложения и умножения является полем; $\mathbb{Q}$ и $\mathbb{C}$ также поля. Эти поля содержат бесконечно много элементов. Существуют и поля с конечным числом элементов; они называются *полями Галуа* (GF). Например, множество $\{0, 1\}$ с операциями сложения и умножения по модулю 2 — поле, обозначаемое $GF(2)$; проверьте это.

Рассмотрим конечное поле из q элементов $GF(q)$ и образуем последовательность сумм

$$\sum_{1 \le i \le k} 1 = 1 + 1 + \cdots + 1 \qquad (k \text{ раз}),$$

$k = 1, 2, 3, \ldots$. Так как поле замкнуто относительно сложения, эти суммы должны быть его элементами; более того, так как количество элементов определяемого поля конечно, в этой последовательности должны быть повторения. Поэтому существуют положительные целые числа k' и k'', $k' < k''$, такие, что

$$\sum_{1 \le i \le k'} 1 = \sum_{1 \le i \le k''} 1.$$

Отсюда следует, что $\sum_{1 \le i \le k^n - k^l} 1 = 0$, и потому существует *наименьшее* целое положительное число λ, такое, что $\sum_{1 \le i \le \lambda} 1 = 0$. Это число λ называется характеристикой поля $GF(q)$. Например, характеристика поля $GF(2)$ равняется двум, так как $1 + 1 = 0$.

2.2

Наибольшие общие делители целых чисел

Вычисление наибольшего общего делителя целых чисел — одна из старейших задач, исследованных математиками; предложенный Евклидом метод ее решения является самым древним из существующих теоретико-числовых алгоритмов. В этом разделе мы рассмотрим алгоритм Евклида и близкие вопросы.

2.2.1. Делимость целых чисел

Мы начнем с очень важного принципа, который будет использоваться в доказательствах (Childs, 1979).

Принцип полной упорядоченности. Пусть k_0 — произвольное целое число. Тогда всякое непустое множество целых чисел, больших или равных k_0, имеет наименьший элемент.

Доказательство. Докажем по индукции, что всякое множество целых чисел $\ge k_0$, не имеющее наименьшего элемента, должно быть пустым. Пусть S — множество целых чисел $\ge k_0$ без наименьшего элемента, и пусть $p(k)$ — это утверждение «S не содержит целых чисел $\le k$». Если мы покажем, что $p(k)$ истинно для всех $k \ge k_0$, то отсюда будет следовать, что S пусто, так как если j содержится в S, то $p(j)$ ложно. Очевидно, что $p(k_0)$ истинно, потому что иначе S имело бы наименьший элемент, поскольку k_0 содержалось бы в S и все содержащиеся в S числа $\ge k_0$. Предположим, что $p(n)$ истинно для некоторого $n \ge k_0$; мы покажем, что $p(n+1)$ также истинно, завершая таким образом доказательство по индукции. Если $p(n+1)$ ложно, то некоторое число $\le n+1$ содержится в S. Однако, поскольку $p(n)$ истинно, S не содержит чисел $\le n$, откуда следует, что $n+1$ содержится в S и является его наименьшим элементом. Мы получили противоречие. Следовательно, $p(n+1)$ истинно и S пусто. □

Можно показать, что принцип полной упорядоченности эквивалентен принципу индукции (см. упр. 3 разд. 2.2.1) и что справедлив

двойственный принцип: если S — множество целых чисел, все элементы которого $\leq k_0$, то S имеет наибольший элемент. Мы будем многократно использовать этот принцип в данной книге.

Будем говорить, что ненулевое целое число a *делит* b или a является *делителем* b (записывается $a \mid b$), если существует c, такое, что $b = a \cdot c$. Например, $\pm 7 \mid 28$, так как $28 = 7 \cdot 4$ и $28 = (-7)(-4)$. Для любого ненулевого a имеем $a \mid 0$, $\pm 1 \mid a$ и $\pm a \mid a$. Понятие делителя будет очень важным в нашем изложении теории целых чисел.

Одно из основных свойств целых чисел — это свойство *делимости* или *евклидовости*, которое хорошо известно из арифметики.

Теорема 2.2.1 (свойство евклидовости). Для любого a и любого ненулевого b существуют единственные (целые) *частное* q и *остаток* r, такие, что $a = b \cdot q + r$, $0 \leq r < |b|$.

Доказательство. Рассмотрим множество целых чисел вида $a - kb$, где k пробегает все множество целых чисел, положительных и неположительных; т.е. рассмотрим прогрессию

$$\ldots, a - 3b, a - 2b, a - b, a, a + b, a + 2b, a + 3b, \ldots .$$

Выберем в этой последовательности наименьшее неотрицательное число и обозначим его r, и пусть q обозначает соответствующее значение k. (Такое r существует, потому что множество $\{ a - kb \}$ содержит отрицательные и неотрицательные значения, а из полной упорядоченности следует, что непустое множество неотрицательных целых чисел содержит наименьший элемент.) По определению

$$r = a - qb \geq 0.$$

Для доказательства единственности допустим, что

$$a = b \cdot q_1 + r_1, \qquad 0 \leq r_1 < |b|,$$

и что $r_1 \neq r$. Пусть для определенности $r_1 < r$, так что $0 < r - r_1 < |b|$; тогда

$$r - r_1 = (q_1 - q)b$$

и $b \mid (r - r_1)$, что противоречит неравенствам $0 < r - r_1 < |b|$. □

Определение 2.2.2. Пусть a и b одновременно не равны нулю. Целое число $d > 0$ называется *наибольшим общим делителем* a и b, если

a. $d \mid a$ и $d \mid b$.

b. Если $c \mid a$ и $c \mid b$, то $c \mid d$.

Наибольший общий делитель чисел a и b обозначается через $\gcd(a,b)$ или (a,b). Последнее обозначение можно спутать с парой чисел, поэтому его смысл должен определяться из контекста. Единственность наибольшего общего делителя следует из свойства **b** в определении и того, что он положителен: если d' — другой наибольший общий делитель, то по свойству **b** $d \mid d'$ и $d' \mid d$, поэтому $d' = d$, поскольку оба положительны. Например, $(12,30) = (12,-30) = (-12,30) = (-12,-30) = 6$. Наибольший общий делитель двух целых чисел, одновременно не равных нулю, всегда существует и может быть представлен в следующем виде.

Теорема 2.2.3 (существование gcd). Если a и b одновременно не равны нулю, то существуют целые числа x и y, такие, что $(a,b) = ax + by$.

Доказательство. Пусть d — наименьшее положительное целое число вида $ax + by$, например, $d = ax_0 + by_0$, где x_0, y_0 не обязательно определены однозначно. (Как и в доказательстве теоремы 2.2.1, существование такого d следует из полной упорядоченности.) Очевидно, что $d > 0$ и d обладает свойством **b** из определения 2.2.2. От противного мы докажем, что d обладает также свойством **a**. Допустим, что свойство **a** не выполняется, и предположим для определенности, что d не делит b. Тогда $b = d \cdot q + r, \quad 0 < r < d$, и, следовательно, $r = b - dq = b - (ax_0 + by_0)q = a(-qx_0) + b(1 - qy_0)$, что противоречит минимальности d. □

Теорема не утверждает, что x и y определены однозначно, она лишь говорит о том, что наибольший общий делитель может быть выражен в таком виде. Например, $6 = (12,-30) = 12(3) + (-30)(1) = 12(-2) + (-30)(-1)$. Ниже, основываясь на свойстве евклидовости, мы опишем алгоритм представления наибольшего общего делителя в форме из теоремы 2.2.3.

Пользуясь понятием наибольшего общего делителя, мы можем охарактеризовать целые решения линейных уравнений от двух переменных (линейных диофантовых уравнений).

Теорема 2.2.4. Рассмотрим уравнение $ax + by = c$, в котором a и b не равны нулю одновременно, и пусть $d = (a,b)$. Тогда

- **a.** Уравнение разрешимо относительно x и y тогда и только тогда, когда $d \mid c$.
- **b.** Если x_0, y_0 — частное решение, то все решения имеют вид $x_0 - n(b/d), \quad y_0 + n(a/d)$ для всех n.

Доказательство. Мы докажем только часть **a**, оставив часть **b** читателю в качестве упражнения. Предположим, что x и y — целые числа, такие, что $ax + by = c$; тогда, так как $d \mid a$ и $d \mid b$, то $d \mid c$. Предположим теперь, что $d \mid c$, т.е. $c = dk$ для некоторого целого k. По теореме 2.2.3 существуют целые числа s, t, такие, что $d = as + bt$. Умножая это равенство на k, получим $c = dk = a(sk) + b(tk)$, откуда следует, что $x = sk$ и $y = tk$ удовлетворяют уравнению $ax + by = c$. □

Пример: уравнение $12x - 30y = 84$ разрешимо, поскольку $(12, -30) = 6 | 84$. Одним его решением является пара $x = 2$, $y = -2$, а все остальные имеют вид $x = 2 + 5n$, $y = -2 + 2n$.

Два целых числа a и b называются *взаимно простыми*, если $(a, b) = 1$. По теореме 2.2.3 это эквивалентно существованию целых чисел s, t, таких, что $as + bt = 1$. Справедлива следующая теорема.

Теорема 2.2.5. Пусть целые числа a и b не равны нулю одновременно, и пусть $d = (a, b)$. Тогда a/d и b/d взаимно просты.

Доказательство. По теореме 2.2.3 существуют целые числа s, t, такие, что $d = as + bt$. Разделив на d, получим $1 = (a/d)s + (b/d)t$, что влечет за собой $(a/d, b/d) = 1$. □

Теорема 2.2.6. Пусть a, b и c — целые числа и $d = (a, b)$. Если a делит bc, то a/d делит c.

Доказательство. Если $a \mid bc$, то $(a/d) \mid (b/d)c$. Однако по теореме 2.2.5 $(a/d, b/d) = 1$, и по теореме 2.2.3 существуют целые числа s, t, такие, что $(a/d)s + (b/d)t = 1$; умножив это равенство на c, получим $c(a/d)s + c(b/d)t = c$. Поскольку a/d делит $c(a/d)s$ и $c(b/d)t$, оно делит c. □

Определение 2.2.7. Пусть a и b — ненулевые целые числа. Целое число $m > 0$ называется *наименьшим общим кратным* чисел a и b, если

a. $a \mid m$ и $b \mid m$.

b. Если $a \mid c$ и $b \mid c$, то $m \mid c$.

Наименьшее общее кратное чисел a и b обозначается через $\mathrm{lcm}(a, b)$ или $[a, b]$. Единственность наименьшего общего кратного следует из части **b** и положительности m.

Теорема 2.2.8 (существование lcm). Если a и b — ненулевые числа, то их наименьшее общее кратное существует и справедливо равенство $[a, b] = |ab|/(a, b)$.

Доказательство. Так как a и b не равны нулю, их наибольший общий делитель $d = (a, b)$ отличен от нуля. Из равенства $(ab)/d = a(b/d) = (a/d)b$ и того, что числа b/d и a/d целые, следует, что $m = |ab|/d$ — положительное общее кратное чисел a и b. Пусть n — любое другое общее кратное чисел a и b; тогда $n = as = bt$ для некоторых целых чисел s, t. Поскольку $a \mid bt$, то по теореме 2.2.6 $(a/d) \mid t$ и $t = u(a/d)$ для некоторого целого числа u. Из равенства $n = bt = u(ab)/d$ вытекает, что $m \mid n$, поэтому m — наименьшее общее кратное чисел a и b. Как уже упоминалось выше, единственность следует из части **b** определения 2.2.7 и того, что m положительно. □

2.2.2. Алгоритм Евклида и теорема Ламе

Теперь мы изложим классический *алгоритм Евклида* вычисления наибольшего общего делителя двух целых чисел. Свойство евклидовости утверждает, что для целого числа a и ненулевого целого числа b существует единственные частное q и остаток r, такие, что $a = bq + r$ и $0 \le r < |b|$. В упр. 1 по программированию для разд. 2.2.1 требуется написать подпрограммы $\mathbf{QUO}(a, b)$ и $\mathbf{MOD}(a, b)$, которые возвращают частное и неотрицательный остаток от деления a на b; эти подпрограммы будут использоваться в различных утверждениях и в описаниях других алгоритмов на протяжении всей этой книги.

Ключевым в алгоритме Евклида является следующий факт: если $a = bq + r$ и d делит a и b, то $d \mid r = a - bq$ (упр. 3 разд. 2.2.1). Поскольку это верно для любого делителя, это верно и для $d = \gcd(\mathbf{a}, \mathbf{b})$; значит, $\gcd(a, b) = \gcd[b, \mathbf{MOD}(a, b)]$. Кроме того, $(a, 0) = |a|$ для всякого a; условимся также считать $\gcd(0, 0)$ равным нулю. Итак, если заданы целое число a и ненулевое целое число b, мы выполняем такую последовательность делений:

Пусть $a_0 = a$ и $a_1 = b$; тогда

$$\begin{aligned}
a_0 &= a_1 q_1 + a_2, && 0 < a_2 < |a_1|,\\
a_1 &= a_2 q_2 + a_3, && 0 < a_3 < a_2,\\
&\dots\\
a_{k-2} &= a_{k-1} q_{k-1} + a_k, && 0 < a_k < a_{k-1},\\
a_{k-1} &= a_k q_k + 0.
\end{aligned}$$

Процесс рано или поздно останавливается, так как остатки $|a_1| > a_2 > a_3 > \cdots > 0$ образуют строго убывающую последовательность неотрицательных целых чисел, и a_k является наибольшим общим делителем.

Мы уже отметили, что $(a_0, a_1) = (a_1, a_2) = \cdots = (a_k, 0) = a_k$; значит, мы вычислили наибольший общий делитель чисел a и b и доказали, что следующий алгоритм работает правильно. (В наших обозначениях $x := y$ означает присвоение переменной x значения y; $(x, y) := (x_1, y_1)$ означает $x := x_1$, $y := y_1$.)

EA. Алгоритм Евклида (**E**uclidean **A**lgorithm)

Вход: a и $b \neq 0$.

Выход: $d = \gcd(a, b)$.

1. [Инициализация] $(a_0, a_1) := (a, b)$.

2. [Основной цикл] Пока $a_1 \neq 0$ выполнять $(a_0, a_1) := [a_1, \mathbf{MOD}(a_0, a_1)]$.

3. [Выход] Вернуть $d := a_0$.

Рассмотрим пример $a = 342$ и $b = 612$. Алгоритм Евклида вычисляет последовательность $(342, 612) = (612, 342) = (342, 270) = (270, 72) = (72, 54) = (54, 18) = (18, 0)$; таким образом, $d = 18$.

Анализ времени работы алгоритма ***EA***. Предположим без потери общности, что $a \geq b$.

Поскольку шаги 1 и 3 выполняются за время ~ 1, очевидно, что время работы алгоритма Евклида определяется временем выполнения шага 2.

На шаге 2 выполняется n целочисленных делений, и нам необходимо оценить сверху n; однако, прежде чем сделать это, заметим, что первое деление выполняется с числами a и b, а из разд. 1.2 мы знаем, что $t_{\mathbf{IDIV}}(a, b) = O\{L(b)[L(a) - L(b) + 1]\}$. Так как последующие деления выполняются с меньшими числами, то выражение $t_{\mathbf{IDIV}}(a, b)$ ограничивает сверху время выполнения для всех остальных делений. Поэтому можно утверждать, что в наихудшем случае *каждое* деление выполняется за время

$$t_{\mathbf{IDIV}}(a, b) = O\{L(b)[L(a) - L(b) + 1]\}.$$

Далее, для того чтобы вычислить верхнюю границу для числа n (целочисленных делений, необходимых для нахождения наибольшего общего делителя a и b), используем теорему Ламе, сформулированную ниже; по этой теореме число $5\cdot$(количество цифр в меньшем числе) является верхней оценкой числа n, и в нашем случае получаем $n \leq 5 \cdot L(b)$.

Таким образом, $t_{\mathbf{EA}}(a,b)$ = (число делений)·(время выполнения каждого деления) $\leq [5\cdot L(b)]\cdot\{L(b)[L(a)-L(b)+1]\}$, откуда получаем

$$t_{\mathbf{EA}}(a,b) = O\{L(b)^2[L(a)-L(b)+1]\}.$$

Теперь мы изложим теорему Ламе, являющуюся, по-видимому, самой первой теоремой, в которой речь идет о сложности вычислений.

Теорема Ламе (Lamé, 1844) (оценка наихудшего случая для алгоритма Евклида). Количество делений, которое нужно выполнить для нахождения наибольшего общего делителя двух целых чисел, не превосходит количества цифр в меньшем числе, умноженного на пять.

Доказательство. Рассмотрим последовательность Фибоначчи (см. историческое замечание 2)

$$1,1,2,3,5,8;13,21,34,55,89;144,233,377,610,987;1597,\ldots,$$

в которой каждый член равен сумме двух предыдущих. (Заметим, что 1 — единственное число, которое появляется дважды; для оставшейся части доказательства последовательность $\{1,1,2,3,5,8\}$ эквивалентна $\{1,2,3,5,8\}$.)

Нетрудно показать, что число членов в последовательности Фибоначчи, имеющих одинаковое количество цифр, не меньше четырех и не больше пяти. Действительно, если мы обозначим через t_1 первый член с $k+1$ цифрами, то очевидным образом выполняются неравенства

$$10^k < t_1 < 2\cdot 10^k,$$

так как t_1 есть сумма двух членов с k цифрами. Точно также, поскольку

$$(1/2)10^k < t_0 < 10^k$$

и $t_2 = t_1 + t_0$, имеем

$$(3/2)10^k < t_2 < 3\cdot 10^k.$$

Аналогичным образом получаем следующие неравенства:

$$\begin{aligned}(5/2)10^k &< t_3 < 5\cdot 10^k,\\ 4\cdot 10^k &< t_4 < 8\cdot 10^k,\\ (13/2)10^k &< t_5 < 13\cdot 10^k,\\ (21/2)10^k &< t_6 < 21\cdot 10^k.\end{aligned}$$

Поэтому группа членов, содержащих $(k+1)$ цифр, имеет не меньше четырех и не больше пяти элементов.

Если обозначить через $f_0, f_1, f_2, \ldots$ члены последовательности Фибоначчи, то число n членов, предшествующих f_n, не больше, чем $5\cdot$(количество цифр в f_n) минус один. Поэтому количество делений, которое нужно выполнить, чтобы найти наибольший общий делитель двух последовательных членов f_n, f_{n+1}, не превосходит количества цифр в f_n, умноженного на пять. Предположим теперь, что мы хотим найти наибольший общий делитель двух целых чисел a, b $(a > b)$. Обозначим через $r_{n'}, r_{n'-1}, \ldots, r_2, r_1, r_0$ убывающую последовательность остатков, полученную по алгоритму Евклида; имеем $r_i = qr_{i-1} + r_{i-2}$, $q \geq 1$. Предположим, кроме того, что целое число b лежит между f_{n-1} и f_n. Покажем, что остатки $r_{n'}, r_{n'-1}, \ldots, r_0$ будут содержаться в различных интервалах, образованных членами последовательности $f_{n+1}, f_n, f_{n-1}, \ldots, f_2, f_1, f_0$.

Сначала рассмотрим случай $q = 1$, т.е. частное равно единице во всех операциях деления. Если два остатка r_h и r_{h+1} попадут в один интервал (f_k, f_{k-1}), так что $f_k > r_h > r_{h-1} > f_{k-1}$, то (поскольку $f_k = f_{k-1} + f_{k-2}$ и f_{k-1} встречается в такой сумме только один раз) мы получим $r_h = r_{h-1} + r_{h-2}$ и $f_{k-2} > r_{h-2}$. Таким образом, интервал (f_{k-1}, f_{k-2}) не будет содержать остатков. Точно такое же заключение справедливо для случаев $f_k = r_h > r_{h-1} > f_{k-1}$ и $f_k > r_h > r_{h-1} = f_{k-1}$. Итак, если все частные в алгоритме Евклида равны 1, то остатки будут распределены таким образом (среди убывающей последовательности чисел Фибоначчи), что в каждом интервале будет не больше двух остатков и за каждым интервалом с двумя остатками будет следовать интервал, не содержащий остатков.

Теперь рассмотрим случай $q > 1$, т.е. при каком-то делении в алгоритме Евклида мы получим $r_i = 2\cdot r_{i-1} + r_{i-2}$ (наименьшее $q > 1$). Пусть f_{j+1} и f_j — два последовательных числа Фибоначчи, между которыми содержится r_i; тогда

$$r_i - 2r_{i-1} > 0, \qquad 2f_j - f_{j+1} > 0$$

и

$$2(f_j - r_{i-1}) - (f_{j+1} - r_i) > 0,$$

откуда следует, что $f_j > r_{i-1}$. Если r_{i-1} также меньше, чем f_{j-1}, то интервал (f_j, f_{j-1}) будет пустым. С другой стороны, если $r_{i-1} > f_{j-1}$, то, так как $f_{j+1} = 2f_{j-1} + f_{j-2}$, $r_i = 2r_{i-1} + r_{i-2}$ и $r_i < f_{j+1}$, должно выполняться неравенство $f_{j-2} > r_{i-2}$, т.е. интервал (f_{j-1}, f_{j-2}) будет пустым. Таким образом, если частное в алгоритме Евклида

отлично от единицы, то найдется хотя бы один интервал в последовательности Фибоначчи, который не будет содержать остатков, и это не восполнится интервалом с двумя остатками.

Итак, для того чтобы последовательность остатков $r_{n'}$, $r_{n'-1}$, $\ldots, r_1, r_0$ имела ту же длину, что и последовательность f_n, f_{n-1}, $\ldots, f_1, f_0$, частные во всех операциях деления должны быть равными единице, так же как и r_0. Как и в случае последовательности Фибоначчи, где $f_0 = 1$, $f_1 = 2$, в последовательности остатков должно быть $r_0 = 1$; однако r_1 не может равняться двум, поскольку тогда эти две последовательности были бы одинаковыми и b было бы равно f_{n+1}, что не так. Следовательно, r_1 должно быть равно как минимум трем и последовательность остатков будет иметь строго меньшую длину, чем последовательность Фибоначчи. □

Ниже мы приведем еще одну оценку наихудшего случая для алгоритма Евклида (Абрамов, 1979; Wilf, 1986). Мы воспользуемся следующей леммой.

Лемма 2.2.9. Если $a \geq b \geq 1$, то $\mathbf{MOD}(a, b) \leq (a-1)/2$.

Доказательство. По определению $\mathbf{MOD}(a, b) = a - \lfloor a/b \rfloor \cdot b \leq a - b$ и, очевидно, $\mathbf{MOD}(a, b) \leq b - 1$. Поэтому $\mathbf{MOD}(a, b) \leq \min(a - b, b - 1)$, и для доказательства леммы нужно рассмотреть два случая:

i. $b \leq (a+1)/2$. Тогда $b - 1 \leq a - b$ и $\mathbf{MOD}(a, b) \leq b - 1 \leq (a+1)/2 - 1 = (a-1)/2$.

ii. $b > (a+1)/2$. Тогда $a - b \leq b - 1$ и $\mathbf{MOD}(a, b) \leq a - b < a - (a+1)/2 = (a-1)/2$. □

Теорема 2.2.10 (другая оценка наихудшего случая для алгоритма Евклида). Пусть a и b — целые положительные числа и $\mathrm{M} = \max(a, b)$. Количество делений, которое нужно выполнить для нахождения наибольшего общего делителя a и b, не превосходит $\lfloor 2 \log_2 M \rfloor + 1$.

Доказательство. Без потери общности можно считать, что $a \geq b$. В алгоритме Евклида порождается убывающая последовательность неотрицательных целых чисел a_0, a_1, $\ldots$, a_k, где $a_0 = a$, $a_1 = b$ и $a_i = \mathbf{MOD}(a_{i-2}, a_{i-1})$, $i \geq 2$. По лемме 2.2.9 имеем $a_i \leq (a_{i-2} - 1)/2 \leq a_{i-2}/2$. Индукцией по i получаем $a_{2i} \leq a_0/2^i$ и $a_{2i+1} \leq a_1/2^i$, $i \geq 0$, и поэтому $a_k \leq 2^{-\lfloor k/2 \rfloor} M$. Алгоритм останавливается, когда $a_k < 1$; это происходит, когда $2^{-\lfloor k/2 \rfloor} M < 1$, т.е. $k > 2 \log_2 M$. □

Пример. Оценим число делений, необходимое для вычисления (144, 89) и (21, 13). По теореме Ламе в обоих случаях число делений меньше, чем $5 \cdot 2 = 10$; используя теорему 2.2.10, получаем, что число делений в первом случае не больше 15, во втором — не больше 9; на самом деле (144, 89) вычисляется после 9 делений, а (21, 13) — после 5.

Приведем в заключение интересный результат Дирихле (Dirichlet, 1849), утверждающий, что если a и b — два случайно выбранных целых числа, то вероятность того, что $\gcd(a, b) = 1$, равна $6/\varpi^2 \approx 0.60793$. (Другие результаты по этой тематике можно найти в работах (Абрамов, Рыбин, 1988; Bradley, 1970; Collins, 1974; Knuth, 1969; Lipson, 1981; Motzkin, 1949 и Schroeder, 1986).)

2.2.3. Расширенный алгоритм Евклида

Для различных приложений очень важно уметь представлять наибольший общий делитель целых чисел a и b в виде $\gcd(a, b) = ax + by$ (теорема 2.2.3). Очевидно, один из способов состоит в применении алгоритма Евклида и затем «обратного» прохода. Например, для $a = 612$ и $b = 342$ получаем

$$\begin{aligned}
612 &= 342 \cdot 1 + 270, && \text{или} && 270 = 612 - 342,\\
342 &= 270 \cdot 1 + 72, && \text{или} && 72 = 342 - 270,\\
270 &= 72 \cdot 3 + 54, && \text{или} && 54 = 270 - 72 \cdot 3,\\
72 &= 54 \cdot 1 + 18, && \text{или} && 18 = 72 - 54,\\
54 &= 18 \cdot 3 + 0, &&&&
\end{aligned}$$

откуда следует, что 18 — наибольший общий делитель чисел 612 и 342. Проведя вычисления в обратном порядке, получим

$$\begin{aligned}
18 &= 72 - 54 = (342 - 270) - (270 - 3 \cdot 72)\\
&= [342 - (612 - 342)] - [8612 - 342) - 3 \cdot (342 - 270)]\\
&= [342 - (612 - 342)] - \{(612 - 342) - 3 \cdot [342 - (612 - 342)]\}\\
&= 9 \cdot 342 + (-5) \cdot 612,
\end{aligned}$$

т.е. $18 = 9 \cdot 342 + (-5) \cdot 612$, и мы решили нашу задачу.

Другой подход к решению этой задачи, имеющий, как мы убедимся в дальнейшем, много приложений, состоит в применении *расширенного алгоритма Евклида*. Значения x и y вычисляются в серии шагов,

в каждом из которых мы выражаем a_i (вычисленное в процессе работы алгоритма Евклида, разд. 2.2.2) в форме $ax_i + by_i$. А именно рассмотрим последовательность

$$\begin{aligned}
&a_0 = a, && a_0 = ax_0 + by_0,\\
&a_1 = b, && a_1 = ax_1 + by_1,\\
&a_2 = a_0 - a_1q_1, && a_2 = ax_2 + by_2,\\
&a_3 = a_1 - a_2q_2, && a_3 = ax_3 + by_3,\\
&\ldots && \ldots\\
&a_i = a_{i-2} - a_{i-1}q_{i-1}, && a_i = ax_i + by_i,\\
&\ldots && \ldots\\
&a_k = a_{k-2} - a_{k-1}q_{k-1}, && a_k = ax_k + by_k,\\
&0 = a_{k-1} - a_kq_k, && 0 = ax_{k+1} + by_{k+1}.
\end{aligned}$$

В левом столбце — последовательность делений, которая нам уже встречалась раньше и которая теперь разрешена относительно остатков. В правом столбце каждый остаток выражен в виде $ax_i + by_i$; мы хотим вычислить x_i и y_i. Очевидно, что $x_0 = 1$, $y_0 = 0$ и $x_1 = 0$, $y_1 = 1$. Сравнивая обе части на i-м шаге, имеем $a_i = ax_i + by_i = a_{i-2} - a_{i-1}q_{i-1} = (ax_{i-2} + by_{i-2}) - (ax_{i-1} + by_{i-1})q_{i-1} = a(x_{i-2} - x_{i-1}q_{i-1}) + b(y_{i-2} - y_{i-1}q_{i-1})$, откуда получается следующая индуктивная процедура вычисления x_i и y_i:

$$\begin{aligned}
q_{i-1} &= \mathbf{QUO}(a_{i-2}, a_{i-1}),\\
a_i &= a_{i-2} - a_{i-1}q_{i-1},\\
x_i &= x_{i-2} - x_{i-1}q_{i-1},\\
y_i &= y_{i-2} - y_{i-1}q_{i-1}.
\end{aligned}$$

Конечно, мы можем также вычислить a_i как $\mathbf{MOD}(a_{i-2}, a_{i-1})$, но приведенное выше выражение подчеркивает, что a_i вычислено таким же способом, как x_i и y_i. Мы получили следующий алгоритм.

XEA. Расширенный алгоритм Евклида (E**x**tended **E**uclidean **A**lgorithm)

Вход: a и $b \neq 0$.

Выход: d, x, y, такие, что $d = \gcd(a, b) = ax + by$.

1. [Инициализация] $(a_0, a_1) := (a, b)$; $(x_0, x_1) := (1, 0)$; $(y_0, y_1) := (0, 1)$.

2. [Основной цикл] Пока $a_1 \neq 0$, выполнять { $q := \mathbf{QUO}(a_0, a_1)$; $(a_0, a_1) := (a_1, a_0 - a_1q)$; $(x_0, x_1) := (x_1, x_0 - x_1q)$; $(y_0, y_1) := (y_1, y_0 - y_1q)$; }.

3. [Выход] Вернуть $(d, x, y) := (a_0, x_0, y_0)$.

Анализ времени работы **EXA** аналогичен проведенному для **EA**, детали мы оставляем читателю. Применяя расширенный алгоритм Евклида в нашем примере $a = 342$, $b = 612$, получим:

Итерация	q	a_0	a_1	x_0	x_1	y_0	y_1
0	–	342	612	1	0	0	1
1	0	612	342	0	1	1	0
2	1	342	270	1	−1	0	1
3	1	270	72	−1	2	1	−1
4	3	72	54	2	−7	−1	4
5	1	54	18	−7	9	4	−5
6	3	18	0	9	−34	−5	19

Заметим, что равенство $a_0 = ax_0 + by_0$ выполняется на каждом шаге. Алгоритм выдает $d = 18$, $x = 9$, $y = -5$; проверим ответ: $18 = 342 \cdot (9) + 612 \cdot (-5)$.

2.2.4. Алгоритм Евклида и цепные дроби

Цепные дроби играют важную роль во многих областях математики. В гл. 7 этой книги мы увидим, как их можно использовать для построения очень эффективного алгоритма отделения и аппроксимации вещественных корней полиномов с целыми коэффициентами. В этом разделе мы введем цепные дроби, используя алгоритм Евклида (Olds, 1963; Richards, 1981).

Рассмотрим произвольную рациональную дробь a_0/a_1, записанную в несократимом виде, т.е. $(a_0, a_1) = 1$ и $a_1 > 0$. Применив к паре a_0, a_1 алгоритм Евклида, получим

$$\begin{aligned}
&a_0 = a_1 c_0 + a_2, && 0 < a_2 < a_1,\\
&a_1 = a_2 c_1 + a_3, && 0 < a_3 < a_2,\\
&a_2 = a_3 c_2 + a_4, && 0 < a_4 < a_3,\\
&\ldots\\
&a_{k-2} = a_{k-1} c_{k-2} + a_k, && 0 < a_k < a_{k-1},\\
&a_{k-1} = a_k c_{k-1}.
\end{aligned}$$

Эти обозначения несколько отличаются от тех, которые мы использовали в разд. 2.2.2; а именно мы заменили $q_1, \ldots, q_k$ на $c_0, \ldots, c_{k-1}$. Если мы запишем ξ_i вместо a_i/a_{i+1} для всех i в пределах $0 \le i \le k-1$, приведенные выше равенства примут вид

$$\xi_i = c_i + \frac{1}{\xi_{i+1}}, \quad 0 \le i \le k-2, \quad \xi_{k-1} = c_{k-1}.$$

Если в $\xi_0 = c_0 + 1/\xi_1$ заменить ξ_1 на $c_1 + 1/\xi_2$, то получится $\xi_0 = c_0 + 1/(c_1 + 1/\xi_1)$. Продолжая этот процесс, мы получим

$$\frac{a_0}{a_1} = \xi_0 = c_0 + \cfrac{1}{c_1 + \cfrac{1}{\ddots + \cfrac{1}{c_{k-2} + \cfrac{1}{c_{k-1}}}}}.$$

Это — представление a_0/a_1 в виде *цепной дроби*. Целые числа c_i называются *неполными частными*. Так как в общем случае a_0 может не быть положительным (мы предполагаем, что $a_1 > 0$), c_0 может быть положительным, отрицательным или нулевым. Однако, поскольку в алгоритме Евклида $0 < a_2 < |a_1|$, частное c_1 положительно и точно также положительны $c_2, \ldots, c_{k-1}$. Мы будем использовать обозначение $(c_0; c_1, \ldots, c_{k-1})$ для цепной дроби, приведенной выше.

Пример. Рассмотрим рациональную дробь $8/5$. Нетрудно видеть, что $8/5 = (1; 1, 1, 2)$. Более того, можно проверить, что $8/5 = (1; 1, 1, 1, 1)$. Оказывается, всякое рациональное число имеет только два представления в виде цепной дроби. В общем случае,

$$\frac{a_0}{a_1} = (c_0; c_1, \ldots, c_{k-2}, c_{k-1}) = (c_0; c_1, \ldots, c_{k-2}, c_{k-1} - 1, 1).$$

Замечание. Наше условие, что $c_1, c_2, \ldots, c_{k-1}$ положительны, не является общепринятым. Если отказаться от него, то дробь $-8/5$ может также быть представлена в виде цепных дробей $(-2; 2, 2)$ и $(-1; -1, -1, -2)$.

Цепные дроби могут быть конечными или бесконечными; например, цепная дробь, представляющая число $8/5$, конечна. Следующие две теоремы устанавливают некоторые свойства конечных цепных дробей. Нам необходимо следующее определение.

Определение 2.2.11. Целой частью $[c]$ числа c называется

$$[c] = \begin{cases} \lfloor c \rfloor, & \text{если } c \geq 0, \\ \lceil c \rceil, & \text{если } c < 0. \end{cases}$$

Теорема 2.2.12 (единственность). Если $(c_0; c_1, \ldots, c_m) = (d_0; d_1, \ldots, d_n)$ и если $c_m > 1$ и $d_n > 1$, то $m = n$ и $c_i = d_i$ для $i = 0, 1, \ldots, n$.

Доказательство. Используем индукцию. Определим числа $s_i = (c_i; c_{i+1}, \ldots, c_m)$ и $t_i = (d_i; d_{i+1}, \ldots, d_n)$. Очевидно, что $s_i = (c_i; c_{i+1}, \ldots, c_m) = c_i + 1/s_{i+1}$ и $t_i = (d_i; d_{i+1}, \ldots, d_n) = d_i + 1/t_{i+1}$. Заметим, что $s_i > c_i$, $s_i > 1$ для $i = 1, 2, \ldots, m-1$, $s_m = c_m > 1$ и $t_i > d_i$, $t_i > 1$ для $i = 1, 2, \ldots, n-1$, $t_n = d_n > 1$; более того, $c_i = [s_i]$ и $d_i = [t_i]$ для всех i в соответствующих пределах. По условию теоремы $s_0 = t_0$, и, рассматривая целые части, имеем $c_0 = [s_0] = [t_0] = d_0$. По определению s_1, t_1 получаем $1/s_1 = s_0 - c_0 = t_0 - d_0 = 1/t_1$, откуда вытекает, что $s_1 = t_1$ и $c_1 = [s_1] = [t_1] = d_1$. Предоставим читателю завершить шаг индукции: из того, что $s_i = t_i$ и $c_i = d_i$, вытекает, что $s_{i+1} = t_{i+1}$ и $c_{i+1} = d_{i+1}$. Кроме того, m должно быть равным n. Чтобы убедиться в этом, предположим без потери общности, что $m < n$. Тогда из предыдущего рассуждения следует, что $s_m = t_m$ и $c_m = d_m$. Однако это противоречит тому, что $s_m = c_m$ и $t_m > d_m$. □

Теорема 2.2.13. Любая конечная цепная дробь представляет рациональное число, и, обратно, всякое рациональное число может быть представлено в виде конечной цепной дроби, причем ровно двумя способами.

Доказательство. Первая часть доказывается индукцией по числу членов в цепной дроби при помощи формулы $(c_0; c_1, \ldots, c_m) = c_0 + 1/(c_1; c_2, \ldots, c_m)$. Вторая часть следует из возможности представления любого рационального числа в виде цепной дроби и теоремы 2.2.12. □

До сих пор мы имели дело только с рациональными числами и конечными цепными дробями, а что можно сказать об иррациональных числах и их разложениях? Некоторые очень важные свойства разложений иррациональных чисел в цепные дроби объединены в теореме 2.2.14, которую мы приводим без доказательства.

Теорема 2.2.14. Любое иррациональное число ξ представимо единственным образом в виде бесконечной цепной дроби $(c_0; c_1, \ldots, c_n, \ldots)$, где значения c_i вычисляются с помощью следующего алгоритма:

> Положим $\xi_0 = \xi$ и определим по индукции $c_i = \lfloor \xi_i \rfloor$ и $\xi_{i+1} = 1/(\xi_i - c_i)$, $i \geq 0$.

Обратно, всякая бесконечная цепная дробь, заданная числами c_i, $c_i > 0$ для любого i, представляет иррациональное число ξ. Более того, если мы положим

$$p_{-2} = 0, \quad p_{-1} = 1, \quad p_i = c_i p_{i-1} + p_{i-2}, \quad i \geq 0,$$
$$q_{-2} = 0, \quad q_{-1} = 1, \quad q_i = c_i q_{i-1} + q_{i-2}, \quad i \geq 0,$$

то конечная цепная дробь $(c_0; c_1, \ldots, c_n)$ будет иметь рациональное значение $r_n = p_n/q_n$, $(p_n, q_n) = 1$, которое называется *n-й подходящей дробью* иррационального числа ξ. Знаменатели q_n подходящих дробей образуют возрастающую последовательность положительных целых чисел для $n > 0$, и выполняются следующие соотношения:

a. Если $\xi = (c_0; c_1, \ldots, c_{n-1}, \xi_n)$, где $\xi_n = (c_n; c_{n+1}, \ldots)$, $n \geq 0$, то

$$\xi = \frac{p_{n-1}\xi_n + p_{n-2}}{q_{n-1}\xi_n + q_{n-2}};$$

b. $$\xi = c_0 + \cfrac{1}{c_1 + \cfrac{1}{\ddots + \cfrac{1}{c_{n-1} + \cfrac{1}{\xi_n}}}}, \qquad n \geq 1;$$

c. $$\frac{p_n}{q_n} = c_0 + \cfrac{1}{c_1 + \cfrac{1}{\ddots + \cfrac{1}{c_{n-1} + \cfrac{1}{c_n}}}}, \qquad n \geq 0.$$

Наконец, всякая периодическая цепная дробь является *квадратичным* иррациональным числом, и обратно. (Квадратичным иррациональным числом называется число вида $(p \pm \sqrt{d})/q$, являющееся корнем квадратного полиномиального уравнения $q^2x^2 - 2pqx + (p^2 - d)$, где d — целое положительное число, не являющееся точным квадратом.)

Отметим, что алгоритм Евклида может быть использован только для разложения в цепную дробь рационального числа; из теоремы 2.2.14 видно, что имеется более общая процедура, которая может быть использована как для рационального, так и для иррационального числа. А именно пусть нам дано число x (рациональное или иррациональное). Вычислим c_0, наибольшее целое число, не превосходящее x, и представим x в форме $x = c_0 + 1/x_1$, $0 < 1/x_1 < 1$, где число $x_1 = 1/(x - c_0) > 1$ иррационально, если x иррационально. После этого вычислим c_1, наибольшее целое число, не превосходящее x_1, и представим x_1 в форме $x_1 = c_1 + 1/x_2$, $0 < 1/x_2 < 1$, $c_1 \geq 1$, где снова число $x_2 = 1/(x_1 - c_1) > 1$ может быть иррациональным. Продолжая этот процесс, мы получим представление x в виде цепной

дроби $x = (c_0; c_1, \ldots)$, которая может быть конечной или бесконечной в зависимости от того, является x рациональным или нет. Примеры будут приведены ниже, но сначала мы докажем следующую теорему.

Теорема 2.2.15. Рассмотрим бесконечную цепную дробь $\xi = (c_0; c_1, \ldots)$, и пусть $r_n = p_n/q_n$ — ее n-я подходящая дробь. Справедливы следующие утверждения:

a. $p_n q_{n-1} - p_{n-1} q_n = (-1)^{n-1}$, $\quad n \geq 1$.

b. $r_n - r_{n-1} = (-1)^{n-1}/q_n q_{n-1}$, $\quad n \geq 1$.

c. $r_n - r_{n-2} = (-1)^{n-2} c_n / q_n q_{n-2}$, $\quad n \geq 2$.

d. Для четных значений n последовательность n-х подходящих дробей монотонно возрастает, и ее предел равен ξ; для нечетных значений n соответствующая последовательность монотонно убывает, ее предел также равен ξ; кроме того, каждое r_{2n} меньше, чем r_{2n-1}, и каждая подходящая дробь r_n, $n \geq 2$, лежит между двумя предыдущими подходящими дробями.

Доказательство.

a. Доказательство будем вести по индукции. Для $n = 1$, пользуясь теоремой 2.2.14, получаем $p_1 q_0 - p_0 q_1 = (c_1 p_0 + p_{-1}) \cdot 1 - c_0 c_1 = (c_1 c_0 + 1) - c_1 c_0 = 1$; поэтому равенство справедливо; отметим, что $p_0 = c_0$, $q_0 = 1$ и $q_1 = c_1 q_0 \geq q_0$. Допустим теперь, что равенство выполняется для $n = k$, т.е. $p_k q_{k-1} - p_{k-1} q_k = (-1)^{k-1}$. Мы покажем, что оно справедливо и для $k + 1$. Пользуясь опять теоремой 2.2.14, получим $p_{k+1} q_k - p_k q_{k+1} = (c_{k+1} p_k + p_{k-1}) q_k - p_k (c_{k+1} q_k - q_{k-1}) = -(p_k q_{k-1} - p_{k-1} q_k) = -(-1)^{k-1} = (-1)^k$.

b. Для доказательства этого пункта разделим обе части равенства $p_n q_{n-1} - p_{n-1} q_n = (-1)^{n-1}$ на $q_n q_{n-1}$, после чего получим $p_n/q_n - p_{n-1}/q_{n-1} = (-1)^{n-1}/q_n q_{n-1}$. Требуемое равенство теперь следует из того, что $r_n = p_n/q_n$.

c. Здесь $r_n - r_{n-2} = p_n/q_n - p_{n-2}/q_{n-2} = (p_n q_{n-2} - p_{n-2} q_n)/q_n q_{n-2}$. Заменяя в числителе p_n и q_n соответствующими им выражениями (из теоремы 2.2.14) $c_n p_{n-1} + p_{n-2}$ и $c_n q_{n-1} + q_{n-2}$ соответственно, получаем $p_n q_{n-2} - p_{n-2} q_n = (c_n p_{n-1} + p_{n-2}) q_{n-2} - p_{n-2} (c_n q_{n-1} + q_{n-2}) = c_n (p_{n-1} q_{n-2} - p_{n-2} q_{n-1}) = (-1)^{n-2} c_n = (-1)^n c_n$.

d. Выведем этот пункт из предыдущих. Из пп. **b** и **c** следует, что $r_{2n} < r_{2n+2}$, $r_{2n+1} < r_{2n-1}$ и $r_{2n} < r_{2n-1}$, потому что q_n положительно для $n \geq 1$. Таким образом, $r_0 < r_2 < \ldots$ и $r_1 > r_3 > r_5 > \ldots$. Последовательность с четными индексами

монотонно возрастает и ограничена сверху числом r_1; аналогично, последовательность с нечетными индексами монотонно убывает и ограничена снизу числом r_0. Эти два предела должны быть одинаковыми, поскольку разность $r_n - r_{n-1}$ стремится к нулю при n, стремящемся к бесконечности, а целые числа q_n возрастают с ростом n. □

Следует отметить, что утверждения теоремы 2.2.15 были бы иными, если бы в теореме 2.2.14 мы определили целые числа p_n и q_n, начав с p_{-1} и q_{-1}, а не с p_{-2} и q_{-2}. А именно если мы положим

$$p_{-1} = 0, \quad p_0 = 1, \quad p_i = c_i p_{i-1} + p_{i-2}, \quad i \geq 1,$$
$$q_{-1} = 0, \quad q_0 = 1, \quad q_i = c_i q_{i-1} + q_{i-2}, \quad i \geq 1,$$

то разложение в цепную дробь начнется с целого числа c_1 вместо c_0, и индексы у последовательности подходящих дробей изменятся с четных на нечетные и обратно. Кроме того, равенство в п. **a** примет вид

$$p_n q_{n-1} - p_{n-1} q_n = (-1)^n, \qquad n \geq 0,$$

а пп. **b** и **c** будут справедливы для $n \geq 2$ и $n \geq 3$ соответственно. В гл. 7 книги мы будем использовать эту форму записи подходящих дробей.

Пример. Разложим рациональное число 144/89 в цепную дробь, используя алгоритм, описанный в теореме 2.2.14; обратим также внимание на то, как ведут себя подходящие дроби. В начале положим $\xi_0 = 144/89$, $c_0 = \lfloor 144/89 \rfloor = 1$, $p_{-2} = 0$, $p_{-1} = 1$, $q_{-2} = 1$ и $q_{-1} = 0$ и, используя соотношения $p_i = c_i p_{i-1} + p_{i-2}$ и $q_i = c_i q_{i-1} + q_{i-2}$, получим $p_0 = c_0$ и $q_0 = 1$; $p_0/q_0 = 1$ — это первая четная подходящая дробь для числа 144/89, аппроксимирующая это число снизу. Затем вычислим $\xi_1 = 1/(\xi_0 - c_0) = 89/55$, $c_1 = \lfloor 89/55 \rfloor = 1$ и, используя те же соотношения, $p_1 = 2$ и $q_1 = 1$; $p_1/q_1 = 2$ — первая нечетная подходящая дробь для числа 144/89, аппроксимирующая это число сверху. Далее имеем $\xi_2 = 1/(\xi_1 - c_1) = 55/34$, $c_2 = 1$, $p_2 = 3$, $q_2 = 2$, и $p_2/q_2 = 3/2$ — вторая четная подходящая дробь для числа 144/89, снова аппроксимирующая его снизу; отметим, что $p_0/q_0 < p_2/q_2$ и $p_0/q_0 < p_2/q_2 < p_1/q_1$. Продолжая этот процесс, мы получим $\xi_3 = 1/(\xi_2 - c_2) = 34/21$, $c_3 = 1$, $p_3 = 5$, $q_3 = 3$ и $p_3/q_3 = 5/3$ — вторая нечетная подходящая дробь для числа 144/89, аппроксимирующая его опять сверху; имеем $p_3/q_3 < p_1/q_1$ и также $p_0/q_0 < p_2/q_2 < p_3/q_3 < p_1/q_1$. Завершение этого примера мы оставляем читателю.

Теорема 2.2.16. Пусть ξ — иррациональное число, и пусть $\xi = (c_0; c_1, \ldots, c_{n-1}, \xi_n)$ — его разложение в цепную дробь, где $\xi_n = (c_n; c_{n+1}, \ldots)$, $n \geq 0$. Справедливы следующие утверждения:

a. Каждая подходящая дробь расположена ближе к ξ, чем предыдущая.

b. $1/(2q_{n+1}q_n) < |\xi - p_n/q_n| < 1/(q_{n+1}q_n) < 1/q_n^2$, $n \geq 0$.

c. Существует бесконечно много рациональных чисел вида p/q, $(p, q) = 1$, таких, что $|\xi - p/q| < 1/q^2$.

Доказательство.

a. По теореме 2.2.14 имеем $\xi = (p_{n-1}\xi_n + p_{n-2})/(q_{n-1}\xi_n + q_{n-2})$, откуда получаем $\xi(q_{n-1}\xi_n + q_{n-2}) = (p_{n-1}\xi_n + p_{n-2})$, или, перегруппировав члены, $\xi_n(\xi q_{n-1} - p_{n-1}) = -(\xi q_{n-2} - p_{n-2}) = -q_{n-2}(\xi - p_{n-2}/q_{n-2})$. Разделив обе части равенства на $\xi_n q_{n-1}$ и взяв абсолютную величину, получим $|\xi - p_{n-1}/q_{n-1}| = |q_{n-2}/\xi_n q_{n-1}| \cdot |\xi - p_{n-2}/q_{n-2}|$. Поскольку $\xi_n > 1$ для $n \geq 1$ и знаменатели q_n подходящих дробей образуют возрастающую последовательность положительных целых чисел, мы имеем $q_{n-1} > q_{n-2}$; поэтому $0 < |q_{n-2}/\xi_n q_{n-1}| < 1$, откуда вытекает, что $|\xi - p_{n-1}/q_{n-1}| < |\xi - p_{n-2}/q_{n-2}|$, или $|\xi - r_{n-1}| < |\xi - r_{n-2}|$ для $n \geq 2$.

b. Из п. **b** теоремы 2.2.15 получаем $|r_{n+1} - r_n| = 1/q_{n+1}q_n$, $n \geq 1$; кроме того, мы только что доказали, что ξ расположено ближе к r_{n+1}, чем к r_n, и, следовательно, $1/2q_{n+1}q_n < |\xi - p_n/q_n| < 1/q_{n+1}q_n < 1/q_n^2$, $n \geq 0$, где $1/q_{n+1}q_n < 1/q_n^2$, потому что $q_{n+1} > q_n$. (См. также рис. 2.2.1.)

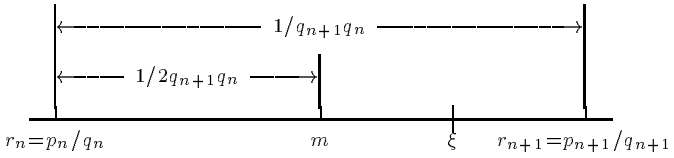


Рис. 2.2.1.
Геометрическое доказательство части **b** теоремы 2.2.16.

c. Этот пункт следует из того, что число ξ иррационально, и существует бесконечно много подходящих дробей p_n/q_n, удовлетворяющих п. **b**. □

Отметим, что неравенство $|x - p_n/q_n| < 1/q_n^2$ выполняется также и для рациональных чисел.

Пример. Вычислим несколько первых членов разложения числа ϖ в цепную дробь. Можно показать, что $\varpi = (3;\ 7,\ 15,\ 1,\ 292,\ 1,\ 1,\ 1,\ 2,\ 1,\ 3,\ 1,\ 14,\ 2, \ldots)$. Как уже было отмечено выше, для иррационального числа x не всегда можно получить его полное разложение в цепную дробь, поскольку алгоритм Евклида не может быть применен; если, однако, известно десятичное приближение числа x, то можно вычислить соответствующую часть цепной дроби, представляющей x. В нашем случае допустим, что нам достаточно приближения $\varpi = \xi_0 = 3.14159$. Используя алгоритм, описанный в теореме 2.2.14, получим

$$\begin{aligned}
3.14159 &= 3 + 0.14159, \quad & c_0 &= 3, \quad & \xi_1 &= 0.14159^{-1},\\
\frac{1}{0.14159} &= 7 + 0.06264, \quad & c_1 &= 7, \quad & \xi_2 &= 0.06264^{-1},\\
\frac{1}{0.06264} &= 15 + 0.96424, \quad & c_2 &= 15, \quad & \xi_3 &= 0.96424^{-1},\\
\frac{1}{0.96424} &= 1 + 0.03708, \quad & c_3 &= 1, \quad & \xi_4 &= 0.03708^{-1},\\
&\ldots
\end{aligned}$$

Итак, мы получили $\varpi = (3; 7, 15, 1, \ldots)$. Мы видим, что первые четыре неполные частные для числа ϖ совпадают с первыми четырьмя неполными частными для разложения в цепную дробь рационального числа 3.14159. Используя соотношения для подходящих дробей, получаем четыре первых приближения для ϖ: $r_0 = 3/1$, $r_1 = 22/7$, $r_2 = 333/106$ и $r_3 = 355/113$. Проверим теперь справедливость неравенства $|\xi - p/q| < 1/q^2$. Для r_2 имеем $|3.14159 - 333/106| < 1/106^2$, и после вычисления обеих частей получим $0.00007 < 0.000089$.

Отметим здесь открытие Эйлера: разложение числа e в цепную дробь обладает в отличие от ϖ замечательной регулярностью: $e = (2; 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, \ldots)$; доказательство достаточно сложно и выходит за рамки этой книги.

Заметим, что неполные частные чисел ϖ и e не периодичны и 1 встречается чаще, чем любое другое число. Интересный результат Кузьмина (Lang, Trotter, 1972) утверждает, что для почти всех чисел вероятность того, что в разложении его в цепную дробь n-е неполное частное c_n равно положительному целому числу j, дается формулой

$$\log_2 \frac{(j+1)^2}{j(j+2)}.$$

Это означает, что для почти всех чисел вероятность того, что $c_n = 1$, примерно равна 0.41.

В конце этой главы рассмотрим бесконечные периодические цепные дроби, представляющие иррациональные числа (см. теорему 2.2.14); предположим, например, что $\xi = (2; 3, 2, 3, 2, 3, \ldots)$. Что из себя представляет ξ? Заметим, что в нашем случае мы можем записать $\xi = (2; 3, \xi_2)$ согласно теореме 2.2.14; далее, поскольку $\xi = \xi_2 = \xi_4, \ldots$, имеем $\xi = (2; 3, \xi)$. Отсюда следует, что $\xi = 2 + 1/(3 + 1/\xi)$, или $3\xi^2 - 6\xi - 2 = 0$. Итак, ξ удовлетворяет квадратному уравнению, и, отбросив отрицательный корень, мы получим $\xi = (3 + \sqrt{15})/3$ — *квадратичное иррациональное число.*

Пример. Пусть $\xi = \sqrt{7}$; вычислим его разложение в цепную дробь. Используя алгоритм из теоремы 2.2.14, мы получим

$$\begin{aligned}
&\sqrt{7}=2+(\sqrt{7}-2), && c_0=2, \quad \xi_1=(\sqrt{7}-2)^{-1},\\
&1/(\sqrt{7}-2)=(\sqrt{7}+2)/3=1+(\sqrt{7}-1)/3, && c_1=1, \quad \xi_2=[(\sqrt{7}-1)/3]^{-1},\\
&3/(\sqrt{7}-1)=(\sqrt{7}+1)/2=1+(\sqrt{7}-1)/2, && c_2=1, \quad \xi_3=[(\sqrt{7}-1)/2]^{-1},\\
&2/(\sqrt{7}-1)=(\sqrt{7}+1)/3=1+(\sqrt{7}-2)/3, && c_3=1, \quad \xi_4=[(\sqrt{7}-2)/3]^{-1},\\
&3/(\sqrt{7}-2)=\sqrt{7}+2=4+(\sqrt{7}-2), && c_4=4, \quad \xi_5=(\sqrt{7}-2)^{-1}.
\end{aligned}$$

Таким образом, мы получили $\xi_5 = \xi_1$ и $\sqrt{7} = (2; 1, 1, 1, 4, 1, 1, 1, 4, \ldots)$.

2.3

Разложение целых чисел на множители

Среди всех задач теории чисел, при решении которых применялся компьютер, ни одна другая, по-видимому, не имела такого значения, как задача разложения на множители (факторизации). Из-за того, что эта задача является одной из основных в теории чисел и при этом просто формулируется, она привлекала людей еще в античности (Dickson, 1952). В самой простой формулировке: пусть нам дано целое число $n > 0$, и требуется, если это возможно, найти два целых числа a и b, таких, что $ab = n$. На самом деле здесь имеются две различные задачи: первая, называемая *тестом на простоту*, — это проверка того, существуют ли такие a и b, вторая, называемая *разложением*, — это задача их нахождения. В этой части мы рассмотрим обе эти задачи и попутно введем дополнительные математические понятия, которые нам понадобятся в дальнейшем.

2.3.1. Простые числа и решето Эратосфена

Начнем с определений. Ненулевое целое число $a \neq \pm 1$ называется *неразложимым*, если все его делители тривиальны, т.е. его делителями являются только числа ± 1 и $\pm a$. Например, числа 13 и -7 неразложимы. Ненулевое целое число $a \neq \pm 1$ называется *разложимым* или *составным*, если у него есть нетривиальные делители, т.е. оно может быть записано в виде $a = bc$, где b и c не равны ± 1 и $\pm a$. Например, $276 = 12 \cdot 23$ — составное число. Делители также называют *сомножителями*. Составное число может быть записано как произведение нетривиальных сомножителей. Если, в свою очередь, сомножители тоже разложимы, то они также могут быть записаны как произведения нетривиальных сомножителей, и так далее. Следующая теорема утверждает, что процесс этот рано или поздно обрывается.

Теорема 2.3.1 (существование неприводимого разложения). Всякое ненулевое целое число $a \neq \pm 1$ может быть записано как $\pm$ произведение конечного числа положительных неразложимых целых чисел, т.е $a = \pm u_1 \cdots u_r$, где $u_i > 1$ для $i = 1, \ldots, r$ и неразложимы.

Доказательство. Мы можем допустить, что $a > 0$. Если a не является уже неразложимым, то $a = bc$, где b и c не равны 1 и $1 < b, c < a$. Точно так же, если b и c разложимы, то их положительные нетривиальные сомножители строго меньше, чем b и c соответственно. В силу принципа полной упорядоченности процесс разложения рано или поздно останавливается, поскольку сомножители образуют строго убывающую последовательность положительных целых чисел. $\square$

Например, $1008 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 3 \cdot 7$. Является ли это разложение единственным, т.е. можно ли записать 1008 как произведение других неразложимых чисел? Перед тем как мы сможем ответить на этот вопрос, нам понадобится следующее определение. Целое число $p > 1$ называется *простым*, если для любых a, b из того, что $p \mid ab$, следует, что $p \mid a$ или $p \mid b$. Следующее предложение характеризует простые числа как положительные неразложимые элементы. Доказательство этого предложения иллюстрирует способ рассуждений, который потребуется для выполнения некоторых упражнений этой части книги.

Предложение 2.3.2. Целое число $p > 1$ является простым тогда и только тогда, когда оно неразложимо.

Доказательство. Предположим, что $p > 1$ просто и что $a \mid p$; нужно показать, что a — тривиальный делитель. Запишем $p = ab$ для некоторого b. Поскольку p просто, $p \mid a$ или $p \mid b$. Если $p \mid a$, то $a = \pm p$ (см. упр. 2 разд. 2.2.1). Если $p \mid b$, то $b = pc$ для некоторого c и $p = ab = apc$, что влечет за собой $1 = ac$; поэтому $a = \pm 1$. Итак, мы показали, что всякий делитель числа p тривиален, следовательно, p неразложимо. Обратно, предположим, что $p > 1$ неразложимо и что $p \mid ab$. Если p не делит a, то $(a, p) = 1$, поскольку p не имеет положительных делителей, отличных от 1 и p. По теореме 2.2.3 существуют x и y, такие, что $ax + py = 1$. Умножив это равенство на b, получим $abx + pby = b$, и, так как p делит левую часть, $p \mid b$. Мы показали, что, если $p \mid ab$ и p не делит a, то $p \mid b$, следовательно, p просто. □

Теорема 2.3.3 (единственность неприводимого разложения). Всякое ненулевое целое число $a \neq \pm 1$ может быть записано как $\pm$ произведение простых чисел только одним способом, с точностью до порядка сомножителей.

Доказательство. Предположим, что положительное целое число a записано в виде $a = u_1 \cdots u_r = v_1 \cdots v_s$, где u_i и v_j положительны и неразложимы и, следовательно, просты. Имеем $u_1 \mid v_1 \cdots v_s$ и, так как u_1 просто, $u_1 \mid v_{j_1}$ для некоторого j_1. Поскольку v_{j_1} не имеет нетривиальных делителей, $u_1 = v_{j_1}$. Продолжая, получаем, что u_2 делит произведение оставшихся v_j и, следовательно, $u_2 = v_{j_2}$ для некоторого j_2, и так далее. В конце концов мы получим, что $r = s$ и после некоторого переупорядочивания $u_i = v_i$ для всех i. □

Теорему 2.3.3 называют *основной теоремой арифметики*. Исторически математики рассматривали теорему 2.3.3 как «данную свыше», однако существуют математические системы, в которых утверждение о единственности разложения не выполняется. В качестве примера такой системы рассмотрим множество E, элементами которого являются положительные *четные* числа 2, 4, 6, 8, ... Отметим, что E замкнуто относительно умножения. Предположив, что все числа, которые мы знаем, являются элементами E, мы видим, что $8 = 2 \cdot 4$ — «составное» число, тогда как 14 — «простое», поскольку оно не является произведением двух или более «чисел»; более того, число 840 имеет два разложения на «простые» числа, а именно $840 = 2 \cdot 14 \cdot 30 = 6 \cdot 10 \cdot 14$, поэтому теорема о единственности разложения не выполняется.

Поскольку теорема 2.3.3 выполняется для целых чисел, $\mathbb{Z}$ является *областью с однозначным разложением на множители*; другие

примеры элементов этого специального класса колец мы рассмотрим позже.

Объединенные вместе, теоремы 2.3.1 и 2.3.3 утверждают, что всякое ненулевое целое число $a \neq \pm 1$ может быть однозначно представлено как произведение конечного числа простых сомножителей (единственность с точностью до порядка сомножителей). Это представление числа известно под названием *разложение числа a в произведение степеней простых чисел*, которое записывается так:

$$a = \pm p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k},$$

где теперь p_i — *различные* простые числа и e_i — положительные целые числа. Иногда, как мы вскоре убедимся, удобнее считать, что показатели степеней могут быть равными нулю.

В качестве первого применения приведенных выше результатов выразим d, наибольший общий делитель чисел a и b, в терминах простых сомножителей чисел a и b.

Пусть

$$a = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}, \quad b = p_1^{f_1} p_2^{f_2} \cdots p_k^{f_k},$$

где $e_i \geq 0$ и $f_i \geq 0$. Тогда, поскольку $p^m \mid p^n$, если и только если $m \leq n$, наибольший общий делитель чисел a и b выражается в виде

$$d = (a, b) = p_1^{\min(e_1, f_1)} p_2^{\min(e_2, f_2)} \cdots p_k^{\min(e_k, f_k)},$$

где, как обычно, $\min(i, j)$ обозначает наименьшее из чисел, стоящих в скобках. Аналогично, наименьшее общее кратное чисел a и b выражается в виде

$$[a, b] = p_1^{\max(e_1, f_1)} p_2^{\max(e_2, f_2)} \cdots p_k^{\max(e_k, f_k)},$$

где $\max(i, j)$ обозначает наибольшее из чисел, перечисленных в скобках. Используя теперь тот факт, что

$$|ab| = p_1^{e_1 + f_1} p_2^{e_2 + f_2} \cdots p_k^{e_k + f_k},$$

а также равенство $\min(a, b) + \max(a, b) = a + b$, мы можем легко получить результат теоремы 2.2.8: $[a, b] = |ab|/(a, b)$.

Применяя результат об однозначности разложения натурального числа в произведение простых сомножителей, мы получаем теорему 2.3.4.

Теорема 2.3.4. $\sqrt{2}$ — иррациональное число.

Доказательство. Допустим, что существуют два натуральных числа a и b, такие, что $\sqrt{2} = a/b$, $(a, b) = 1$. Тогда, используя разложение в произведение степеней простых чисел, мы видим, что в левую часть равенства $2b^2 = a^2$ простой множитель 2 входит с нечетной степенью, а в правую — с четной, что невозможно. □

Теорема 2.3.5 (Евклид). Существует бесконечно много простых чисел.

Доказательство. Предположим противное. Пусть существует лишь конечное число простых чисел $p_1, p_2, \ldots, p_m$. Рассмотрим число $n = p_1 \cdot p_2 \cdots p_m + 1$, которое либо является простым, и тогда оно будет новым простым числом, либо имеет простой сомножитель p. (Отметим, что первые несколько чисел n такого вида просты; например, $2 + 1 = 3$, $2 \cdot 3 + 1 = 7$, $2 \cdot 3 \cdot 5 + 1 = 31$, $2 \cdot 3 \cdot 5 \cdot 7 + 1 = 211$, $2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 + 1 = 2311$. Однако $2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 + 1 = 30031 = 59 \cdot 509$.) Если бы p было одним из перечисленных простых чисел p_i, то p делило бы произведение $p_1 \cdot p_2 \cdots p_m$ и, поскольку p делит $n = p_1 \cdot p_2 \cdots p_m + 1$, оно делило бы и разность этих чисел, т.е. единицу, что невозможно. Поэтому p должно быть новым простым числом. □

Теорема 2.3.6. Для произвольного положительного целого числа n существуют n последовательных (т.е. отличающихся на единицу) составных чисел. Другими словами, в последовательности простых чисел существуют сколь угодно большие промежутки.

Доказательство. Рассмотрим числа $(n + 1)! + 2$, $(n + 1)! + 3$, $\ldots$, $(n + 1)! + (n + 1)$, где $k! = 1 \cdot 2 \cdots k$. Очевидно, каждое из этих чисел составное, потому что $(n + 1)! + i$ делится на i, когда $2 \leq i \leq n + 1$. □

Как мы уже говорили, два целых числа m и n называются взаимно простыми, если $(m, n) = 1$; заметим, что m и n не обязаны быть простыми, как, например, в случае 9 и 14. Для заданного $n > 0$ через $\phi(n)$ обозначается количество положительных целых чисел m, таких, что $m \leq n$ и $(m, n) = 1$; ϕ называется *функцией Эйлера*. Заметим, что для всякого простого числа p мы имеем $\phi(p) = p - 1$. Например, $\phi(5) = 4$, потому что все целые числа 1, 2, 3 и 4 взаимно просты с числом 5.

Выведем формулу, которая выражает $\phi(n)$ в терминах разложения n в произведение степеней простых чисел; $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$. Нам нужно сосчитать все положительные целые числа m, $m \leq n$, которые не делятся на простые числа p_i, встречающиеся в разложении числа

n. Очевидно, имеется n возможностей для m; мы, однако, должны отбросить n/p_j, $j = 1, 2, \ldots, k$, из них, потому что столько чисел m делятся на p_j. Таким образом, остается $n - n/p_1 - n/p_2 - \cdots - n/p_k$ возможных значений m. Но теперь мы отбросили слишком много, потому что, например, числа, которые делятся одновременно на p_1 и p_2, были выброшены как минимум два раза. Скорректируем ошибку, прибавив к последнему выражению число $n/(p_1p_2) + n/(p_1p_3) + \cdots + n/(p_{k-1}p_k)$. Однако теперь мы прибавили слишком много, потому что целые числа, которые делятся, скажем, на $p_1p_2p_3$, были возвращены по меньшей мере дважды, и последнее выражение нужно снова скорректировать. Продолжая этот процесс, мы получим формулу

$$\begin{aligned}\phi(n) = n &- \frac{n}{p_1} - \frac{n}{p_2} - \cdots - \frac{n}{p_k} \\ &+ \frac{n}{p_1p_2} + \frac{n}{p_1p_3} + \cdots + \frac{n}{p_{k-1}p_k} \\ &+ \cdots + (-1)^k \frac{n}{p_1p_2\cdots p_k}.\end{aligned}$$

Читатель может проверить, что эту формулу можно записать в виде[1)]

$$\phi(n) = n\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\cdots\left(1 - \frac{1}{p_k}\right).$$

Пример. Для того чтобы вычислить $\phi(60)$, разложим сначала 60 в произведение степеней простых чисел: $60 = 2^2 \cdot 3 \cdot 5$. Затем, применив нашу формулу, получим

$$\phi(60) = 60(1 - 1/2)(1 - 1/3)(1 - 1/5) = 16.$$

Мы завершим эту главу обсуждением способов нахождения простых чисел. Как крупинки золота, простые числа находят обычно с помощью решета. Первое решето было предложено греческим математиком Эратосфеном из Кирен, который жил в третьем столетии до нашей эры. Идея решета Эратосфена замечательна по своей простоте.

[1)] Другая форма записи этой формулы:

$$\phi(n) = (p_1 - 1)p_1^{e_1-1}(p_2 - 1)p_2^{e_2-1}\cdots(p_k - 1)p_k^{e_k-1}.$$

Более строгое доказательство можно получить, используя китайскую теорему об остатках. — *Прим. перев.*

Для того чтобы найти все простые числа $\leq n$, запишем по порядку все целые числа от 2 до n. Затем вычеркнем все четные числа, кроме 2, поскольку они делятся на 2 и потому не являются простыми. Потом вычеркнем все числа, кратные 3, и так далее. После i-го прохода будут вычеркнуты все числа, которые делятся на первые i простых чисел $p_1, \ldots, p_i$. Первое число $x > p_i$, которое останется невычеркнутым, будет $i+1$-м простым числом. (Почему?) Затем будут вычеркнуты все числа, кратные p_{i+1}, и процесс остановится, когда в списке не останется невычеркнутых чисел, которые больше последнего найденного простого числа. Целые числа, которые остались в списке, прошли сквозь решето и являются простыми $\leq n$.

В следующем примере мы найдем все простые числа ≤ 60, в исходный список включены только нечетные числа ≥ 2:

2	3	5	7	9/	11	13	15/	17	19
21/	23	25\	27/	29	31	33/	35\	37	39/
41	43	45\	47	49—	51/	53	55\	57/	59

Числа, кратные 3, $\geq 3^2$, зачеркнуты символом «/», кратные 5, $\geq 5^2$, — символом «\», и кратные 7, $\geq 7^2$, — символом «—». Следующее после 7 простое число — это 11, но $11^2 > 60$, поэтому по причине, которая будет объяснена ниже, процесс останавливается, и невычеркнутые числа являются простыми ≤ 60. (См. также (Dudley, 1983) и (Mills, 1947).)

Мы можем использовать этот метод для проверки простоты заданного числа n, которая зависит от того, будет ли оно вычеркнуто. Приведенная выше процедура может быть сделана более эффективной, если учитывать, что у составного числа n обязательно есть простой делитель $\leq \sqrt{n}$. (Причина в том, что делители всегда составляют пары; если число имеет делитель, больший, чем квадратный корень из него, оно также должно иметь делитель, меньший чем этот корень.) Поэтому при вычеркивании чисел, кратных p_i, мы можем рассматривать только простые числа $\leq \sqrt{n}$, и процесс остановится, когда $p_i^2 \geq n$ для некоторого i. Однако, несмотря на это ускорение, приведенный метод не подходит для проверки наибольших из известных простых чисел. Например, для проверки 13395-значного числа $2^{44497} - 1$, простота которого была доказана в 1979 г., компьютеру, выполняющему миллион операций в секунду, потребовалось бы 10^{6684} лет для завершения работы, если бы он остановился только при достижении квадратного корня числа.

Для некоторых последующих приложений нам нужно уметь находить n наибольших простых чисел $\leq M$, где M — максимальное

целое число, представимое аппаратно на имеющемся в нашем распоряжении компьютере. Например, обычные размеры компьютерного слова — это 16 или 32 бита, и, отбрасывая знаковый бит, получаем, что наибольшие целые числа, представимые в таких машинах, равны $2^{15}-1=32767$ и $2^{31}-1=2147483647$ соответственно. Для нахождения таких простых чисел построим сначала таблицу простых чисел $\leq\sqrt{M}$, используя метод решета. Кратные каждого из этих простых чисел затем вычеркиваются из списка $L,\ L+1,\ \ldots,\ M$, где интервал (L,M) достаточно велик для того, чтобы содержать как минимум n простых чисел. Целые числа, оставшиеся невычеркнутыми, будут простыми.

Для того чтобы этот метод можно было реализовать практически, список простых чисел $\leq M$ должен быть не слишком велик и нужно выбрать интервал (L,M), который содержал бы по крайней мере n простых чисел и был бы не слишком большим для небольшого n. Замечательная теорема о количестве простых чисел может быть использована для решения обеих этих задач ((Erdoes, 1949; Goldstein, 1973; Levinson, 1969), см. также историческое замечание 3.) Пусть $\varpi(x)$ обозначает количество простых чисел $\leq x$; из примера с решетом, рассмотренного выше, мы видим, что $\varpi(60)=17$. В теореме утверждается, что $\lim_{x\to\infty}\varpi(x)/(x/\ln x)=1$. Для больших x величина $x/\ln x$ аппроксимирует $\varpi(x)$; на самом деле это оценка снизу. Грубо говоря, теорема утверждает, что из каждых $\ln M$ чисел одно является простым. Таким образом, интервал, достаточно превышающий по длине $n\cdot\ln M$, должен содержать n простых чисел. Далее, размер таблицы простых чисел примерно равен $\sqrt{M}/\ln\sqrt{M}$. Если $n=10$, то для 16-битной машины $10\cdot\ln 2^{25}\approx 104$, поэтому, скорее всего, хватит интервала длины 200. Таблица простых чисел должна содержать около $\sqrt{2^{15}}/\ln\sqrt{2^{15}}\approx 35$ простых чисел. В действительности $\varpi(\sqrt{2^{15}})=42$. Для 32-битной машины $10\cdot\ln 2^{31}\approx 214$, поэтому возьмем интервал длины 500. Размер таблицы простых чисел — примерно $\sqrt{2^{31}}/\ln\sqrt{2^{31}}\approx 4314$. Настоящий размер равен $\varpi(\sqrt{2^{31}})=4691$. Следующий алгоритм объединяет сформулированные выше соображения.

GENPR. Генерация простых чисел (**Gen**erate **Pr**imes)

Вход: Два целых числа k и m одинарной точности и одномерный массив A длины k; m — *нечетное* целое число ≥ 3.

Выход: Простые числа $p_1<p_2<\ldots<p_r$ одинарной точности, лежащие в замкнутом отрезке $[m,m+2k-2]$.

1. [Инициализация] $n := m + 2k - 2$; для $i := 1, 2, \ldots, k$ выполнять $A(i) := 1$; $d := 3$.

2. [Если $d^2 > n$, то получить простые числа и закончить работу] Если $d > [n/d]$, то перейти к шагу 6.

3. [Вычислить наименьшее положительное число j, такое, что $d \mid (m + 2j - 2)$ и $m + 2j - 2 \geq 3$] $r := \mathbf{MOD}(m, d)$; $j := 1$; если $r > 0$ и r четно, то $j := j + d - r/2$; если $r > 0$ и r нечетно, то $j := j + (d - r)/2$; если $m \leq d$, то $j := j + d$.

4. [Вычеркивание составных] Для $i := j$, $j + d$, $j + 2d, \ldots$ пока $j > k$ выполнять $A(j) := 0$.

5. [Изменение d] Если $\mathbf{MOD}(d, 6) = 1$, то $d := d + 4$, иначе $d := d + 2$; перейти к шагу 2.

6. [Получить простые числа] Для $i := k$, $k - 1$, $\ldots, 1$ выполнять {Если $A(i) = 1$, то выдать простое число $m + 2i - 2$ }; закончить работу.

Анализ времени работы алгоритма ***GENPR***. Все арифметические операции в алгоритме выполняются с короткими числами, следовательно, каждая операция выполняется за время порядка $O(1)$.

Шаги 1 и 6 выполняются только один раз, каждый из них включает k операций; поэтому время выполнения обоих шагов равно $O(k)$.

Шаги от 2 до 5 образуют цикл, для которого, как можно видеть из описания шага 2, число $\sqrt{n}$, где $n = m + 2k - 2$, является верхней оценкой количества повторений. Кроме того, при каждом выполнении тела цикла шаг 4 включает не более k операций, а шаги 2, 3 и 5 — одну или две операции. Поэтому весь цикл выполняется за время $O(k\sqrt{n})$.

Объединяя эти результаты и учитывая, что $O(k + k\sqrt{n}) = O(k\sqrt{n})$, получаем

$$t_{\mathbf{GENPR}}(A, k, m) = O(k\sqrt{n}),$$

где $n = m + 2k - 2$.

Пример. С помощью предыдущего алгоритма найдем все простые числа между 3 и 21 (т.е. в случае $m = 3$ и $k = 10$).

После первого шага $d = 3$ и $A(i) = 1$, $i = 1, \ldots, 10$. На шаге 2 условие окончания не выполняется, поэтому мы производим шаг 3, где мы получаем $r = 0$ и $j = 4$ (поскольку $m = d$). На шаге 4 мы изменяем массив A, присваивая $A(4) := 0$, $A(7) := 0$ и $A(10) := 0$, и на шаге 5 мы изменяем d, присваивая $d := 5$. Этим завершается первое выполнение цикла, который включает шаги от 2 до 5.

В начале второго выполнения цикла проверка условия окончания оказывается успешной, и мы переходим к шагу 6. На шаге 6 для $i = 10$ мы ничего не выдаем, поскольку $A(10) = 0$; однако для $i = 9$ $A(9) = 1$; поэтому мы выдаем простое число 19, и так далее. Таким путем будут выданы простые числа 19, 17, 13, 11, 7, 5 и 3.

2.3.2. Целые числа по модулю m и алгоритм в грекокитайской теореме об остатках

Вспомним, что при фиксированном $m > 1$ по определению $b \equiv_m a$ тогда и только тогда, когда $b - a = mq$ для некоторого q (иначе говоря, b содержится в арифметической прогрессии $\{\ldots, -3m+a, -2m+a, -m+a, a, m+a, 2m+a, 3m+a, \ldots\}$), или, на языке этой главы, тогда и только тогда, когда m делит $b - a$. Вместо записи $b \equiv_m a$ мы будем также использовать запись $b \equiv a \pmod{m}$, которая читается «b равно a по модулю m» или «b сравнимо с a по модулю m». Это обозначение восходит к Гауссу, и мы называем m модулями. Множество классов эквивалентности $\mathbb{Z}/\equiv_m$ обозначается также через $\mathbb{Z}_m$ и называется множеством *вычетов* или *целыми числами по модулю* m. Следующая теорема перечисляет основные свойства этого отношения эквивалентности и позволяет заключить, что важнейшие алгебраические свойства целых чисел переносятся на целые числа по модулю m.

Теорема 2.3.7.

a. Если $a \equiv b \pmod{m}$ и d делит m, то $a \equiv b \pmod{d}$.

b. Если $a \equiv b \pmod{m}$ и $a \equiv b \pmod{n}$, то $a \equiv b \pmod{[m,n]}$.

c. Если $a \equiv c \pmod{m}$ и $b \equiv d \pmod{m}$, то $a + b \equiv c + d \pmod{m}$, $a - b \equiv c - d \pmod{m}$ и $ab \equiv cd \pmod{m}$.

d. (Свойство сокращения.) Если $ab \equiv ac \pmod{m}$, то $b \equiv c \pmod{m/d}$, где $d = (a, m)$. В частности, если $(a, m) = 1$, то $ab \equiv ac \pmod{m}$ влечет за собой $b \equiv c \pmod{m}$.

Доказательство. Мы докажем часть **a**, а остальное предоставим читателю. Из $a \equiv b \pmod{m}$ следует, что $a - b = km$ для некоторого целого числа k. Если $d|m$, то d также делит km, откуда вытекает, что $d|(a-b)$. Следовательно, $a - b = k'd$ для некоторого целого числа k' и $a \equiv b \pmod{d}$. □

Пример. Из равенства $3 \cdot 4 \equiv 3 \cdot 6 \pmod{6}$ мы получаем $4 \equiv 6 \pmod{2}$, тогда как из $3 \cdot 4 \equiv 3 \cdot 9 \pmod{5}$ получаем $4 \equiv 9 \pmod{5}$.

Полезная характеризация областей целостности дается следующим правилом сокращения.

Теорема 2.3.8. Нетривиальное коммутативное кольцо D является областью целостности тогда и только тогда, когда из условий $ab = ac$, $a \neq 0$ следует, что $b = c$.

Доказательство. Докажем теорему только в одну сторону, оставив обратную импликацию читателю. Пусть D — область целостности и a, b — элементы из D; тогда из равенства $ab = 0$ вытекает, что $a = 0$ или $b = 0$. Из того, что $ab = ac$, $a \neq 0$, следует, что $a(b - c) = 0$, $a \neq 0$, и, поскольку в D нет делителей нуля, то $b = c$. □

Рассмотрим классы эквивалентности $\mathbf{a}$ и $\mathbf{b}$ в $\mathbb{Z}_m$; мы определим сумму и произведение $\mathbf{a} + \mathbf{b}$ и $\mathbf{a} \cdot \mathbf{b}$ этих классов через сумму и произведение их представителей. А именно мы имеем сюръективное отображение $s\colon \mathbb{Z} \to \mathbb{Z}_m$; положим $\mathbf{a} + \mathbf{b} = s(a + b)$ и $\mathbf{a} \cdot \mathbf{b} = s(a \cdot b)$. Из теоремы 2.3.7 вытекает, что это определение корректно, т.е. если $c \in \mathbf{a}$ и $d \in \mathbf{b}$ — другие представители, то мы получим те же самые классы эквивалентности для суммы и произведения, поскольку $s(c+d) = s(a+b)$ и $s(c \cdot d) = s(a \cdot b)$. Отношение эквивалентности, которое сохраняет алгебраические свойства (такое, как $\equiv_m$, сохраняющее сложение и умножение), называется *конгруэнцией*.

Используя свойство евклидовости, мы можем рассматривать арифметику целых чисел по модулю m как *арифметику остатков* или *модулярную арифметику*. *Полная система остатков* по модулю m состоит из m целых чисел, по одному представителю из каждого класса эквивалентности. Чаще всего используются следующие две системы: система *неотрицательных* остатков по модулю m, состоящая из чисел $0, 1, 2, \ldots, m-1$, и система *наименьших по абсолютной величине* остатков, или *симметричная* система остатков, состоящая из чисел $0, \pm 1, \pm 2, \ldots, \pm(m-1)/2$ для нечетного числа m.

Запишем целое число a как $a = mq + r$, где $0 \leq r < m$. Остаток r, который иногда обозначается через $r_m(a)$ или $r(a)$, называется *остатком по модулю* m. Следующее предложение утверждает, что два целых числа конгруэнтны в точности тогда, когда они имеют одинаковые остатки по модулю m.

Предложение 2.3.9. $b \equiv a \pmod{m}$ тогда и только тогда, когда $r_m(a) = r_m(b)$.

Доказательство. Заметим, что для всякого a выполняется $r(a) \equiv a \pmod{m}$. По транзитивности отсюда следует, что $b \equiv a \pmod{m}$

тогда и только тогда, когда $r_m(a) \equiv r_m(b)$, т.е. тогда и только тогда, когда $m \mid [r(b) - r(a)]$. Но поскольку $0 \leq r(b), r(a) < m$, m делит $r(a) - r(b)$ в точности тогда, когда $r(b) = r(a)$. □

Это предложение утверждает, что для целого числа a его класс эквивалентности $\mathbf{a} = a + m\mathbb{Z}$ является в точности множеством чисел, остатки которых совпадают с $r(a)$. Остатки $0, 1, \ldots, m-1$ являются представителями классов эквивалентности. Допуская вольность обозначений, мы иногда отождествляем класс эквивалентности с представляющим его остатком и рассматриваем $\mathbb{Z}_m$ просто как множество $\{0, 1, \ldots, m-1\}$.

В модулярной арифметике мы работаем с остатками по модулю m. Противоположный (аддитивный обратный) элемент к произвольному числу a из $\mathbb{Z}_m$ — это $m - a$, однако мультипликативный обратный элемент к a, определяемый как решение уравнения $ax \equiv 1 \pmod{m}$, существует не всегда.

Теорема 2.3.10. Пусть $a \in \mathbb{Z}_m$. Тогда a имеет мультипликативный обратный по модулю m элемент в том и только том случае, когда $(a, m) = 1$.

Доказательство. С помощью расширенного алгоритма Евклида мы можем найти целые числа x и y, такие, что $(a, m) = ax + my$, откуда вытекает, что $ax \equiv (a, m) \pmod{m}$. Если $(a, m) = 1$, то предыдущее сравнение означает, что x является мультипликативным обратным к a по модулю m. Если же $(a, m) > 1$, то не существует числа x, для которого выполняется сравнение $ax \equiv 1 \pmod{m}$, поскольку $ax = 1 + km$ влечет за собой $(a, m) = 1$. □

Теорема 2.3.11. Для всякого целого числа $m > 1$ множество $\mathbb{Z}_m = \{0, 1, 2, \ldots, m-1\}$ с операциями умножения и сложения по модулю m является коммутативным кольцом с единицей и называется кольцом вычетов по модулю m. Оно является полем тогда и только тогда, когда m — простое число.

Доказательство. С помощью определенного ранее сюръективного отображения $s: \mathbb{Z} \leftarrow \mathbb{Z}_m$, а также определения сложения и умножения в $\mathbb{Z}_m$ мы легко можем вывести, что $\mathbb{Z}_m$ является коммутативным кольцом с единицей, из того, что $\mathbb{Z}$ является таким кольцом. Например, для проверки ассоциативности умножения заметим, что для любых a, b и c в $\mathbb{Z}$ выполняются соотношения $s(a) \cdot [s(b) \cdot s(c)] = s(a) \cdot s(b \cdot c) = s[a \cdot (b \cdot c)] = s[(a \cdot b) \cdot c] = s(a \cdot b) \cdot s(c) = [s(a) \cdot s(b)] \cdot s(c)$.

Пусть теперь m — простое число; тогда все ненулевые элементы в $\mathbb{Z}_p$ обратимы, и, следовательно, Z_p является полем. С другой стороны, если m не является простым, то $\mathbb{Z}_m$ — не поле. Чтобы убедиться в этом, запишем $m = a \cdot b$, $a, b < m$; тогда $s(a) \cdot s(b) = s(m) = s(0)$, но $s(a) \neq s(0)$ и $s(b) \neq s(0)$, откуда вытекает, что $s(a)$ и $s(b)$ являются делителями нуля. □

Пример. Рассмотрим $p = 5$. Кольцо $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ является полем, так как все его ненулевые элементы 1, 2, 3 и 4 обратимы (обратные элементы — это 1, 3, 2 и 4 соответственно). Кольцо $\mathbb{Z}_8 = \{0, 1, 2, 3, 4, 5, 6, 7\}$ полем не является, поскольку в нем есть делители нуля, например $2 \cdot 4 = 0 \pmod 8$.

Отметим попутно, что мультипликативная группа кольца $\mathbb{Z}_m$ имеет $\phi(m)$ элементов, где ϕ — функция Эйлера; другими словами, это группа порядка $\phi(m)$.

Теорема 2.3.12. Характеристика λ конечного поля — простое число.

Доказательство. Предположим, что характеристика λ конечного поля — составное число, т.е. $\lambda = ab$, $1 < a, b < \lambda$. Тогда числа $a \cdot 1$ и $b \cdot 1$ отличны от нуля, но $(a \cdot 1)(b \cdot 1) = (ab) \cdot 1 = \lambda \cdot 1 = 0$, что противоречит отсутствию делителей нуля в поле. □

Применения линейных сравнений по модулю m. Одно из многих применений — древний алгоритм проверки ошибок, который изучается в средней школе и известен под названием «отбрасывание девяток». Когда мы складываем десятичные числа в столбик, то если в некотором столбце сумма цифр превышает 9, мы приводим ее по модулю 10 и прибавляем 1 или 2, или 3 и т.д. к следующему столбцу. Если рассматривать сумму десятичных цифр всех чисел, то мы прибавляем к ней 1 или 2, или 3 и т.д. и вычитаем из нее 10 или 20, или 30 и т.д. Поэтому сумма цифр не изменяется по модулю 9. [Можно также заметить, что $abc = 100a + 10b + c = 99a + 9b + (a + b + c)$.]

Пример. Рассмотрим сложение двух чисел:

$$\begin{array}{rrll} & 89 & \text{сумма цифр} = 17, & \text{сумма суммы цифр} = 8 \\ + & \underline{89} & \text{сумма цифр} = \underline{17}, & \text{сумма суммы цифр} = \underline{8} \\ & 178 & 34 & 16 \end{array}$$

Мы видим, что сумма цифр числа 178 равняется $16 \equiv 7 \pmod 9$, сумма цифр числа 34 и сумма цифр числа 16 также равны 7.

Аналогичный алгоритм проверки ошибок применим и к умножению:

$$\begin{array}{rrl} & 89 & \text{сумма цифр} = 17 \\ \times & \underline{87} & \text{сумма цифр} = 15 \\ & 7743 & \end{array}$$

Произведение сумм цифр равно $17 \cdot 15 = 255$, сумма цифр этого произведения равна $12 \equiv 3 \pmod 9$; точно так же сумма цифр произведения 7743 равна $21 \equiv 3 \pmod 9$. Причиной того, что произведение сумм цифр чисел равно по модулю 9 произведению самих чисел, является сравнение $10^k \equiv 1 \pmod 9$, $k \geq 0$. Единственным недостатком этого древнего алгоритма проверки ошибок является то, что он не всегда находит ошибку; 10% случайных ошибок остаются не обнаруженными.

Вычисление мультипликативных обратных. Мультипликативные обратные элементы по модулю m играют важную роль в этой книге. Рассмотрим два способа их вычисления.

Как уже отмечалось в доказательстве теоремы 2.3.10, для вычисления мультипликативного обратного по модулю m к элементу a, который обозначается через $a^{-1} \pmod m$ или просто a^{-1}, можно воспользоваться расширенным алгоритмом Евклида. Из условия $(a, m) = 1$ получаем

$$ax + my = 1,$$

или, перенеся my в правую часть,

$$ax = 1 - my,$$

откуда вытекает, что

$$ax \equiv 1 \pmod m,$$

и, следовательно, x — мультипликативный обратный к a по модулю m. [Отметим, что в нашем случае мы имеем дело с короткими целыми числами; поэтому, полагая $L(b) = 1$ в выражении для времени работы алгоритма Евклида, получаем, что мультипликативный обратный вычисляется за время ~ 1.]

Пример. В поле $GF(11)$ вычислим $4^{-1} \pmod{11}$ — мультипликативный обратный элемент к 4 по модулю 11. Применяя расширенный алгоритм Евклида к 11 и 4, получим следующую таблицу:

Итерация	q	a_0	a_1	x_0	x_1	y_0	y_1
0	–	11	4	1	0	0	1
1	2	4	3	0	1	1	−2
2	1	3	1	1	−1	−2	3
3	3	1	0	−1	4	3	−11

Из последней строки вытекает, что $1 = 11 \cdot (-1) + 4 \cdot 3$, где числа -1 и 3 расположены в столбцах x_0 и y_0 соответственно; следовательно, $4^{-1} \pmod{11} = 3$. Можно ускорить вычисление мультипликативных обратных, заметив, что в нашем случае не обязательно вычислять последовательность элементов x. (Обратный элемент появляется в столбце y_0, который содержит множители при 4; он мог бы появиться в столбце x_0, если бы мы поменяли местами числа 4 и 11.)

Другой способ вычисления мультипликативного обратного по модулю простого числа m состоит в применении следующей замечательной теоремы (см. историческое замечание 4).

Малая теорема Ферма (1640). Если m — простое число и a — произвольное целое число, не делящееся на m, то $a^{m-1} \equiv 1 \pmod{m}$.

Доказательство. Рассмотрим числа a, $2a$, $3a$, $\ldots$, ma. Никакие два из этих чисел не дают при делении на m одинаковых остатков. [Если бы остатки были одинаковы, то $(i-j)a$ было бы кратно m, потому что при вычитании остатки сокращаются; так как m не делит a, то m делило бы $i-j$, что невозможно, поскольку i и j принадлежат последовательности 1, 2, 3, $\ldots$, m.] Поэтому при делении на m числа a, $2a$, $3a$, $\ldots$, ma дают остатки 0, 1, 2, $\ldots$, $m-1$ в каком-то порядке, где 0 получается при делении ma на m. Опуская 0, имеем $a^{m-1}(m-1)! \equiv (m-1)! \pmod{m}$. Вычитая правую часть равенства из левой, получаем $a^{m-1}(m-1)! - (m-1)! \equiv 0 \pmod{m}$, откуда следует, что $(a^{m-1}-1)(m-1)!$ делится на m. Так как m не делит $(m-1)!$ и является простым числом, то m делит $a^{m-1}-1$. □

Следствие 2.3.13. Если m — простое число, то в кольце $\mathbb{Z}_m$ выполняется равенство $a^{-1} = a^{m-2}$.

Следующее утверждение, принадлежащее Эйлеру, обобщает малую теорему Ферма (m — не обязательно простое число).

Теорема 2.3.14 (Эйлер). Если $(a, m) = 1$, то $a^{\phi(m)} \equiv 1 \pmod{m}$.

Доказательство. Доказательство ведется параллельно доказательству теоремы Ферма. Рассмотрим *простую* систему остатков по модулю m r_1, r_2, $\ldots$, $r_{\phi(m)}$ и домножим каждое r_k на a, где $(a, m) = 1$. (Эта система остатков называется «простой», потому что она состоит из остатков, взаимно простых с m и образующих мультипликативную группу.) Домножение изменяет последовательность остатков, но не меняет их общего произведения, так как остатки образуют мультипликативную группу. Следовательно, $a^{\phi(m)} r_1 r_2 \cdots r_{\phi(m)} \equiv$

$r_1 r_2 \cdots r_{\phi(m)} \pmod m$. Поскольку по определению остатки взаимно просты с m, мы можем применить правило сокращения из теоремы 2.3.7, после чего получим $a^{\phi(m)} \equiv 1 \pmod m$. □

Следствие 2.3.15. В кольце $\mathbb{Z}_m$ из $(a, m) = 1$ вытекает, что $a^{-1} = a^{\phi(m)-1}$.

Из сказанного выше следует, что для вычисления мультипликативного обратного к a по модулю m нужно возвести a в некоторую степень k, где k может равняться либо $m-2$, либо $\phi(m)-1$. Рассмотрим два способа возведения в степень.

Первый, использующий «грубую силу», требует выполнения k умножений. Поскольку мы имеем дело с арифметикой коротких чисел, то каждое умножение выполняется за время ~ 1, и, следовательно, все время выполнения равно $O(m)$. [Мы воспользовались тем, что $\phi(m) \leq m$.]

Второй, так называемый «бинарный метод», является гораздо более эффективным способом возведения числа a в степень k; этот метод был известен в Индии 2000 лет назад. Он работает следующим образом: запишем k в двоичной системе счисления, опустив нули перед первой значащей цифрой,

$$k = \sum_{0 \leq i \leq n-1} k_i 2^i.$$

Заменим каждую цифру «1» на пару букв «SM_a» и каждую цифру «0» на букву «S»; после этого вычеркнем пару букв «SM_a» слева. Получившаяся последовательность букв представляет собой правило для вычисления a^k, если интерпретировать «S» как «*возвести в квадрат и взять остаток по модулю m*», а «M_a» как «*умножить на a и взять остаток по модулю m*».

Пример. В $GF(11)$ для того, чтобы найти $4^{-1} \pmod{11}$, мы должны вычислить 4^9, причем двоичное представление 9 равно 1001. Образуем последовательность SM_4 S S SM_4 и, вычеркнув левые SM_4, получим последовательность SSSM_4, которая означает, что мы должны «возвести в квадрат, возвести в квадрат, возвести в квадрат и умножить на 4», выполняя, конечно, приведение по модулю 11 на каждом шаге; иначе говоря, $4^9 = [(4^2)^2]^2 4$. Проводя вычисления, последовательно получим в $GF(11)$ $4^2 = 5$, $4^4 = 5^2 = 3$; выполняя последний шаг, т.е. возведение в квадрат и умножение на 4, заметим, что $4^9 = (4^4)^2 \cdot 4 = (3^2)4 = 9 \cdot 4 = 3$, откуда получаем, что мультипликативный обратный к 4 по модулю 11 равен 3.

Описанная выше «бинарная» процедура работает слева направо по отношению к битовому представлению числа k и, поскольку она не использует временной памяти, хорошо подходит для аппаратной реализации. Однако удобнее работать справа налево, поскольку мы можем получать биты с помощью обычного сдвига вправо на единицу. Следующий алгоритм использует приведенные выше идеи и работает справа налево.

E. Возвести в степень (**E**xponentiate)

Вход: Ненулевые a, k и m; a — элемент из $\mathbb{Z}_m$, $k = \sum_{0\le i\le n-1} k_i 2^i$, k совпадает либо с $m-2$, либо с $\phi(m)-1$.

Выход: a^{-1}, мультипликативный обратный к a элемент по модулю m, где $a^{-1} = a^k$ в кольце $\mathbb{Z}_m$.

1. [Инициализация] $K := k$; $B := 1$; $A := a$.

2. [Вычисление следующего бита] $q := [K/2]$; $r := K - 2\cdot q$; $K := q$; если $r = 0$, то перейти к шагу 5.

3. [Умножить и взять остаток по модулю m] $B := A \cdot B \pmod m$.

4. [Закончить?] Если $K = 0$, то вернуть $a^{-1} \pmod m := B$.

5. [Возвести в квадрат и взять остаток по модулю m] $A := A^2 \pmod m$; перейти к шагу 2.

Анализ времени работы алгоритма ***E***. Пусть двоичное представление числа k состоит из n битов, и пусть j из них равны единице. Тогда в алгоритме выполняется $n + j$ умножений, каждое из которых занимает время ~ 1. Поскольку j не больше, чем n, имеем $O(2\cdot n) = O(2\cdot \log_2 m)$ (проверьте это) $= O(\log_2 m)$ и

$$t_{\mathbf{E}} = O[L(m)],$$

что является значительным улучшением по сравнению с примитивным подходом.

Предположим теперь, что мы применяем алгоритм **E** к целым числам, а не к элементам кольца $\mathbb{Z}_m$. Пусть на вход алгоритма подаются целые числа a, b. Тогда, если мы заменим шаг 3 на «$B := A + B$» и шаг 5 на «$A := A + A$» и если на шаге 1 вместо «$B := 1$» мы присвоим начальное значение «$B := 0$» и вместо «$K := k$» — значение «$K := b$», то алгоритм выдаст $B = a\cdot b$. Этот удобный на практике метод выполнения умножения называют часто «русским крестьянским» методом, подразумевая, что русские крестьяне использовали его, потому что будто бы умели только умножать и делить на 2 и складывать.

Пример. Чтобы умножить 38 на 19, запишем эти числа в вершинах двух столбцов с метками a и b. Следующие элементы этих столбцов получаются умножением на 2 в столбце a и делением на 2 в столбце b. Если число в столбце b нечетно, вычтем из него 1 перед делением на 2 и запишем соседнее число из столбца a в третий столбец, который называется *сумма*. Когда мы получим число 1 в столбце b, сложим числа в столбце *сумма* и получим ответ. Получается следующая таблица:

a	b	*сумма*
38	19	38
76	9	76
152	4	
304	2	
608	1	608
		$\overline{722 = 38 \cdot 19}$

Русский крестьянский метод основан на том, что $a \cdot b = (2a) \cdot (b/2)$, если b четно, и $a \cdot b = (2a) \cdot (b-1)/2 + a$, если b нечетно.

Еще раз о теореме Ферма. Малая теорема Ферма играет важную роль в цифровом кодировании, и мы рассмотрим ее более подробно. Если для некоторого числа a число $a^m - a$ (полученное из $a^{m-1} \equiv 1 \pmod m$) дает ненулевой остаток при делении на m, то m — составное число. Предположим, напротив, что $a^m - a$ делится на m. Следует ли из этого, что m — простое число? Отдельные примеры наводят на мысль, что да: $2^2 - 2$ делится на 2, $2^3 - 2$ делится на 3, $2^5 - 2$ делится на 5, $2^7 - 2$ делится на 7, и числа 2, 3, 5, 7 простые.

У древних китайцев был следующий тест простоты: число m является простым *тогда и только тогда, когда* m делит $2^m - 2$, или $2^{m-1} \equiv 1 \pmod m$. Эта теорема не обсуждалась в течение многих столетий, и Лейбниц верил, что она верна. На самом деле верна лишь вторая часть — «только тогда», а первая часть неверна, как продемонстрировал в 1819 г. французский математик Пьер Фредерик Сарю. Он показал, что $2^{341} - 2$ делится на 341, в то время как 341 — составное число, произведение 11 на 13. Это нетрудно проверить, если воспользоваться сравнением

$$2^{10} \equiv 1 \pmod{341},$$

откуда вытекает, что

$$2^{340} \equiv (2^{10})^{34} \equiv 1^{34} \equiv 1 \pmod{341}, \quad \text{или} \quad 341 \mid (2^{341} - 2).$$

После открытия Сарю было найдено много других контрпримеров с различными значениями основания a; например, $3^{91}-3$ делится на составное число 91, и так далее. Конечно, не всегда проверка будет такой простой, как в случае вычисления $2^{340} \pmod{341}$, когда мы воспользовались тем, что $2^{10} \equiv 1 \pmod{341}$. Поэтому в общем случае придется возводить число в степень, но мы уже знаем эффективный метод для этого.

Составные числа, которые ведут себя так, как простые в теореме Ферма для данного основания a, называются *псевдопростыми по основанию* a. Число 341 является наименьшим псевдопростым по основанию 2, а число 91 — наименьшим псевдопростым по основанию 3. Оказывается, что для всякого основания a существует бесконечно много псевдопростых чисел (см. теорему 2.3.28). Существуют даже составные числа, такие, как $561 = 3 \cdot 11 \cdot 17$ и $1729 = 7 \cdot 13 \cdot 19$, которые являются псевдопростыми по *любому* основанию a. Числа такого вида, называемые *кармайкловыми*, будут рассмотрены ниже (см. историческое замечание 5).

По контрасту со сказанным выше следующая теорема справедлива тогда и только тогда, когда m — простое число.

Теорема 2.3.16 (Вильсон). $(m-1)! \equiv -1 \pmod{m}$ тогда и только тогда, когда m — простое число.

Доказательство. Для доказательства, не использующего теорию групп, в случае простого m рассмотрим произведение $2 \cdot 3 \cdot 4 \cdots (m-3) \cdot (m-2)$. Здесь все сомножители имеют различные мультипликативные обратные по модулю m (см. также упр. 7 к этому разделу); например, для $m = 7$ рассмотрим $2 \cdot 3 \cdot 4 \cdot 5$, где числа 2 и 4, а также 3 и 5 образуют пары взаимно обратных по модулю 7. Произведение элементов такой пары $\equiv 1 \pmod{m}$, и то же самое выполняется для произведения всех $(m-3)/2$ таких пар. Другими словами,

$$2 \cdot 3 \cdot 4 \cdots (m-3) \cdot (m-2) \equiv 1 \pmod{m};$$

домножая его на $m-1$, получим

$$2 \cdot 3 \cdot 4 \cdots (m-3) \cdot (m-2) \cdot (m-1) = (m-1)! \equiv$$
$$\equiv m-1 \equiv -1 \pmod{m}. \qquad \square$$

Рассмотрим функцию $f(m) = \sin\{\varpi \cdot [(m-1)!+1]/m\}$. Из теоремы Вильсона получаем, что $f(m)$ принимает нулевое значение тогда и только тогда, когда m — простое число. К сожалению, теорема

Вильсона в качестве теста простоты не имеет никакой практической ценности, потому что приходится вычислять $(m-1)!$ — очень большое число[1)] даже для небольших значений m.

Некоторые результаты из теории групп. Рассмотрим теперь некоторые интересные результаты теории групп, которые нам понадобятся при изложении греко-китайской теоремы об остатках. Как уже отмечалось в теореме 2.3.11, кольцо $\mathbb{Z}_m$ не всегда является полем, потому что в нем могут быть необратимые элементы. Мы уже рассматривали кольцо $\mathbb{Z}_8$, в котором элементы 2, 4 и 6 не имеют мультипликативных обратных, тогда как элементы 1, 3, 5 и 7 имеют обратные элементы. Различие проистекает из того, что 1, 3, 5 и 7 взаимно просты с 8, а для 2, 4 и 6 это не так.

Обратимые элементы кольца $\mathbb{Z}_m$ образуют мультипликативную группу, которая называется *группой обратимых элементов*[2)] *кольца* $\mathbb{Z}_m$. Эта группа обозначается через $U_m = \{\, a : (a, m) = 1 \,\}$ и имеет $\phi(m)$ элементов, т.е. ее порядок равен $\phi(m)$. Заметим, что U_8 содержит четыре элемента $\{\, 1,\ 3,\ 5,\ 7 \,\}$, каждый из которых имеет мультипликативный обратный; более того, отметим, что в ее таблице умножения, показанной на рис. 2.3.1, каждая строка содержит перестановку элементов группы.

*	1	3	5	7
1	1	3	5	7
3	3	1	7	5
5	5	7	1	3
7	7	5	3	1

Рис. 2.3.1.
Таблица умножения группы U_8.

Пусть G — *абелева* группа из n элементов, т.е. любые два элемента из G коммутируют между собой. Для произвольного a из G обозначим $a \cdot a \cdots a$ (k раз) через a^k; $a^0 = e$ — единичный элемент группы G. Обычные соотношения для степеней по-прежнему выполняются. Справедливо следующее обобщение теоремы Ферма.

[1)] Если вычисления проводить в кольце $\mathbb{Z}_m$, то потребуется $m-2$ умножения, т.е. количество умножений экспоненциально зависит от длины записи числа. — *Прим. перев.*

[2)] *«Group of units»*— иногда ее называют *группой единиц кольца*. — *Прим. перев.*

Теорема 2.3.17. Если G — абелева группа, состоящая из n элементов, то для всякого a из G выполняется равенство $a^n = e$.

Доказательство. Дадим только набросок доказательства. Пусть a_1, a_2, ..., a_n — элементы группы G. Тогда элементы $a \cdot a_1$, $a \cdot a_2$, ..., $a \cdot a_n$ попарно различны, и множество $\{a \cdot a_1,\ a \cdot a_2, \ldots,\ a \cdot a_n\}$ совпадает с множеством $\{a_1,\ a_2, \ldots,\ a_n\}$. Завершение доказательства проводится так же, как и в теореме Ферма, нужно только показать (по индукции), что произведение более чем двух элементов группы не зависит от расстановки скобок (обобщенная ассоциативность) и от порядка сомножителей (обобщенная коммутативность). □

Эта абстрактная версия теоремы Ферма справедлива и для неабелевых групп; ее доказательство можно найти где угодно (см. например, (Cnilds, 1979)).

Пусть G — группа из n элементов, $a \in G$ и $S = \{\, k \geq 1 : a^k = e \,\}$. Так как $a^n = e$, то S непусто и по принципу полной упорядоченности S имеет *наименьший* элемент k_0, который называется *порядком* элемента a. Группа называется *циклической*, если в ней существует элемент a, степени 1, a, a^2, ... которого пробегают все элементы группы. Этот элемент называется *образующим* или, в случае группы U_m, *примитивным корнем по модулю* m.

Примитивные корни могут быть использованы для генерации случайных чисел на компьютере. Выберем примитивный корень по модулю m, где длина записи m равна длине машинного слова, и всякий раз, когда пользователю потребуется случайное число, будем выдавать следующую степень этого примитивного корня по модулю m. Выбор именно примитивного корня обеспечивает нам максимально возможную длину цикла в порождаемой таким способом последовательности «случайных» чисел. Справедлива следующая теорема.

Теорема 2.3.18. Пусть G — группа, состоящая из n элементов, и пусть k_0 — порядок элемента a из G. Тогда $k_0 \mid n$.

Доказательство. Нетрудно видеть, что элементы e, a, a^2, ..., a^{k_0-1} различны. Если эти k_0 элементов не исчерпывают всей группы G, то в ней должен быть еще какой-то элемент, скажем a_2. Тогда нетрудно видеть, что a_2, $a_2 \cdot a$, $a_2 \cdot a^2$, ..., $a_2 \cdot a^{k_0-1}$ — это k_0 различных элементов, ни один из которых не совпадает с предыдущими k_0 элементами группы G. Если G не исчерпана, то должен быть еще один элемент a_3, и так далее. Этот процесс получения новых элементов a_i должен рано или поздно оборваться, потому что G состоит из n

элементов. Ясно, что порядок n группы G равен $k_0 \cdot j$, где a_j — последний полученный элемент. □

Теорема 2.3.19. Если k_0 — порядок элемента a из G и $a^k = e$, то $k_0 \mid k$.

Доказательство. Пусть $k = k_0 q + r$, где $0 \leq r < k_0$. Если $r = 0$, то доказывать нечего, поэтому предположим, что $r > 0$. Тогда $a^k = a^{k_0 q + r} = (a^{k_0})^q a^r = e$, откуда следует, что $a^r = e$. Это, однако, противоречит тому, что k_0 — наименьшее число со свойством $a^{k_0} = e$. □

Полагая $G = U_m$, получаем из доказанного выше, что порядок любого элемента группы U_m делит $\phi(m)$. Примитивные корни, если они существуют, являются в точности элементами максимального возможного порядка $\phi(m)$. Очевидно также, что теорема 2.3.14 (Эйлера) — следствие этих замечаний.

Проверим, является ли группа U_8 циклической, т.е. существует ли элемент a в U_8, степени $1, a, a^2, a^3, \ldots$ которого пробегают все элементы этой группы. Рассмотрим порядки различных элементов из U_8 по модулю 8:

n	$=$	1	2	3	4	5	6	7	...
3^n	$\equiv$	3	1	3	1	3	1	3	...
5^n	$\equiv$	5	1	5	1	5	1	5	...
7^n	$\equiv$	7	1	7	1	7	1	7	...

Период этих последовательностей равен 2, поэтому порядок любого отличного от единицы элемента U_8 также равен 2, порядок делит $\phi(8) = 4$ (теорема 2.3.18). Поэтому группа U_8 не циклическая, и не существует примитивных корней по модулю 8. Следующая теорема объясняет, почему это так.

Теорема 2.3.20. Группа U_m является циклической тогда и только тогда, когда m равно 1, 2, 4, p^a или $2p^a$, где p — нечетное простое число и $a > 0$. Значит, примитивные корни по модулю m существуют в точности для таких значений m.

Доказательство. Доказательство можно найти в (LeVeque, 1977). □

Оказывается, что 8 — наименьшее число, не имеющее примитивных корней. С другой стороны, по теореме 2.3.20 U_{18} — циклическая группа, потому что $18 = 2 \cdot 3^2$. Группа U_{18} состоит из $\phi(18) = 6$

элементов, а именно $U_{18} = \{1, 5, 7, 11, 13, 17\}$, и элемент 5 является примитивным корнем, поскольку

n	$=$	0	1	2	3	4	5	6	...
5^n	$\equiv$	1	5	7	17	13	11	1	...

Предлагаем читателю найти все примитивные корни по модулю 18. Непосредственно из теоремы 2.3.20 получаем

Следствие 2.3.21. Если m — нечетное простое число, то группа U_m циклическая и уравнение $x^2 = 1$ в U_m не имеет решений, отличных от $x = \pm 1$.

Как только мы нашли примитивный корень a по модулю m в U_m, мы можем сразу же получить другой корень — его мультипликативный обратный a^{-1} по модулю m. В случае U_{18} $5^{-1} \equiv 11 \pmod{18}$, и 11 также является примитивным корнем. (Этот факт будет проверен ниже.)

Сколько примитивных корней содержится, например, в U_{18}? Если возвести примитивный корень r в степень $k > 1$, где $\gcd[k, \phi(m)] = 1$, то $r' = r^k$ будет другим примитивным корнем, потому что порядок элемента r' равен $\phi(m)$ (это будет доказано ниже); в случае $U = 18$ $\phi(18) = 6$ и, поскольку $(5, 6) = 1$, вычислим 5^5 в кольце $\mathbb{Z}_{18}$. Получим 11, другой примитивный корень. В общем случае число примитивных корней равно $\phi[\phi(m)]$. Например, в U_{18} имеются два примитивных корня, 2 и 11.

Если, однако, $\gcd[k, \phi(m)] = d > 1$, то порядок элемента $r' = r^k$ равен $\phi(m)/d$. Чтобы убедиться в этом, заметим сначала, что число $\phi(m)/d$ является периодом элемента r', т.е. $(r')^{\phi(m)/d} = r^{\phi(m)k/d} \equiv 1^{k/d} = 1 \pmod{m}$. Покажем теперь, что это минимальный период. Используя наименьшее общее кратное $[k, \phi(m)]$, получаем $(r')^{\phi(m)k/d} = r^{\phi(m)k/d} = r^{[k,\phi(m)]} = r^{\phi(m)j} \equiv 1^j = 1 \pmod{m}$.

Уравнения по модулю некоторого числа m. В заключение рассмотрим уравнения по модулю m и греко-китайский алгоритм. Заметим, что уравнение $6x \equiv 4 \pmod 8$ имеет решения $x \equiv 2$, $x \equiv 6$, ... $\pmod 8$, тогда как уравнение $6x \equiv 5 \pmod 8$ не имеет решений [попробуйте подставить $x \equiv 0, 1, \ldots, 7 \pmod 8$]. Следующая теорема говорит о том, в каких случаях можно решить такое уравнение.

Теорема 2.3.22. Уравнение $ax \equiv b \pmod m$ имеет решение тогда и только тогда, когда $(a, m) \mid b$. Если решение существует, то оно единственно по модулю m/d, где $d = (a, m)$; по модулю m уравнение имеет d решений.

Доказательство. Целое число удовлетворяет уравнению $ax \equiv b \pmod{m}$ тогда и только тогда, когда существует целое число y, такое, что $ax + my = b$. По теореме 2.2.4 уравнение $ax + my = b$ имеет решения тогда и только тогда, когда $(a, m) \mid b$. Для доказательства второй части предположим, что x удовлетворяет условию $ax \equiv b \pmod{m}$, и пусть z сравнимо с x по модулю m/d, где $d = (a, m)$. Тогда $z = x + w(m/d)$ для некоторого $w \in \mathbb{Z}$, и $az = ax + aw(m/d) = ax + mw(a/d) \equiv ax \equiv b \pmod{m}$; таким образом, $az \equiv b \pmod{m}$. Обратно, пусть $ax \equiv az \equiv b \pmod{m}$. Тогда $ax - az \equiv b - b \equiv 0 \pmod{m}$; следовательно, $m \mid a(x - z)$. По теореме 2.2.6 m/d делит $x - z$, и, значит, $x \equiv z \pmod{m/d}$. □

Пример. Найдем решения уравнения $270x \equiv 36 \pmod{342}$. Применяя расширенный алгоритм Евклида, получим, что $(-5) \cdot 270 + 4 \cdot 342 = 18$, и $18 \mid 36$. По теореме 2.3.22 это уравнение имеет решение, единственное по модулю $19 = 342/18$. Для нахождения этого решения умножим равенство $(-5) \cdot 270 + 4 \cdot 342 = 18$ на $2 = 36/18$ и получим $(-10) \cdot 270 + 8 \cdot 342 = 36$, откуда следует, что (-10) — одно из решений уравнения по модулю 342. По модулю 19 это единственное решение, равное 9, поскольку $9 \equiv -10 \pmod{19}$. Другими решениями по модулю 342 являются числа 9, 28, 47, 66, 85, 104, 123, 142 и так далее.

Частным случаем теоремы 2.3.22 является следующее утверждение.

Следствие 2.3.23. Уравнение $ax \equiv 1 \pmod{m}$ имеет решение тогда и только тогда, когда $(a, m) = 1$. Решение $a^{-1} \pmod{m}$ единственно по модулю m и является мультипликативным обратным к a элементом по модулю m.

Пример. Уравнение $2x \equiv 1 \pmod{26}$ не имеет решений, поскольку $(2, 26) = 2$. В данном случае это можно показать и более простым способом: мы ищем число x, такое, что $2x - 1 = k \cdot 26$, однако левая часть последнего уравнения — всегда нечетное число, а правая — четное.

Рассмотрим теперь задачу решения системы линейных уравнений по модулю некоторого числа m. С этой задачей связана греко-китайские теорема об остатках и алгоритм (см. историческое замечание 6). Для того чтобы представить замечательную греко-китайскую теорему об остатках в общем виде, обсудим сначала утверждение о том, что если целое число m может быть разложено в произведение $m = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ (разложение в произведение степеней простых

чисел), то кольцо $\mathbb{Z}_m$ также может быть «разложено» в прямое произведение колец $\mathbb{Z}_{p_i^{e_i}}$. (Полное доказательство будет приведено в теореме 2.3.25.)

Например, $\mathbb{Z}_6 = \mathbb{Z}_2 \times \mathbb{Z}_3$, поскольку $6 = 2 \cdot 3$. Это означает, что мы должны рассматривать пары (x_1, x_2), где $x_1 \in \mathbb{Z}_2$ и $x_2 \in \mathbb{Z}_3$; например, шести элементам кольца $\mathbb{Z}_6$ соответствуют пары $(0,0)$, $(0,1)$, $(0,2)$, $(1,0)$, $(1,1)$, $(1,2)$. Арифметические операции выполняются покомпонентно: если обозначить значком $\bullet$ одну из операций $+$ или $\cdot$, то $(x_1, x_2) \bullet (y_1, y_2) = (x_1 \bullet y_1, x_2 \bullet y_2)$, где операция $x_1 \bullet y_1$ выполняется в $\mathbb{Z}_2$ (арифметика по модулю 2), а операция $x_2 \bullet y_2$ — в $\mathbb{Z}_3$ (арифметика по модулю 3); например, $(0,2) \cdot (1,2) = (0,1)$, где умножение $2 \cdot 2$ выполняется по модулю 3. Справедлива следующая теорема.

Теорема 2.3.24 (греко-китайская теорема об остатках). Пусть m_1, m_2, $\ldots$, m_k — попарно взаимно простые целые числа > 1, и пусть $M = m_1 m_2 \cdots m_k$. Тогда существует единственное неотрицательное решение по модулю M следующей системы уравнений:

$$\begin{aligned} x &\equiv a_1 \pmod{m_1}, \\ x &\equiv a_2 \pmod{m_2}, \\ &\ldots \\ x &\equiv a_k \pmod{m_k}. \end{aligned}$$

Другими словами, отображение, которое каждому целому числу x, $0 \le x \le M - 1$, ставит в соответствие строку $(a_1, a_2, \ldots, a_k)$, где $x_i \equiv a_i \pmod{m_i}$, $i = 1, 2, \ldots, k$, является биекцией кольца $\mathbb{Z}_M$ на $\mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \mathbb{Z}_{m_k}$.

Доказательство. Дадим конструктивное доказательство этой теоремы. Нужно найти число x, $0 \le x \le M - 1$, удовлетворяющее одновременно всем сравнениям $x \equiv a_i \pmod{m_i}$, $i = 1, 2, \ldots, k$. Будем решать уравнения по 2 одновременно.

Рассмотрим сначала первые два сравнения. Первое сравнение $x \equiv a_1 \pmod{m_1}$ справедливо для всякого x вида $x = a_1 + m_1 q$, q произвольное. Для нахождения q подставим значение x во второе сравнение $x \equiv a_2 \pmod{m_2}$, после чего получим $x = a_1 + m_1 q \equiv a_2 \pmod{m_2}$, откуда $q \equiv (m_1)^{-1}(a_2 - a_1) \pmod{m_2}$. (Конечно, нам придется сначала вычислить обратный к m_1 по модулю m_2; см. упр. 2 по программированию к этому разделу, в котором описана процедура **MODINV**.) Таким образом, $q = m_1^{-1}(a_2 - a_1) + r m_2$ для некоторого r.

Подставив значение q в выражение $x = a_1 + m_1 q$, получим, что решение x первых *двух* уравнений представляется в виде $x = a_{12} + r(m_1 m_2)$ для некоторого r.

Теперь первые два сравнения могут быть заменены на одно, $x \equiv a_{12} \pmod{m_1 m_2}$, которое мы рассматриваем по модулю произведения $m_1 m_2$. Применим описанную выше процедуру к $x \equiv a_1 2 \pmod{m_1 m_2}$ и сравнению, которое первоначально было третьим, и будем повторять этот процесс, пока не найдем число x, удовлетворяющее всем сравнениям.

Для доказательства единственности предположим, что существует x', $0 \leq x' \leq M - 1$, такой, что $x' \equiv a_i \pmod{m_i}$ для любого i. Тогда $x - x' \equiv 0 \pmod{m_i}$ для всех i, откуда следует, что $m_i \mid (x - x')$ для любого i. Но тогда $M \mid (x - x')$ и, поскольку $|x - x'| < M$, $x = x'$. $\square$

Интересно отметить, что если в теореме 2.3.24 модули m_i не являются взаимно простыми, то решение существует тогда и только тогда, когда $(m_i, m_j) \mid (a_i - a_j)$ для всех пар i, j. Если решение существует, то оно единственно по модулю наименьшего общего кратного $[m_1, m_2, \ldots, m_k]$ чисел m_i; см. также упр. 10 к этому разделу.

Пример. Решим систему уравнений

$$x \equiv 1 \pmod{2},$$
$$x \equiv 2 \pmod{5},$$
$$x \equiv 5 \pmod{7}.$$

В соответствии с процедурой, описанной в доказательстве теоремы 2.3.24, мы видим, что первое уравнение выполняется для $x = 1 + 2q$. Чтобы вычислить q, подставим x во второе уравнение; получим $1 + 2q \equiv 2 \pmod{5}$, или $2q \equiv (2 - 1) \pmod{5}$. Затем вычислим мультипликативный обратный элемент к $2 \pmod{5}$, который равен 3, и, таким образом, $q \equiv 3 \pmod{5}$ или $q = 3 + 5r$ для некоторого r. Следовательно, решением первых двух уравнений является $x = 1 + 2(3 + 5r) = 7 + 2 \cdot 5r$, т.е. $x \equiv 7 \pmod{2 \cdot 5}$.

Теперь нам нужно решить систему двух уравнений $x \equiv 7 \pmod{2 \cdot 5}$ и $x \equiv 5 \pmod{7}$. Имеем $x = 7 + 2 \cdot 5q \equiv 5 \pmod{7}$, или $2 \cdot 5q \equiv (5 - 7) = -2 \equiv 5 \pmod{7}$. Мультипликативный обратный элемент к 10 по модулю 7 совпадает с обратным к 3 по модулю 7, который равен 5. Далее получаем $q \equiv 5 \cdot 5 \pmod{7} \equiv 4 \pmod{7}$, или $q = 4 + 7r$ для некоторого r. Следовательно, решением трех уравнений является число $x = 7 + 2 \cdot 5(4 + 7r)$, или $x \equiv 47 \pmod{2 \cdot 5 \cdot 7}$.

Отметим, что $47 = 1 + 3 \cdot (2) + 4 \cdot (2 \cdot 5)$, где коэффициенты 3 и 4 являются значениями q.

Перед тем как описать греко-китайский алгоритм, основанный на теореме 2.3.14, мы приведем две близкие к ней теоремы.

Теорема 2.3.25. Пусть $m = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$. Тогда функция, которая каждому $x \in \mathbb{Z}_m$ ставит в соответствие строку $(x_1, x_2, \ldots, x_k)$, где $x \equiv x_i \pmod{p_i^{e_i}}$, $i = 1, 2, \ldots, k$, является кольцевым изоморфизмом (т.е. взаимно однозначным гомоморфизмом на) кольца $\mathbb{Z}_m$ и кольца строк $(x_1, x_2, \ldots, x_k)$, где $x_i \in \mathbb{Z}_{p_i^{e_i}}$ для $i = 1, 2, \ldots, k$. Более того, если обозначить через $\bullet$ любую из операций $+$ или $\cdot$, имеем $(x_1, x_2, \ldots, x_k) \bullet (y_1, y_2, \ldots, y_k) = (x_1 \bullet y_1, x_2 \bullet y_2, \ldots, x_k \bullet y_k)$, где знак «$\bullet$» в правой части равенства обозначает соответствующую операцию в $\mathbb{Z}_{p_i^{e_i}}$, $\quad i = 1, 2, \ldots, k$.

Доказательство. Доказательство вытекает непосредственно из теоремы 2.3.24 и оставляется читателю в качестве упражнения. □

Разложение, указанное в теореме 2.3.25, записывается как $\mathbb{Z}_m \cong \times_{1 \le i \le k} \mathbb{Z}_{p_i^{e_i}}$. Это разложение колец индуцирует разложение групп их обратимых элементов $U_m \cong \times_{1 \le i \le k} U_{p_i^{e_i}}$.

Одним из применений теоремы 2.3.25 (и греко-китайской теоремы об остатках в общем виде), которое будет подробно рассмотрено в разд. 2.4, является *греко-китайское представление* числа x. А именно произвольное целое положительное число x, $0 < x < M$, где $M = m_1 m_2 \cdots m_k$ и $(m_i, m_j) = 1$ для $i \neq j$, однозначно представимо своими наименьшими неотрицательными остатками по модулю m_i, причем сложение и умножение выполняется покомпонентно.

Пример. Для $m_1 = 3$ и $m_2 = 5$ имеем $6 = (0, 1)$ и $7 = (1, 2)$; сумма этих чисел равна $[0 + 1 \pmod 3, 1 + 2 \pmod 5] = (1, 3)$, последняя пара представляет число 13. Что можно сказать об их произведении?

Заметим, что по паре $(1, 3)$ можно найти соответствующее целое число с помощью либо греко-китайского алгоритма, который будет описан ниже, либо специальных таблиц.

Следующая теорема является обобщением теоремы Ферма. Будем называть натуральное число *свободным от квадратов*, если оно является произведением различных простых чисел, т.е. не делится ни на какой квадрат > 1; например, числа 1, 2, 3, 5, 6, ... свободны от квадратов.

Теорема 2.3.26. Натуральное число m свободно от квадратов тогда и только тогда, когда существует $q > 1$ (которое будет определено ниже), такое, что для любого целого числа b выполняется сравнение $b^q \equiv b \pmod{m}$. А именно, если $m = p_1 p_2 \cdots p_k$ для различных простых чисел p_i и $q = \lambda[p_1 - 1, p_2 - 1, \ldots, p_k - 1] + 1$ для произвольного натурального λ, то $b^q \equiv b \pmod{m}$ для любых b.

Доказательство. Предположим, что m свободно от квадратов, т.е. $m = p_1 p_2 \cdots p_k$ для различных простых чисел. Тогда из утверждения о единственности в греко-китайской теореме об остатках мы получаем, что для произвольного $b \in \mathbb{Z}$ и любого $q > 1$ сравнение $b^q \equiv b \pmod{m}$ выполняется тогда и только тогда, когда $b^q \equiv b \pmod{p_i}$ для всех $i = 1, 2, \ldots, k$. Пусть $q = q_m + 1$, где $q_m = [p_1 - 1, p_2 - 1, \ldots, p_k - 1]$ либо любое другое общее кратное чисел $p_1 - 1, p_2 - 1, \ldots, p_k - 1$. Это означает, что $q_m = (p_i - 1)q_i$ для любого i и некоторого q_i; поэтому $b^q = b^{q_m+1} = b^{(p_i-1)q_i+1} = b \cdot b^{(p_i-1)q_i}$. Если $(b, p_i) = 1$, то по теореме Ферма $b^{p_i-1} \equiv 1 \pmod{p_i}$, в противном случае $b \equiv 0 \pmod{p_i}$. В любом случае мы получаем $b^q \equiv b \pmod{p_i}$ для всех $i = 1, 2, \ldots, k$, и, следовательно, $b^q \equiv b \pmod{m}$.

Предположим теперь, что m не свободно от квадратов, т.е. $m = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$, где $e_j > 1$ для некоторого j, $1 < j < k$. Найдем с помощью греко-китайской теоремы об остатках решение $x = b$ системы сравнений $\{x \equiv 0 \pmod{p_i}, i = 1, 2, \ldots, k, i \neq j, x \equiv p_j^{e_j - 1} \pmod{p_j}\}$. Получим, что m не делит b, поскольку $p_j^{e_j}$ не делит b, но m делит b^q для любого $q > 1$. Таким образом, сравнение $b^q \equiv b \pmod{m}$ не выполняется ни для какого $q > 1$. □

В заключение мы представим греко-китайский алгоритм решения системы уравнений над кольцами вычетов для случая, когда модули попарно взаимно просты (в этом случае решение существует по теореме 2.3.24). Более общий алгоритм в случае, когда модули не взаимно просты, также не очень сложен, но в приложениях модули всегда будут различными нечетными простыми числами.

Рассмотрим сначала систему из двух уравнений:

$$\begin{aligned} x &\equiv a \pmod{m_1}, \\ x &\equiv b \pmod{m_2}, \quad (m_1, m_2) = 1. \end{aligned}$$

Эта система решается методом, описанным в доказательстве теоремы 2.3.24. Мы также будем использовать процедуру **MODINV**, которая вычисляет (наименьший неотрицательный) обратный элемент по модулю m_2 к элементу m_1 (см. упр. 2 по программированию к этому разделу). Оформив описанный метод в виде алгоритма, получим

GCRA2. Греко-китайский алгоритм для двух сравнений (**G**reek-**C**hinese **R**emainder **A**lgorithm-**2** congruences)

Вход: a, m_1, b, m_2, такие, что $x \equiv a \pmod{m_1}$ и $x \equiv b \pmod{m_2}$, где m_1, m_2 — целые числа одинарной точности, такие, что $(m_1, m_2) = 1$ и m_1, m_2 оба > 1.

Выход: x — единственное наименьшее неотрицательное решение по модулю $m_1 m_2$ системы сравнений.

1. [Если $a > 0$, то ничто не меняется] $x := \mathbf{MOD}(a, m_1)$.

2. [Вычисление m^{-1}] $m^{-1} := \mathbf{MODINV}(m_1, m_2)$.

3. [Вычисление q] $q := \mathbf{MOD}[m_1^{-1} \cdot (b - x), m_2]$.

4. [Выход] Вернуть $x := x + m_1 q$ (поскольку $0 \le q < m_2$, то возвращаемое значение x удовлетворяет неравенствам $0 \le x < m_1 m_2$).

Анализ времени работы алгоритма ***GCRA2***. Заметим, что шаги 1 и 2 выполняются за время ~ 1, поскольку мы работаем с короткими целыми числами (используя расширенный алгоритм Евклида для вычисления мультипликативного обратного). На шаге 3 мы выполняем умножение и деление, на шаге 4 — только одно умножение. Время выполнения каждой из этих операций доминируется временем вычисления произведения $m_1 m_2$ (проверьте это), и, следовательно,

$$t_{\mathbf{GCRA2}}(a, m_1, b, m_2) = O[L(m_1, m_2)].$$

Рассмотрим теперь общий случай, когда нам нужно решить следующую систему уравнений по попарно взаимно простым модулям:

$$\begin{aligned} &x \equiv a_1 \pmod{m_1}, \\ &x \equiv a_2 \pmod{m_2}, \\ &\dots \\ &x \equiv a_k \pmod{m_k}, \quad (m_1, m_2) = 1 \quad \text{для} \quad i \ne j. \end{aligned}$$

Идея состоит в том, чтобы использовать **GCRA2** для последовательного решения пар уравнений. На первом шаге мы получаем решение x_0 первых двух уравнений, где x_0 — наименьшее неотрицательное решение по модулю $m_1 m_2$. На следующем шаге мы получаем решение по модулю $m_1 m_2 m_3$ пары уравнений

$$x \equiv x_0 \pmod{m_1 m_2}, \quad x \equiv a_3 \pmod{m_3},$$

и так далее. Получаем следующий алгоритм.

GCRAk. Греко-китайский алгоритм для k сравнений (**G**reek-**C**hinese **R**emainder **A**lgorithm-**k** conqruences)

Вход: Пары a_i, m_i, такие, что $x \equiv a_i \pmod{m_i}$, $i = 1, 2, \ldots, k$; каждое m_i является коротким целым числом, $m_i > 1$ и $(m_i, m_j) = 1$ для $i \neq j$.

Выход: x — единственное наименьшее неотрицательное решение по модулю $m_1 m_2 \cdots m_k$ системы из k уравнений.

1. [Инициализация] $m := 1$; $x := \mathbf{MOD}(a_1, m_1)$.

2. [Применение в цикле **GCRA2**] Для i от 1 до $k-1$ выполнять $\{m := m \cdot m_i$; $m^{-1} := \mathbf{MODINV}(m, m_{i+1})$; $q := \mathbf{MOD}[m^{-1}(a_{i+1} - x), m_{i+1}]$; $x := x + mq\}$.

3. [Вычисление q] $q := \mathbf{MOD}[m_1^{-1} \cdot (b - x), m_2]$.

4. [Выход] Вернуть x.

Анализ времени работы алгоритма ***GCRAk***. Ясно, что время работы алгоритма **GCRAk** доминируется временем выполнения второго шага, на котором в цикле выполняется **GCRA2**. Если $M = m_1 \cdots m_k$, то i-е выполнение тела цикла требует времени $\sim L(m_1 \cdot m_2 \cdots m_i) L(m_{i+1})$. Поэтому в целом цикл, состоящий из $(k-1)$ шага, выполняется за время $\leq L(M)\{\sum_{2 \leq i \leq k} L(m_i)\} \sim L(M) \cdot L(\prod_{2 \leq i \leq k} m_i)$ (напомним, что функция L ведет себя как логарифм) $\leq L^2(M)$, и мы имеем

$$t_{\mathbf{GCRAk}}(a_i, m_i, i = 1, 2, \ldots, k) = O[L^2(m_1 \cdot m_2 \cdots m_k)].$$

Пример, иллюстрирующий работу этого алгоритма, был приведен непосредственно после теоремы 2.3.24. Читатель может попробовать применить этот алгоритм для системы $x \equiv -2 \pmod{15}$, $x \equiv 6 \pmod{8}$ и $x \equiv 11 \pmod{7}$, единственным решением которой по модулю 840 является $x = 718$; алгоритм формирует ответ в виде $718 = 13 + 7 \cdot 15 + 5 \cdot (15 \cdot 8)$. В общем случае, если мы имеем k уравнений, то ответ представляется в виде

$$x = q_1 + q_2 \cdot (m_1) + q_3 \cdot (m_1 m_2) + \cdots q_k \cdot (m_1 \cdot m_2 \cdots m_{k-1}),$$

где каждое q_i неотрицательно и меньше соответствующего модуля m_i, а $q_1 = \mathbf{MOD}(a_1, m_1)$.

Мы закончим этот раздел описанием применения греко-китайского алгоритма к задаче о безопасном хранении ключа (Asmuth, C., Bloom, J.: A modular approach to key safeguarding. Mathematics Department, Texas A and M University, College Station, TX, 77844).

Пусть K — ключ, который нужно сохранить. При этом требуется, чтобы любые L человек из тех k $(k > L)$, которые получили «информацию» о ключе, могли бы вместе восстановить ключ, но никакая группа из $L-1$ человек или менее не могла этого сделать. Информация, передаваемая различным людям, будет описана ниже.

Для решения этой задачи выберем множество целых чисел $\{p, d_1, d_2, \ldots, d_k\}$, такое, что

a. $p > K$.
b. $d_1 < d_2 < \cdots < d_k$.
c. $\gcd(p, d_i) = 1$ для всех $i = 1, 2, \ldots, k$.
d. $\gcd(d_i, d_j) = 1$ для $i \neq j$.
e. $d_1 \cdot d_2 \cdots d_L > p \cdot d_{k-L+2} \cdot p \cdot d_{k-L+3} \cdots d_k$.

Пункт **e** означает, что произведение L наименьших чисел d_i больше, чем произведение p и $L-1$ наибольших чисел d_i. Пусть $D = d_1 \cdot d_2 \cdots d_L$; тогда D/p больше, чем произведение любых $L-1$ чисел d_i. Выберем теперь случайное число r в интервале $[0, (D/p) - 1]$ и вычислим $K' = K + r \cdot p$, чтобы K попало в интервал $[0, D-1]$. Между разными людьми распределяются числа

$$K_i \equiv K' \pmod{d_i}, \quad i = 1, 2, \ldots, k.$$

Пример. $K = 5$, $L = 2$, $k = 3$, $p = 7$, $d_1 = 11$, $d_2 = 13$, $d_3 = 17$. Далее, $D = d_1 \cdot d_2 = 11 \cdot 13 = 143 > 119 = 7 \cdot 17 = p \cdot d_3$, как и требуется. Выберем случайное число r в интервале $[0, (143/7) - 1] = [0, 19]$, например 2. Имеем

$$K' = K + r \cdot p = 5 + 2 \cdot 7 = 19.$$

Распределяемые числа равны

$$K_1 = 19 \pmod{11} = 8,$$
$$K_2 = 19 \pmod{13} = 6,$$
$$K_3 = 19 \pmod{17} = 2.$$

По любым двум из этих чисел можно восстановить K. Например, для K_1 и K_2 имеем

$$K' \equiv 8 \pmod{11}, \quad K' \equiv 2 \pmod{17}.$$

Применив греко-китайский алгоритм, получим $K' = 19$, откуда $K = K' - r \cdot p = 5$.

2.3.3. Тесты простоты

В этом разделе рассматривается следующая задача: *для заданного положительного целого числа m определить, является ли оно простым*. При этом мы не интересуемся делителями числа m. Наиболее очевиден следующий способ решения.

Первый детерминистический тест. Разделим число m последовательно на числа 2, 3, ..., $\lfloor\sqrt{m}\rfloor$. Если при каком-нибудь делении мы получим нулевой остаток, то число m составное, а делитель и частное являются его сомножителями; в противном случае m простое. (Почему нужно проверять только до $\lfloor\sqrt{m}\rfloor$?)

Каково время работы этого теста? Очевидно, необходимо выполнить $\sqrt{m}$ делений, поэтому время проверки простоты числа m равно $O(\sqrt{m})$. Эта оценка, однако, не является полиномиальной, поскольку, если учитывать длину записи m, она принимает вид $O(2^{L(m)/2})$; таким образом, это экспоненциальный тест, т.е. очень медленный. Простая на первый взгляд задача оказывается достаточно сложной. Конечно, первый тест (известный также под названием «метод пробных делений») делает гораздо больше, чем требуется: он не только определяет, является ли число простым, но и находит сомножители составного числа.

Значительный прогресс был достигнут в последние годы. Существуют две группы алгоритмов проверки простоты: детерминистические и *вероятностные*. Литература по этому предмету весьма обширна (Adleman et al., 1983; Cohen et al., 1982; Dixon, 1984; Lucas, 1961; Pomerance, 1981; Pratt, 1975; Rabin, 1980; Solovay et al., 1977; Williams, 1978), и пока не сделано попыток дать полный обзор всех алгоритмов. Вместо этого мы приведем несколько тестов простоты, представляющих обе эти группы; практически все тесты так или иначе связаны с малой теоремой Ферма.

Продолжим рассмотрение детерминистических тестов.

Второй детерминистический тест. Число m просто тогда и только тогда, когда $m \mid \{(m-1)!+1\}$.

Тест основан на теореме 2.3.16 (Wilson). Как уже отмечалось, факториал $(m-1)!$ уничтожает всякий интерес к этому тесту; крайне медленный метод решета оказывается очень быстрым по сравнению с проверкой делимости $(m-1)!+1$ для больших m. Если m имеет 100 цифр, то $(m-1)!$ состоит примерно из 100^{102} *цифр* — не путайте с числом 100^{102}.

Для третьего теста, который будет приведен ниже, нам понадобится следующее утверждение: если m — простое число, то существует натуральное число $b < m$, порядок которого по модулю m равен $m-1$, т.е. $b^{m-1} \equiv 1 \pmod{m}$ и никакая меньшая степень числа b не равна $1 \pmod{m}$. Следующая теорема является обратным утверждением.

Теорема 2.3.27. Пусть m — целое число ≥ 2. Если существует $b < m$, такое, что порядок b по модулю m равен в точности $m-1$, то m — простое число.

Доказательство. Если m не просто, то $\phi(m) < m-1$. Для числа b из условия теоремы либо $(b, m) = 1$, и в этом случае его порядок должен делить $\phi(m) < m-1$, либо $(b, m) > 1$, и тогда никакая степень b не сравнима с $1 \pmod{m}$. □

Таким образом, получаем следующий тест простоты.

Третий детерминистический тест. Число m просто тогда и только тогда, когда существует элемент b, порядок которого по модулю m в точности равен $m-1$. Эквивалентная формулировка: m просто тогда и только тогда, когда существует b, $(b, m) = 1$, такое, что $b^{m-1} \equiv 1 \pmod{m}$ и $b^{(m-1)/p}$ не равно $1 \pmod{m}$ для *каждого* простого делителя p числа $m-1$.

Этот тест известен под названием «тест Лукаса» (1961). Для того чтобы определить, является ли m простым, нужно действовать следующим образом. Если b^{m-1} не равно $1 \pmod{m}$ для некоторого $b < m$, то m не является простым (*негативный* тест, вытекающий из теоремы Ферма). С другой стороны, если для некоторого b порядок b равен $m-1$, то m просто (*позитивный* тест, вытекающий из теоремы 2.3.27).

К сожалению, очень сложно проверить, равен ли порядок b по модулю m числу $m-1$, потому что нужно для *каждого* простого делителя p числа $m-1$ показать, что $b^{(m-1)/p}$ не равно $1 \pmod{m}$.

Приведем два небольших примера. Пусть сначала $m = 899$. После проверки того, что 899 не делится на 2, 3, 5, 7 и 11, можно предположить, что 899 — простое число. Для того чтобы наверняка убедиться в этом, применим тест; взяв $b = 2$, получим, что $2^{898} \equiv 683 \pmod{899}$. Поэтому 899 не является простым числом, и, в самом деле, оно равно $29 \cdot 31$. Мы воспользовались только негативным тестом. Рассмотрим теперь число 341, для которого мы знаем, что $2^{340} \equiv 1 \pmod{341}$. Для применения позитивного теста мы должны вычислить порядок

числа 2 по модулю 341, а это требует разложения на множители числа 340. Можно убедиться, что $340 = 2^2 \cdot 5 \cdot 17$, и $2^{340/17} \equiv 1 \pmod{341}$. Позитивный тест не проходит и 341 — составное число.

Если мы вместо 341 возьмем некоторое 40-значное число m, то для того чтобы использовать позитивный тест, нам придется разложить на множители число $m-1$, и с достаточно большой вероятностью после выделения маленьких простых делителей останется число из 35 или более цифр, которое необходимо будет разложить на множители — задача почти такая же сложная, как и разложение самого m. Поэтому тест Лукаса применим только в тех случаях, когда $m-1$ разлагается на множители просто.

Четвертый детерминистический тест. Этот тест был вначале разработан в 1980 г. Адлеманом, Померанцом и Рюмли (Adleman et al., 1983) и впоследствии улучшен Коэном и Ленстрой (Cohen, Lenstra, 1982). Его детали требуют знакомства с техникой алгебраической теории чисел, но по существу он близок к тесту Ферма.

Показано, что время работы для этого теста равно

$$O(L(m)^{L\{L[L(m)]\}}).$$

Несмотря на то что мы имеем дело с экспоненциальным тестом, выражение $L\{L[L(m)]\}$ в показателе стремится к бесконечности очень медленно; например, первое число, для которого $L\{L[L(m)]\} = 2$, равно $10^{999999999}$. Поэтому все числа, меньшие $10^{999999999}$, могут быть проверены на простоту за полиномиальное время. Таким образом, время работы этого теста очень близко к полиномиальному, и это лучшее, что известно на сегодняшний день. Неизвестно, существует ли полиномиальный детерминистический алгоритм проверки простоты.

Все приведенные выше тесты были детерминистическими; это означает, что для заданного числа m мы всегда получаем ответ, является ли оно простым или составным. Если заменить слово «всегда» на «с некоторой вероятностью», то оказывается возможным построить вероятностные тесты простоты, которые работают за полиномиальное время; такие тесты называют также тестами *псевдопростоты*. Ниже мы рассмотрим несколько таких тестов.

Более точно, под тестом псевдопростоты мы будем понимать тест, применяемый к паре целых чисел (b, m), обладающий следующими свойствами:

1. Тест может выдавать следующие ответы: «m — составное число» или «не удалось определить».
2. Если тест выдал ответ «m — составное число», то m — составное число.
3. Время выполнения теста полиномиально зависит от $L(m)$.

Для *хорошего* теста псевдопростоты существует фиксированное положительное вещественное число k, такое, что для любого составного целого числа m тест выдает ответ «составное» по крайней мере для km выборов различных оснований b, где $1 \leq b \leq m$. Кроме того, мы будем говорить, что целое число m *является простым с большой вероятностью*, если мы подвергли его хорошему тесту псевдопростоты и получили ответ «не удалось определить» для всех этих оснований b.

Первый вероятностный тест. Для заданного m выберем случайным образом b, $1 < b < m$. Если $b \mid m$, то тест выдает ответ «m — составное число», в противном случае — «не удалось определить».

Вероятность того, что выдается ответ «m — составное число», равна вероятности того, что $b \mid m$. Если $d(m)$ — число делителей m и b случайно выбрано в пределах $1 < b < m$, то вероятность этого равна $p = [d(m) - 2]/m$. Ясно, что это очень слабый тест.

Второй вероятностный тест. Для заданного m выберем случайным образом b, $1 < b < m$. Если $(b, m) \neq 1$, то тест выдает ответ «m — составное число», в противном случае — «не удалось определить».

Если m составное, то количество чисел $b < m$, для которых тест выдает ответ «m — составное число», равно $m - \phi(m)$. Это число велико, если m имеет маленькие простые делители. Если, однако, $m = pq$, где p, q — большие простые числа, то доля хороших оснований очень мала, и потому этот тест не лучше предыдущего.

Третий вероятностный тест. Если для заданных чисел b, m степень b^{m-1} не равна $1 \pmod{m}$, то тест выдает ответ «m — составное число», в противном случае — «не удалось определить».

Этот тест гораздо лучше двух предыдущих, но и он несовершенен, поскольку для всех псевдопростых по основанию b чисел он выдает ответ «не удалось определить». Как уже отмечалось, число 341 является псевдопростым по основанию 2. Из следующей теоремы вытекает, что существует бесконечно много псевдопростых по основанию 2 чисел.

Теорема 2.3.28. Если m — псевдопростое число по основанию 2, то то же самое верно для числа $n = 2^m - 1$.

Доказательство. Пусть число $m = ab$, $a > 1$, $b > 1$, является псевдопростым по основанию 2, т.е. $2^{m-1} \equiv 1 \pmod{m}$. Докажем, что $n = 2^m - 1$ также является псевдопростым по основанию 2, используя тождество

$$2^{ab} - 1 = (2^a - 1)(1 + 2^a + \cdots + 2^{a(b-1)}).$$

Более точно, для того чтобы доказать, что $2^{n-1} \equiv 1 \pmod{m}$, мы покажем, что $n = 2^m - 1$ делит $2^{n-1} - 1$. Согласно этому тождеству, достаточно доказать, что $m \mid (n-1)$. Поскольку m псевдопросто по основанию 2, то $m \mid (2^{m-1} - 1)$; кроме того, $n - 1 = 2^m - 1 - 1 = 2(2^{m-1} - 1)$, откуда следует, что $m \mid (n-1)$. □

Как мы уже отмечали, кроме псевдопростых существуют составные числа, которые называются *абсолютными псевдопростыми* или *кармайкловыми* числами, определяемые условиями $m \mid (b^{m-1} - 1)$ для всех b, таких, что $(b, m) = 1$.

Как может получиться, что $b^{m-1} - 1$ сравнимо с 1 по модулю m для всех b, $(b, m) = 1$? Например, для числа $m = 561$ имеется $\phi(561) = 320$ таких значений b. Ответ оказывается совсем простым. Все, что требуется, — это три или более нечетных простых числа p_i, такие, что $(p_i - 1) \mid (m-1) = (\prod p_i - 1)$ для каждого простого числа p_i; проверьте это для $561 = 3 \cdot 11 \cdot 17$. Докажем достаточность. Зафиксируем некоторое b, $(b, m) = 1$. По теореме Ферма $b^{p_i - 1} \equiv 1 \pmod{p_i}$. Поскольку $m - 1$ делится на $p_i - 1$, то то же самое верно и для $m - 1$: сравнение $b^{m-1} \equiv 1 \pmod{p_i}$ справедливо для любого i. Отсюда и из утверждения о единственности в греко-китайской теореме об остатках вытекает, что сравнение $b^{m-1} \equiv 1$ справедливо по модулю произведения чисел p_i. (Действительно, запишем сравнения $b^{m-1} \equiv 1 \pmod{p_i}$ в виде $x \equiv b^{m-1} - 1 \pmod{p_i}$. Они образуют систему, имеющую единственное решение по модулю m; конечно, это единственное решение есть $x = 0$, т.е. $b^{m-1} \equiv 1 \pmod{m}$.)

Таким образом, эти числа псевдопросты по любому основанию b, и для всех них третий вероятностный тест будет выдавать ответ «не удалось определить». Наименьшее такое число — это $561 = 3 \cdot 11 \cdot 17$; другие кармайкловы числа: $1105 = 5 \cdot 13 \cdot 17$, $1729 = 7 \cdot 13 \cdot 19$ и т.д. Следующий тест справляется с такими числами.

Четвертый вероятностный тест. Это тест сильной псевдопростоты. Пусть заданы b и m. Пусть $m - 1 = t2^s$, где t — нечетное число, и рассмотрим числа $x_r \equiv b^{t2^r} \pmod{m}$ для $0 \leq r < s$ (x_r —

наименьший по абсолютной величине остаток по модулю m). Если либо $x_0 = 1$, либо найдется индекс i, $0 \le i < s$, такой, что $x_i = -1$, то m называется *сильно псевдопростым* по основанию b и тест выдает ответ «не удалось определить», в противном случае ответ — «m — составное число».

Этот тест успешно применяется и к псевдопростым числам, таким, как $m = 561$. Например, $560 = 35 \cdot 2^4$; для $r = 4$, 3 имеем $2^{35 \cdot 2^4} = 2^{560} \equiv 1 \pmod{561}$ и $2^{35 \cdot 2^3} = 2^{280} \equiv 1 \pmod{561}$ соответственно, тогда как для $r = 2$ получаем $2^{35 \cdot 2^2} = 2^{140} \equiv 67 \pmod{561}$, и, следовательно, 561 — составное число. Итак, справедлива следующая теорема.

Теорема 2.3.29. Если тест сильной псевдопростоты выдает ответ «m — составное число», то m — составное число.

Доказательство. Докажем это от противного. Предположим, что m — нечетное простое число. Покажем по индукции, что $b^{t2^r} \equiv 1 \pmod{m}$ для любого r, $0 \le r \le s$, что будет противоречить условию теоремы. Очевидно, это справедливо для $r = s$ по теореме Ферма. Предполагая справедливость утверждения для i, нетрудно видеть, что оно справедливо и для $i - 1$, потому что равенство $(b^{t2^{i-1}})^2 = b^{t2^i} \equiv 1 \pmod{m}$ влечет за собой, что возводимое в квадрат число рано ± 1. Но -1 не подходит по условию (иначе бы тест выдал ответ «не удалось определить»). □

Анализ времени работы алгоритма четвертого вероятностного теста. Число b^t вычисляется за время $O[L^3(m)]$, поскольку в алгоритме быстрого возведения в степень выполняется $O[L(t)]$ умножений, $t \le m$, и, так как длины умножаемых по модулю m чисел $\sim L(m)$, каждое умножение выполняется за время $O[L^2(m)]$. После этого при вычислении последовательности b^{t2^i}, $i = 1, 2, \ldots$, производится r возведений в квадрат, где также $r \le L(m)$ и каждое возведение в квадрат выполняется за время $O[L^2(m)]$. Таким образом, тест выполняется за время $O[L^3(m)]$.

Доказано, что четвертый вероятностный тест обладает следующим свойством: если m — составное число, то вероятность того, что тест выдаст ответ «m — составное число», не меньше $1/2$. Основная идея доказательства состоит в том, что собственная подгруппа конечной группы не может содержать больше половины ее элементов; детали см. в (Wilf, 1986). Более того, Рабином было показано, что не существует нечетного составного числа m, которое является сильно псевдопростым по более чем $1/4$ части всех оснований, меньших m. На

практике для заданного числа m мы применяем тест 100 раз, используя 100 случайно и независимо выбранных оснований b_i, $0 \le b_i \le m$. Если m составное, то тест определит это с вероятностью $\ge 1 - 2^{-100}$, и каждая проверка выполняется за полиномиальное время.

2.3.4. Разложение на множители больших целых чисел

С задачей о нахождении делителей больших простых чисел дело обстоит гораздо хуже, чем с проверкой простоты. Мы не знаем даже, существует ли вероятностный алгоритм, который выдает за полиномиальное время делитель большого составного числа с вероятность $> 1/2$. Литература по этому предмету также весьма обширна (Dixon, 1981, 1984; Guy, 1976; Knuth, 1969; Lehman, 1974; Morrison et al., 1975; Williams, 1982 и 1984; Wunderlich, 1979); мы представим следующий метод разложения на множители (упрощенный вариант) для чисел общего вида, который является наиболее сильным из известных.

Метод основывается на идее Лежандра (1798): если $u^2 \equiv v^2 \pmod m$, $0 < u, v < m$, $u \not\equiv \pm v \pmod m$, то m делит $(u-v)(u+v)$, но не делит ни $(u-v)$, ни $(u+v)$; поэтому $(u-v, m)$, наибольший общий делитель чисел $u-v$ и m, является нетривиальным делителем m и может быть легко вычислен с помощью алгоритма Евклида. Поиск таких u и v происходит в два этапа, как описано ниже.

Пусть мы хотим разложить на множители число m. Пусть $n = [\sqrt{m}]$ — максимальное число, не превосходящее $\sqrt{m}$, и вычислим числа $a_k = (n+k)^2 - m$ для небольших k (числа k могут быть и отрицательными). [Вместо чисел a_k Моррисон и Бриллхарт (Morrison, Brillhart, 1975) использовали величины Q_k: если $\sqrt{cm} = (b_0; b_1, b_2, \ldots, b_{k-1}, \xi_k)$ — цепная дробь, представляющая число $\sqrt{cm}$, c — небольшой множитель, то Q_k определяются формулой $\xi_k = (P_k + \sqrt{cm})/Q_k$.]

Пусть $\{q_i,\ i = 1, 2, \ldots, j\}$ — множество небольших простых чисел, которые могут делить выражения вида $x^2 - m$ (т.е. m является квадратом по модулю q_i). Такое множество обычно называют *мультипликативной базой* B. Запомним все числа a_k, которые могут быть разложены по мультипликативной базе, т.е. записаны в виде

$$a_k = (-1)^{\omega_{k_0}} \prod_{1 \le i \le j} q_i^{\omega_{k_i}}.$$

Такие a_k называются B-числами. С каждым B-числом a_k связывается вектор показателей

$$\mathbf{e}_k = (w_{k_0}, w_{k_1}, \ldots, w_{k_j}), \quad w_{k_i} \equiv \omega_{k_i} \pmod 2,\ i = 0, 1, 2, \ldots, j.$$

Если мы найдем достаточно много B-чисел, чтобы множество соответствующих векторов показателей было линейно зависимо по модулю 2 (любое множество из $j+2$ B-чисел обладает этим свойством), то можно будет представить нулевой вектор в виде суммы векторов показателей некоторого множества S, скажем, $\sum_{k:a_k\in S}\mathbf{e}_k \equiv (0,\,0,\,\ldots,\,0) \pmod 2$. Определим теперь целые числа

$$e_i' = \frac{1}{2}\sum_{k:a_k\in S}\omega_{k_i}, \quad i=0,1,\ldots,j,$$
$$u = \prod_{k\in S}(n+k) \pmod m, \quad v = \prod_{1\le i\le j} q_i^{e_i'} \pmod m$$

Из сказанного выше следует, что $u^2 \equiv v^2 \pmod m$, и $(u-v,m)$ может быть нетривиальным делителем m.

Пример. Разложим на множители 1729, третье кармайклово число. В этом случае $m=1729$, $n=41$; вычислим числа $a_k=(n+k)^2-m$ для небольших k. Имеем $a_1=35$, $a_2=120$, $a_3=207$, $a_4=296$, $a_5=387$, $a_6=480$, $a_7=575$, $a_8=672$, $a_9=771$ и т.д. Зафиксируем множество небольших простых чисел $\{2,3,5,7\}$; мы видим, что только 35, 120, 480 и 672 являются B-числами, а именно $35=(-1)^0\cdot2^0\cdot3^0\cdot5^1\cdot7^1$, $120=(-1)^0\cdot2^3\cdot3^1\cdot5^1\cdot7^0$, $480=(-1)^0\cdot2^5\cdot3^1\cdot5^1\cdot7^0$ и $672=(-1)^0\cdot2^5\cdot3^1\cdot5^0\cdot7^1$. Кроме 480, все числа включаются в множество S, поскольку сумма векторов показателей всех чисел, кроме 480, равна $(0,\,0,\,0,\,0,\,0)$. Далее вычисляются показатели $e_0'=0$, $e_1'=8/2=4$, $e_2'=2/2=1$, $e_3'=2/2=1$, $e_4'=2/2=1$ и числа $u=(41+1)\cdot(41+2)\cdot(41+8)\equiv315 \pmod{1729}$ (где $(41+1)^2\equiv a_1=35$, $(41+2)^2\equiv a_2=120$, $(41+8)^2\equiv a_8=672 \pmod{1729}$) и $v=(-1)^0\cdot2^4\cdot3^1\cdot5^1\cdot7^1=1680$. Квадраты чисел u и v совпадают по модулю 1729: $u^2=99225\equiv672$, $v^2=2822400\equiv672 \pmod{1729}$. Имеем $(315-1680,1729)=(-1365,1729)=91$, и 91 является делителем числа 1729.

Диксоном (Dixon, 1981) доказано, что число m разлагается этим методом на множители за время

$$O\bigl(e^{(\alpha+o(1))\cdot\sqrt{(\ln m)(\ln\ln m)}}\bigr),$$

где α — некоторая константа, $o(1)\to0$ при $m\to\infty$. Эта величина растет медленнее, чем экспонента, но быстрее, чем любая степень числа $L(m)$.

Отметим, что числа вида $F_m = 2^{2^m} + 1$ и $M_m = 2^m - 1$, которые называются соответственно числами Ферма и Мерсенна, оказали глубокое влияние на развитие техники разложения на множителм и проверки простоты. Еще в 1640 г. Ферма предположил, что любое число F_m, $m \geq 1$, является простым. Однако это предположение было опровергнуто в 1729 г. Эйлером, который показал, что число F_5 делится на 641.

2.4

Точные вычисления, использующие модулярную арифметику

Основываясь на результатах теории чисел, представленных в предыдущих разделах, мы собираемся обсудить интересный способ выполнения точных арифметических действий с (большими) целыми числами. Основная идея состоит в том, чтобы использовать один или несколько модулей (теорема 2.3.25), не имеющих общих делителей, и вместо действий с самими числами выполнять действия с их остатками (Gregory et al., 1984; Knuth, 1969; Scott, 1985).

Рассмотрим сначала случай одного модуля. Пусть нам дано выражение $e(i_1, i_2, \ldots, i_h)$ над $\mathbb{Z}$, зависящее от целочисленных аргументов $i_1, i_2, \ldots, i_h$, и нужно вычислить (оценить) его. Тривиальный подход состоит в непосредственном вычислении выражения над $\mathbb{Z}$; при этом, однако, промежуточные результаты могут не быть целыми числами (могут иметь бесконечные представления, как, например, $1/3 = 0.333\ldots$), и отбрасывание цифр или округление может привести к неточности окончательного результата. Для того чтобы избежать этого, нужно воспользоваться обходным путем при вычислении e. Этот подход проиллюстрирован на рис. 2.4.1.

Вместо вычисления e над $\mathbb{Z}$ мы сначала по выражению $e(i_1, i_2, \ldots, i_h)$ над $\mathbb{Z}$ получаем эквивалентное выражение $e(i'_1, i'_2, \ldots, i'_h)$ над $\mathbb{Z}_m$ для некоторого m, где $i'_j \equiv i_j \pmod m$, или, что эквивалентно, $i'_j \equiv r_m(i_j)$, $j = 1, 2, \ldots, h$. Затем мы вычисляем выражение e_m над $\mathbb{Z}_m$ и получаем эквивалентный результат res_m, где $\mathrm{res}_m \equiv \mathrm{res} \pmod m = r_m(\mathrm{res})$; в завершение мы отображаем res_m обратно в множество целых чисел.

Следует отметить, что отношение эквивалентности $\mathrm{res} \equiv \mathrm{res}_m \pmod m$ не определяет *однозначно* окончательный результат res.

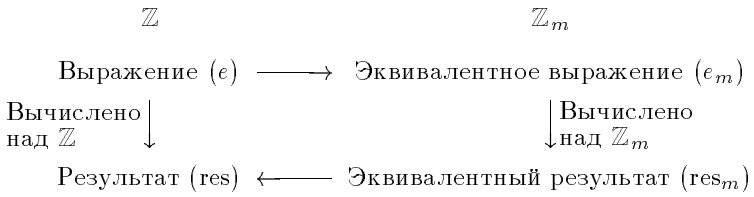


Рис. 2.4.1.

Окольный путь вычисления выражения (e), использующий один модуль.

[Аналогия с задачей «чему равно значение $x \equiv 7 \pmod{13}$?». Очевидно, что значение x не определено однозначно, оно может быть равным 7, 20 и т.д. Если, однако, мы откуда-либо знаем, что $x < 13$, то x может быть равным только 7.] Для того чтобы однозначно определить res, нужно иметь априорную оценку его величины; эта оценка используется в качестве модуля m, и все операции выполняются в кольце $\mathbb{Z}_m$. Если мы имеем оценку на величину res, то мы ищем наименьшее неотрицательное решение уравнения $\mathrm{res} \equiv \mathrm{res}_m \pmod{m}$; если же мы имеем оценку на $|\mathrm{res}|$, то ищем наименьшее по абсолютной величине решение.

Сложности с изложенным методом обхода возникают, когда выполняются операции деления. Как уже отмечалось, в случае когда p — простое число, кольцо $(\mathbb{Z}_p, +, \cdot)$ является конечным полем [оно обозначается также $GF(p)$], в котором мы можем выполнять все арифметические операции. Поскольку обратный элемент к любому ненулевому элементу всегда существует, мы определяем деление по модулю p следующим образом:

$$\frac{a}{b} \pmod{p} = a \cdot [b^{-1} \pmod{p}] \pmod{p},$$

где, как отмечалось выше, $b^{-1} \pmod{p}$ — мультипликативный обратный элемент к элементу b по модулю p, который мы будем также обозначать просто через b^{-1}. Частное двух целых чисел в $GF(p)$ также является целым числом, даже если b не делит a в $\mathbb{Z}$.

Пример.

$$\begin{aligned} 3/4 \pmod{11} &\equiv 3 \cdot 4^{-1} \pmod{11} \\ &\equiv 3 \cdot 3 \pmod{11} \\ &\equiv 9. \end{aligned}$$

Число, полученное в этом примере (рассматриваемое как промежуточный результат), имеет смысл, поскольку

$$\begin{aligned}(3/4)\cdot 4 \pmod{11} &\equiv [3/4 \pmod{11}]\cdot 4 \pmod{11}\\ &\equiv 9\cdot 4 \pmod{11} \equiv 3.\end{aligned}$$

Предположим теперь, что модуль p ограничивает окончательный результат res. Если res не является целым числом, принадлежащим $GF(p)$, то $\text{res}_p \neq \text{res}$, и для того чтобы получить последнее число, требуется дополнительная информация; эта априорная информация различна для разных случаев и проиллюстрирована ниже двумя примерами. Если, однако, res лежит в $GF(p)$, то $\text{res}_p = \text{res}$. [Напомним еще раз, что res — это результат, полученный при вычислении выражения над кольцом целых чисел, тогда как res_p получается при вычислении в $GF(p)$.] Таким образом, арифметика одного модуля может быть использована для выполнения последовательности точных арифметических операций над целыми числами в $GF(p)$, даже если эта последовательность включает операции деления; трудности могут возникнуть лишь при интерпретации результатов.

Когда мы хотим вычислить выражение, окончательное значение которого может быть положительным или отрицательным, нам нужно уметь работать с отрицательными числами. Это можно делать, используя симметричное множество $\mathbb{Z}_p = \{-(p-1)/2, \ldots, -2, -1, 0, 1, 2, \ldots, (p-1)/2\}$, которое, как нетрудно видеть, изоморфно множеству $\{0, 1, \ldots, p-1\}$; отображение между этими двумя множествами показано на рис. 2.4.2.

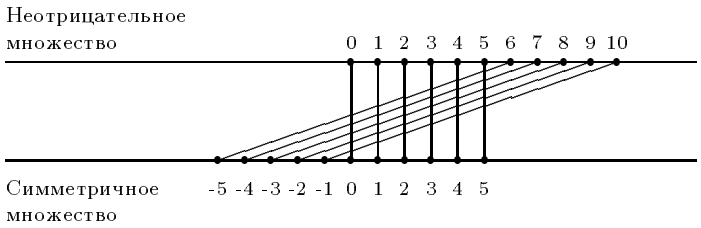


Рис. 2.4.2.

Отображение между симметричным и неотрицательным множествами для $p = 11$.

Например, $x \equiv 17 \pmod{11}$ имеет наименьшее неотрицательное решение $x = 6$ и наименьшее по абсолютной величине решение $x = -5$.

Конечно, точные арифметические операции можно выполнять в любом из этих двух множеств, но проще отобразить наши данные (особенно если они содержат отрицательные числа) в неотрицательное множество, выполнить в нем все операции и затем отобразить обратно в симметричное множество.

Пример. Выполним точные арифметические операции в $GF(11)$, для того чтобы вычислить $x = 1/3 - 4/3$; в этом примере априорная информация состоит в том, что мы ищем результат в симметричном множестве.

$$\begin{aligned} x \pmod{11} &\equiv (1/3 + (-4)/3) \pmod{11} \\ &\equiv (1/3 + 7/3) \pmod{11} && \text{(в неотрицательном множестве)} \\ &\equiv (1 \cdot 3^{-1} + 7 \cdot 3^{-1}) \pmod{11} && (3^{-1} \text{ является мультипликативным обратным к } 3 \pmod{11}) \\ &\equiv (1 \cdot 4 + 7 \cdot 4) \pmod{11} \\ &\equiv 32 \pmod{11} \end{aligned}$$

Отображая результат обратно в симметричное множество, мы получаем правильный ответ $x = -1$.

В приведенном примере $\text{res}_p = \text{res}$, потому что мы знаем с самого начала, что res лежит в симметричном множестве для $GF(p)$. В следующем примере это не так.

Пример. Вычислим $x = 1/2 - 2/3$, пользуясь точно такой же априорной информацией, как и в предыдущем примере. Проводя вычисления, как и в предыдущем примере, получим

$$\begin{aligned} x \pmod{11} &\equiv (1/2 + (-2)/3) \pmod{11} \\ &\equiv (1/2 + 9/3) \pmod{11} \\ &\equiv (1 \cdot 2^{-1} + 9 \cdot 3^{-1}) \pmod{11} \\ &\equiv (1 \cdot 6 + 9 \cdot 4) \pmod{11} \\ &\equiv 42 \pmod{11} \\ &\equiv 9. \end{aligned}$$

Если сейчас мы отобразим результат обратно в симметричное множество, то получим неправильный ответ $x = -2$. Поэтому нам требуется дополнительная информация: достаточно знать, что мы ищем рациональное число $x \pmod{11} = (a/b) \pmod{11}$, где $b = 6$ — наименьшее

общее кратное знаменателей двух дробей. Тогда $a = [x \pmod{11}] \cdot b \pmod{11} = 9 \cdot 6 \pmod{11} = 10 \pmod{11} = -1$ (отображение на симметричное множество). Следовательно, $x = (-1)/6$, что является правильным ответом.

Перейдем теперь к арифметике нескольких модулей. Использование арифметики нескольких модулей является «естественным» решением проблемы, с которой мы сталкиваемся в арифметике одного модуля: как мы видели, модуль m должен быть достаточно большим, чтобы результат res определялся однозначно по его остатку res_m ($m > \mathrm{res}$). Однако, когда нам приходится выбирать значение m большим, чем размер компьютерного слова, схема с одним модулем теряет всю свою привлекательность. Поэтому мы используем несколько модулей и применяем окольный путь для вычисления выражения; эта схема иллюстрируется на рис. 2.4.3.

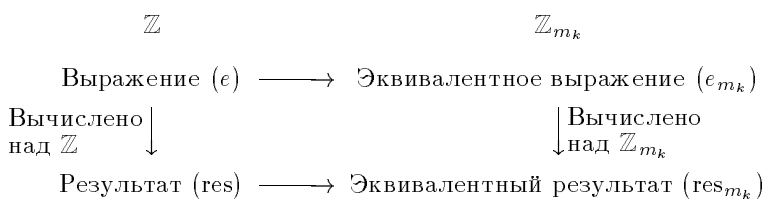


Рис. 2.4.3.

Окольный путь вычисления выражения (e), использующий n модулей $(k = 1, 2, \ldots, n)$.

Для заданного выражения $e(i_1, i_2, \ldots, i_h)$, зависящего от целочисленных аргументов $i_1, i_2, \ldots, i_h$, мы сначала вычисляем $e_{m_k}(i_{1k}, i_{2k}, \ldots, i_{hk})$, где $i_{jk} = r_{m_k}(i_j)$, $j = 1, 2, \ldots, h$ и $k = 1, 2, \ldots, n$, для коротких модулей m_k. При условии, что выражения e_{m_k} определены над $\mathbb{Z}_{m_k}$, мы вычисляем их над $\mathbb{Z}_{m_k}$ и получаем эквивалентные результаты res_{m_k}, $k = 1, 2, \ldots, n$; в завершение, пользуясь греко-китайским алгоритмом либо таблицами, мы получаем окончательный результат res. Здесь снова модули m_k должны быть выбраны таким образом, чтобы $m_1 \cdot m_2 \cdots m_n > \mathrm{res}$. Если дана оценка на res, то мы ищем наименьшее неотрицательное решение греко-китайской задачи об остатках, если же оценивается $|\mathrm{res}|$, то ищется наименьшее по абсолютной величине решение.

Рассмотрим более подробно арифметику нескольких модулей. Знакомые нам всем системы счисления являются линейными, позицион-

ными и весовыми. Это означает, что всем позициям соответствуют веса, зависящие от одного основания. Например, в десятичной системе счисления используются веса 10^0, 10^1, 10^2, 10^3 и т.д. Вместо этого многомодульная система счисления использует взаимно простые позиционные основания, например 3, 5, 7. В табл. 2.4.1 перечислены числа 0–29 и их остатки по основаниям 3, 5 и 7.

Таблица 2.4.1

Остатки чисел 0–29 по основаниям 3, 5, 7

	остатки				остатки				остатки		
N	3	5	7	N	3	5	7	N	3	5	7
0	0	0	0	10	1	0	3	20	2	0	6
1	1	1	1	11	2	1	4	21	0	1	0
2	2	2	2	12	0	2	5	22	1	2	1
3	0	3	3	13	1	3	6	23	2	3	2
4	1	4	4	14	2	4	0	24	0	4	3
5	2	0	5	15	0	0	1	25	1	0	4
6	0	1	6	16	1	1	2	26	2	1	5
7	1	2	0	17	2	2	3	27	0	2	6
8	2	3	1	18	0	3	4	28	1	3	0
9	0	4	2	19	1	4	5	29	2	4	1

Остатки в табл. 2.4.1 однозначно определяют число; всего имеется $3 \cdot 5 \cdot 7 = 105$ различных чисел, представимых в этой системе. Например, вектор $[2, 1, 3]$ однозначно определяет десятичное число 8 и называется *стандартным набором остатков числа* 8 *относительно данного вектора оснований*. В нашем случае вектор $\beta = [3, 5, 7]$ является *вектором оснований*. Мы будем пользоваться обозначением

$$8 \pmod{\beta} = [8 \pmod 3,\ 8 \pmod 5,\ 8 \pmod 7] = [2, 3, 1].$$

Как и в случае одного модуля, мы также можем определить *наименьшую неотрицательную числовую систему* и при условии, что все модули нечетные, *наименьшую по абсолютной величине числовую систему*, или *симметричную систему остатков*, относительно данного вектора оснований β.

Рассмотрим теперь вектор оснований общего вида $\beta = [m_1, m_2, \ldots, m_n]$, $(m_i, m_j) = 1$ для $i \neq j$, и пусть $\mathrm{M} = m_1 \cdot m_2 \cdots m_n$. (Отметим, что, поскольку модули попарно взаимно просты, M является их наименьшим общим кратным.) Справедлив следующий важный результат.

Теорема 2.4.1. Два целых числа n_1 и n_2 имеют одинаковые стандартные наборы остатков относительно вектора оснований $\beta = [m_1, m_2, \ldots, m_n]$ тогда и только тогда, когда $n_1 \equiv n_2 \pmod{m_1 \cdot m_2 \cdots m_n}$.

Доказательство. Доказательство оставляется читателю в качестве упражнения. □

Пример. Рассмотрим $\beta = [3, 5, 7]$, как в табл. 2.4.1, где $M = 105$. Тогда $9 \equiv 114 \pmod{105}$, и, значит, $9 \pmod{\beta} = [0, 4, 2] = 114 \pmod{\beta}$.

Из теоремы 2.4.1 следует, что множество $\mathbb{Z}_\beta = \{n \pmod{\beta} : n \in \mathbb{Z}\}$ содержит M элементов, которые взаимно однозначно отображаются на элементы множества $\mathbb{Z}_M$. Нетрудно видеть, что два множества $\mathbb{Z}_\beta$ и Z_M с соответствующими операциями сложения и умножения представляют собой *изоморфные* конечные коммутативные кольца, и, следовательно, *многомодульная арифметика эквивалентна арифметике по модулю M*.

Главное преимущество многомодульной числовой системы состоит в отсутствии переносов при выполнении операций сложения и умножения. Арифметика замкнута в каждой позиции (т.е. арифметические действия выполняются полностью и независимо в разных позициях). Поэтому можно выполнять сложение и умножение длинных целых чисел так же быстро, как и обычных (коротких) чисел.

На двоичных компьютерах удобно использовать модули вида $m = 2^e - 1$, т.е. каждый модуль на единицу меньше, чем степень двойки. Имеем

$$u + v \pmod{2^e - 1} = \begin{cases} u + v, & \text{если } u + v < 2^e - 1, \\ \{u + v - 2^e\} + 1, & \text{если } u + v \geq 2^e - 1, \end{cases}$$
$$u \cdot v \pmod{2^e - 1} = (uv \mod 2^e) + \lfloor uv/2^e \rfloor.$$

В последнем выражении мы складываем младший и старший разряды произведения. Как мы видим, действия по модулю $2^e - 1$ выполняются сравнительно просто. (Если модуль m не имеет такого вида, то при сложении нужно взять $u + v - m$, когда $u + v \geq m$.)

В некоторых случаях необходимо знать, являются ли модули взаимно простыми. Если они имеют вид $2^e - 1$, то для проверки этого

можно использовать следующее простое правило: $(2^e - 1, 2^f - 1) = 2^{(e,f)} - 1$. Из него вытекает, что модули взаимно просты тогда и только тогда, когда числа e и f взаимно просты. Это правило следует из алгоритма Евклида и тождества $2^e - 1 \pmod{2^f - 1} = 2^{e \pmod f} - 1$.

Пример. В табл. 2.4.1 рассмотрим числа $4 = [1, 4, 4]$ и $5 = [2, 0, 5]$; их сумма равна $[1 + 2 \pmod 3, 4 + 0 \pmod 5, 4 + 5 \pmod 7] = [0, 4, 2] = 9$. Точно так же их произведение равно $[1 \cdot 2 \pmod 3, 4 \cdot 0 \pmod 5, 4 \cdot 5 \pmod 7] = [2, 0, 6] = 20$. Заметим, что при отсутствии таблицы ответ восстанавливается с помощью греко-китайского алгоритма. Кроме того, если результат операции больше, чем M, то мы не получим правильного ответа; поэтому критическим моментом является правильный выбор вектора оснований.

Мы видим, что время, которое требуется для сложения, вычитания и умножения двух n-значных чисел при использовании многомодульной арифметики, равно $O[L(n)]$ без учета времени преобразования к модулярному представлению и обратно. Для сложения и вычитания здесь нет никаких преимуществ, но для умножения может быть достигнуто существенное улучшение по сравнению с обычным методом, требующим времени $O[L^2(n)]$. Многомодульная арифметика также идеально подходит для параллельных компьютеров.

Для выполнения деления определим $b^{-1} \pmod \beta$, мультипликативный обратный к элементу $b = [b_1, b_2, \ldots, b_n]$ по модулю вектора оснований $m = [m_1, m_2, \ldots, m_n]$, следующим образом:

$$b^{-1} \pmod \beta = [b_1^{-1} \pmod{m_1}, b_2^{-1} \pmod{m_2}, \ldots, b_n^{-1} \pmod{m_n}].$$

Далее, если $a = [a_1, a_2, \ldots, a_n]$, то

$$\frac{a}{b} \pmod \beta = [a_1 \cdot b_1^{-1} \pmod{m_1}, a_2 \cdot b_2^{-1} \pmod{m_2}, \ldots, a_n \cdot b_n^{-1} \pmod{m_n}].$$

Конечно, как и в случае одного модуля, если b не делит a, то результат не может быть проинтерпретирован без дополнительной информации; однако он допустим в качестве промежуточного результата.

Основная трудность при работе с многомодульными числовыми системами заключается в сравнении величины целых чисел. Конечно, можно использовать симметричную систему остатков, вычесть из одного числа другое и затем определить знак разности. К сожалению, проблема этим не решается, поскольку остатки в симметричной системе не несут информации о знаке числа; один из способов определения

знака числа x состоит в обратном преобразовании к обычному виду, что разрушает саму идею многомодульной арифметики. Задача определения знака числа может быть решена гораздо лучшим способом с помощью преобразования числа x к так называемому представлению *со смешанными основаниями* (*mixed-radix* representation). При этом преобразовании мы выполняем *только* операции многомодульной арифметики. Мы уже встречались с представлением со смешанными основаниями, когда выражали решение x в греко-китайском алгоритме в виде

$$x = q_1 + q_2 \cdot m_1 + q_3 \cdot m_1 \cdot m_2 + \cdots + q_n \cdot m_1 \cdot m_2 \cdots m_{n-1}, \qquad (*)$$

где каждое q_i не превосходит соответствующего модуля m_i; q_n называется *старшим членом* числа x. В этой форме знак числа x совпадает со знаком его старшего члена. (Отметим, что в форме со смешанными основаниями мы имеем $x = \sum_{1 \le i \le n} q_i \cdot M_i$, где $M_i = m_1 \cdot m_2 \cdots m_{i-1}$ и $M_1 = 1$; частные M_i/M_{i-1} различны для различных позиций i. Если $m_1 = m_2 = \cdots = m_n$, то мы имеем представление *с фиксированным основанием*; в частности, при $m_1 = m_2 = \cdots = m_n = 10$ получается хорошо знакомое десятичное представление.) При определении знака удобно, чтобы последний модуль в векторе оснований был равен 2, поскольку нужно знать, в какой половине множества возможных чисел лежит результат. Например, на рис. 2.4.2 числа $0, 1, \ldots, 5$ образуют нижнюю половину множества возможных чисел, соответствующую значению $q_n = 0$, а числа $6, 7, \ldots, 10$ — верхнюю половину, соответствующую значению $q_n = 1$.

Предположим теперь, что нам дано представление $x = [a_1, a_1, \ldots, a_n]$ относительно вектора оснований $\beta = [m_1, m_1, \ldots, m_n]$, и мы хотим определить знак числа x. Нам нужно преобразовать x к форме со смешанными основаниями и определить знак старшего члена. Для этого необходимо вычислить цифры $q_1, q_2, \ldots, q_n$, упомянутые выше. Очевидно, что из $(*)$ вытекает

$$x \equiv q_1 \pmod{m_1},$$

и, следовательно, $q_1 = a_1$. Мы получили первую цифру. Далее возьмем разность $x - q_1$ (вычитая q_1 из *каждого* остатка, представляющего x). Имеем

$$x - q_1 = q_2 \cdot m_1 + q_3 \cdot m_1 \cdot m_2 + \cdots + q_n \cdot m_1 \cdot m_2 \cdots m_{n-1}.$$

Первая цифра (в смешанном представлении) числа $x - q_1$ равна нулю, и, следовательно, первые цифры всех последующих чисел можно

будет не рассматривать. Будем считать, таким образом, что размерность вектора $x - q_1$ равна $n - 1$. Найдем затем $(m_1)^{-1} \pmod{\beta_r}$, (многомодульный) мультипликативный обратный к элементу m_1 по модулю β_r, где $\beta_r = [m_2, \ldots, 2]$ (также размерности $n - 1$), и вычислим (многомодульное) произведение $(x - q_1) \cdot (m_1)^{-1}$, для того чтобы получить вторую цифру q_2. Будем продолжать этот процесс до тех пор, пока не вычислим q_n; если его значение равно 0 или 1, то число x соответственно положительное или отрицательное, потому что лежит соответственно в нижней или верхней половине возможного множества значений.

Пример. Определим знак числа $x = [4, 2, 0, 1]$ с вектором оснований $\beta = [7, 5, 3, 2]$. Очевидно, что $q_1 = 4$, и $x' = x - 4 = [0, 3, 2, 1]$, или, как было объяснено выше, $x' = [3, 2, 1]$; мы сократили вектор оснований до $\beta_r = [5, 3, 2]$. Для того чтобы получить вторую цифру q_2, вычислим $(m_1)^{-1} \pmod{\beta_r} = 7^{-1} \pmod{\beta_r} = [3, 1, 1]$; умножив x' на этот элемент, получим $[4, 2, 1]$. Следовательно, $q_2 = 4$, и, вычитая q_2 из последнего модулярного выражения, получим $x'' = [0, 1, 1]$, или $x'' = [1, 1]$. (Теперь мы уже сократили вектор оснований до $\beta_r = [3, 2]$.) Далее вычисляется $(m_2)^{-1} \pmod{\beta_r} = 5^{-1} \pmod{\beta_r} = [2, 1]$, и, умножая x'' на этот элемент, мы получаем $[2, 1]$; поэтому $q_3 = 2$. Вычитая q_3 из последнего модулярного выражения, получаем $x''' = [0, 1]$, или $x''' = 1$. (Теперь $\beta_r = [2]$.) В заключение мы вычисляем $(m_3)^{-1} \pmod{\beta_r} = 3^{-1} \pmod{\beta_r} = [1]$, и, умножая x''' на этот элемент, получаем $[1]$, откуда следует, что $q_4 = 1$. Поэтому x является отрицательным числом; действительно, $x = -3$.

Упражнения

Раздел 2.1.1

1. Завершите доказательство предложения 2.1.2.

2. Пусть A, B и C обозначают множества.

- **a.** Покажите, что $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ (объединение дистрибутивно относительно пересечения).
- **b.** Покажите, что $B \subset A \iff B \cap A = B$.
- **c.** Покажите, что множества $A - B$, $A \cap B$ и $B - A$ попарно не пересекаются и $A \cup B = (A - B) \cup (A \cap B) \cup (B - A)$.

3. (Индукция.) Пусть $p(n)$ — утверждение о натуральном числе n. Допустим, что любое непустое множество натуральных чисел имеет наименьший элемент (это принцип полной упорядоченности, доказательство которого дано в разд. 2.2.1). Покажите, что если (i) $p(0)$ справедливо и (ii) из справедливости $p(n)$ вытекает справедливость $p(n+1)$, то $p(n)$ справедливо для любого n. [*Указание.* Если нет, то множество $S = \{m\colon\ p(m)$ не выполняется$\}$ имеет наименьший элемент m_0.]

4. (Законы Де Моргана.) Для данного множества $C \subset S$ положим $C' = \{x \in S$ и $x \notin C\}$. Пусть A, B – два подмножества множества S. Докажите, что имеют место следующие равенства множеств:

a. $(A \cup B)' = A' \cap B'$.

b. $(A \cap B)' = A' \cup B'$.

5. Бинарная операция на множествах, называемая *симметрической разностью*, определяется формулой $A \oplus B = (A \setminus B) \cup (B \setminus A)$.

a. Докажите, что $A \oplus B = (A \cup B) \setminus (A \cap B)$.

b. Чему равно $A \oplus A$ для произвольного множества A?

c. Докажите, что $(A \oplus B) \oplus C = A \oplus (B \oplus C)$ для любых множеств A, B и C.

6. (Принцип включения и исключения.) Если S – конечное множество, то число элементов множества S обозначается $|S|$. Покажите, что для конечных множеств A и B имеем $|A \cup B| = |A| + |B| \setminus |A \cap B|$. По индукции это соотношение может быть распространено на случай $|A_1 \cup A_2 \cup \cdots \cup A_n|$.)

7. (Парадокс Рассела.) Множества могут содержать множества в качестве элементов. Пусть B — множество $B = \{S : S$ — множество и $S \notin S\}$. Приведите рассуждения, показывающие, что и $B \in B$, и $B \notin B$ истинно (см. также историческое замечание 1).

Раздел 2.1.2

1. Завершите доказательство предложения 2.1.4.

2. **a.** Покажите, что $\equiv_m$ является отношением эквивалентности на $\mathbb{Z}$ и что классом эквивалентности целого числа a служит множество $\mathbf{a} = a + m\mathbb{Z}$.

b. Покажите, что множество классов эквивалентности равно $\mathbb{Z}/\equiv_m = \{\mathbf{0}, \mathbf{1}, \ldots, \mathbf{M-1}\}$.

3. Для различных множеств S (определенных ниже), проверьте следующие бинарные отношения ρ на рефлексивность, симметричность и транзитивность:

a. $S = \mathbb{Q}$; $x \rho y$ тогда и только тогда, когда $|x| \leq y$.

b. $S = \mathbb{Z}$; $x \rho y$ тогда и только тогда, когда $x - y$ нацело делится на 5.

c. $S = \mathbb{N}$; $x \rho y$ тогда и только тогда, когда $x \cdot y$ четно.

d. $S = \mathbb{N}$; $x \rho y$ тогда и только тогда, когда $x \cdot y$ нечетно.

4. Определите, какие из бинарных отношений в упр. 3 являются отношениями эквивалентности, если такие среди них есть, и опишите соответствующие классы эквивалентности.

5. Пусть $S = \{0, 1, 2, 4, 6\}$. Проверьте следующие бинарные отношения на рефлексивность, симметричность и транзитивность:

a. $\rho = \{(0,0),\ (1,1),\ (2,2),\ (4,4),\ (6,6),\ (0,1),\ (1,2),\ (2,4),\ (4,6)\}$.

b. $\rho = \{(0,1), (1,0), (2,4), (4,2), (4,6), (6,4)\}$.

6. Дано разбиение $\{a, b, c\}$ и $\{d, e\}$ множества $\{a, b, c, d, e\}$. Перечислите упорядоченные пары в соответствующем отношении эквивалентности.

7. Вычислите число всех возможных разбиений (a) 4-элементного множества; (b) 5-элементного множества.

Раздел 2.1.3

Обозначение в упр. **2** и **3**: для натурального числа n положим $\tilde{n} = \{1, \ldots, n\}$ — множество из n элементов; $\tilde{0} = \emptyset$.

1. Завершите доказательство предложения 2.1.9.

2. Покажите по индукции, что множество всех биекций множества из n элементов $Bij(\tilde{n})$ содержит $n!$ элементов.

3. Покажите по индукции, что множество всех подмножеств $\wp(\tilde{n})$ имеет 2^n элементов.

4. Покажите, что логарифм по любому основанию $\log: R^+ \to R$ является биекцией. Чему равна функция $\log^{-1}$?

5. Найдите бесконечное множество S и функцию $f : S \to S$ такую, что

a. f взаимно однозначна, но не на.

b. f отображает на, но не взаимно однозначна.

6. Пусть f — функция $f : S \to T$. Покажите, что $f(A \cap B) \subset f(A) \cap f(B)$ для всех подмножеств A и B множества S и, в частности, $f(A \cap B) = f(A) \cap f(B)$ тогда и только тогда, когда f взаимно однозначна.

7. Пусть $f : S \to T$ и $g : T \to U$ — функции.

a. Докажите, что если функция $g \circ f$ взаимно однозначна, то взаимно однозначна и f.

b. Найдите пример, когда $g \circ f$ взаимно однозначна, но g таковой не является.

c. Докажите, что если функция $g \circ f$ отображает на, то этим свойством обладает и g.

d. Найдите пример, когда $g \circ f$ отображает на, но f этим свойством не обладает.

8. Пусть f — функция $f : S \to T$. Определим бинарное отношение ρ на S правилом: $x \rho y$ тогда и только тогда, когда $f(x) = f(y)$. Докажите, что ρ является отношением эквивалентности.

Раздел 2.2.1

1. Покажите, что не каждое непустое подмножество вещественных чисел имеет минимальный и максимальный элемент.

2. Покажите, что если $a|b$ и $b|a$, то $a = \pm b$.

3. Покажите, что если $d \mid a, b$, то $d \mid ax + by$ для любых x, y.

4. Покажите, что если $a \mid bc$ и $(a, b) = 1$, то $a \mid c$. (*Указание.* Используйте теорему 2.2.3.)

5. Покажите, что если a и b оба ненулевые, то $m = ab/(a, b)$ — наименьшее общее кратное чисел a и b.

6. Докажите часть **b** теоремы 2.2.4. (*Указание.* Сначала покажите, что любое решение однородного уравнения $ax + by = 0$ имеет вид $x = -nb/d$, $y = na/d$ для некоторого n.)

7. Покажите, что $\gcd(a, b) = \gcd(a, b + ax)$ для всякого целого x.

8. Докажите, что если $d = \gcd(a, b)$ и $d = ax + by$, то $\gcd(x, y) = 1$.

9. Пусть $a_1, \ldots, a_n$ — целые числа. Покажите, что уравнение $a_1x_1 + \cdots + a_nx_n = b$ имеет целое решение тогда и только тогда, когда $\gcd(a_1, \ldots, a_n) \mid b$, где $\gcd(a_1, \ldots, a_n) = \gcd[a_1, \gcd(a_2, \ldots, a_n)]$, $n \geq 2$.

10. Воспользуемся рядами Фарея для доказательства частного случая теоремы 2.2.3. А именно если $\gcd(a, b) = 1$ и $0 \leq a \leq b$, то целые числа x, y, удовлетворяющие соотношению $ax + by = 1$, могут быть получены из ряда Фарея F_b.

a. *Определение.* Ряд Фарея порядка N, обозначаемый F_N, — это множество всех приведенных дробей между 0 и 1, знаменатели которых $\leq N$, расположенных в порядке

возрастания. (Дробь a/b приведена, если $\gcd(a,b) = 1$.) Например, ряд Фарея порядка 3 равен

$$F_3 = 0/1, 1/3, 1/2, 2/3, 1/1.$$

b. *Построение рядов Фарея.* Для получения F_N в общем случае начинаем с $F_1 = 0/1, 1/1$ и повторяем следующую операцию столько раз, сколько потребуется (все знаменатели должны быть меньше или равны N):

Вставить $(a+a')/(b+b')$ между двумя соседними дробями a/b и a'/b'. (Новая дробь $(a + a')/(b + b')$ называется *медиантой* дробей a/b и a'/b'.) Например, чтобы вычислить F_3, первый шаг дает нам один новый элемент между $0/1$ и $1/1$:

$$0/1, 1/2, 1/1,$$

а следующий (последний) дает еще два:

$$0/1, 1/3, 1/2, 2/3, 1/1.$$

Отметим, что для получения F_N из F_{N-1} мы вставляем дробь $(a + a')/N$ между двумя соседними дробями a/b и a'/b' ряда F_{N-1}, знаменатели которых в сумме дают N. Например, чтобы получить F_4 из элементов ряда F_3, мы просто вставляем $1/4$ и $3/4$ (почему не $2/4$?) в соответствии со сформулированным правилом:

$$F_4 = 0/1,\ 1/4,\ 1/3,\ 1/2,\ 2/3,\ 3/4,\ 1/1.$$

c. Проверьте, что если $a/b < a'/b'$ и если все значения неотрицательны, то

$$a/b < (a + a')/(b + b') < a'/b'.$$

d. Докажите по индукции, что если a/b и a'/b' — две соседние дроби (на любом шаге построения), то

$$a'b - ab' = 1.$$

(*Указание.* Изначально это верно; посмотрите, что случится, когда вставляется новая медианта, и проверьте, что получен инвариант для всех шагов построения.)

e. Постройте F_7 и получите два решения уравнения $5x + 7y = 1$.

Раздел 2.2.2

1. Последовательность Фибоначчи определяется равенствами $f_0 = 1$, $f_1 = 1$, $f_{n+2} = f_{n+1} + f_n$, $n \geq 0$. Сколько операций нужно выполнить, чтобы вычислить $\gcd(f_{n+1}, f_n)$? Чему равен $\gcd(f_{n+1}, f_n)$?

2. (Теорема Лукаса.) Покажите, что $\gcd(f_m, f_n) = f_{\gcd(m,n)}$. (*Указание.* Примените алгоритм Евклида к m и n и используйте следующее соотношение (которое доказывается по индукции): $f_{m-1}f_n + f_m f_{n+1} = f_{m+n}$ и $f_n \mid f_{kn}$.)

3. **a.** Докажите, что для остатков, получающихся при выполнении алгоритма Евклида, выполняется неравенство

$$a_{k+2} < (1/2)a_k, \quad k > 1.$$

Другими словами, остатки не просто убывают, но убывают достаточно быстро. (*Указание.* Рассмотрите два возможных неравенства, связывающих a_{k+1} и a_k.)

b. Вместо теоремы Ламе воспользуйтесь результатом п. (a) для получения функции времени вычислений алгоритма Евклида **EA**.

4. Для каждой из следующих пар целых чисел найдите наибольший общий делитель, пользуясь алгоритмом Евклида:
(a) 34,21; (b) 136,51; (c) 481,325; (d) 8771,3206.

5. Часто алгоритм Евклида удается немного ускорить, если разрешить выполнять деление с отрицательными остатками, т.е. $a_k = a_{k+1}q_{k+1} - a_{k+2}$ наравне с $a_k = a_{k+1}q_{k+1} + a_{k+2}$, и выбирать наименьшее a_{k+2}. Таким образом, мы всегда получаем $a_{k+2} < (1/2)a_{k+1}$. Выполните четыре примера упр. 4, пользуясь этим методом.

6. Рассмотрим следующий рекурсивный алгоритм вычисления $d = \gcd(a, b)$, где a и b положительны. (Мы можем считать $a > 0$, $b > 0$ без потери общности, поскольку, как нам известно, $\gcd(a, b) = \gcd(|a|, |b|)$.)

[1] Если $a = b$, то положить $d := a$ и закончить работу.

[2] Если a и b оба четные, то положить $d := 2 \cdot \gcd(a/2, b/2)$.

[3] Если одно из двух чисел, например a, четно, то положить $d := \gcd(a/2, b)$.

[4] Если оба числа нечетны и не равны, например, $a > b$, то положить $d := \gcd(a - b, b)$.

Чему равно время вычислений этого алгоритма?

7. Разработайте алгоритм Евклида, который находит наибольший общий делитель двух гауссовых целых чисел. Нужно принять во внимание следующие факты. *Гауссовы целые числа* — это комплексные числа, вещественная и мнимая части которых являются целыми числами. Если α и β — два целых гауссовых числа, то мы говорим, что $\alpha|\beta$, если существует гауссово число γ такое, что $\beta = \alpha\gamma$. (Напомним, что поскольку $(a+bi)(a-bi) = a^2+b^2$ — вещественное число, мы можем выполнить деление, написав $(c+di)/(a+bi) = (c+di)(a-bi)/(a^2+b^2)$.) Мы определим $\gcd(\alpha,\beta)$ как гауссово число δ с максимальным абсолютным значением, которое делит как α, так и β. (Напомним, что абсолютное значение $|\delta|$ — это его расстояние до 0, т.е. квадратный корень из суммы квадратов вещественной и мнимой частей.) Отметим также, что gcd определен неоднозначно, поскольку мы всегда можем умножить его на ± 1 или $\pm i$ и получить другое δ с тем же самым абсолютным значением, которое также делит и α, и β. Любое из этих четырех возможных значений рассматривается как gcd. Наконец, заметим, что любое комплексное число может быть записано в виде суммы гауссова целого числа и комплексного числа, вещественная и мнимая части которого расположены каждая между $-1/2$ и $1/2$, вследствие чего мы можем делить одно гауссово число α на другое β и получать в частном гауссово число, а остаток будет меньше, чем β, по абсолютной величине.

Используйте полученный алгоритм для вычисления $\gcd(12277, 399+20i)$. (*См. также упр. 4 разд. 3.2.1.*)

8. Пусть f_n и f_{n+1} — последовательные члены последовательности Фибоначчи. Покажите, что $\gcd(f_{n+1}, f_n) = 1$.

9. Функция gcd является бинарной операцией на $\mathbb{Z}$; покажите, что она коммутативна и ассоциативна.

10. Покажите, что если $d = \gcd(m,n)$ и $a > 1$, то $\gcd(a^m-1, a^n-1) = a^d-1$. (*Указание.* Примените алгоритм Евклида, чтобы найти $\gcd(a^m-1, a^n-1)$ и чтобы найти $\gcd(m,n)$. Например, пусть a^r-1 — остаток, полученный после первого деления a^m-1 на a^n-1, в предположении, что $m > n$. Как связано r с m и n?)

11. Пользуясь тем, что $H_\infty^{(2)} = \sum_{d>1} 1/d^2 = \varpi^2/6$, $H_\infty^{(4)} = \varpi^4/90$, $H_\infty^{(6)} = \varpi^6/945$ и $H_\infty^{(8)} = \varpi^8/9450$, вычислите вероятность, с которой два, четыре, шесть и восемь (соответственно) случайно выбранных чисел взаимно просты. (См. также упр. 1 по

программированию разд. 2.2.2; также представляет интерес статья: Stark E.I., The Series $\sum_{d>1} k^{-s}$, $s = 2, 3, 4, \ldots$, Once More, Mathematics Magazine 47, 1974, 197–202.)

Раздел 2.2.3

1. Примените **XEA** к паре 217, 413.

2. Найдите целые числа x, y, такие, что

a. $1 = 3x + 5y$.

b. $1 = 12x + 21y$.

3. Пусть $d = ax + by$, где a, b, x, y, d — целые числа; верно ли, что $d = \gcd(a, b)$? Существует ли какая-либо взаимосвязь между d и $\gcd(a, b)$?

4. Найдите *все* решения приведенных выше упр. 2a и 2b.

5. Для каждого из следующих уравнений найдите целочисленное решение или докажите, что такового не существует: **(a)** $3x + 2y = 5$, **(b)** $2x + 6y = 7$.

6. Почему алгоритм, описанный в упр. 6 разд. 2.2.2, не всегда предпочтителен по отношению к алгоритму Евклида? (*Указание.* Можно ли выразить d как целую комбинацию a и b?)

7. Разработайте расширенный алгоритм Евклида для гауссовых целых чисел и примените его к $12277, 399 + 20i$.

Раздел 2.2.4

1. Завершите доказательства теорем 2.2.12 и 2.2.13.

2. Завершите пример, следующий за теоремой 2.2.15.

3. Покажите, что разложение числа $\sqrt{5}$ в цепную дробь имеет вид $\sqrt{5} = (2;\ 4,\ 4, \ldots)$. [*Указание.* Заметим, что в нашем случае $\xi = 2 + 1/[4 + (\xi - 2)]$.]

4. **a.** Разложите рациональные дроби $14/3$ и $3/14$ в конечные цепные дроби.

b. Преобразуйте в рациональные числа $(2; 1, 4)$ и $(0; 1, 1, 100)$.

5. Даны положительные целые числа i_1, i_2 и i_3, причем $i_2 > i_3$. Покажите, что для любого целого i

$$(i; i_2) < (i; i_3), \quad \text{но} \quad (i; i_1, i_2) > (i; i_1, i_3).$$

6. Пусть $i_1, i_2, \ldots, i_n$ и j — положительные вещественные числа. Докажите, что неравенство

$$(i_0; i_1, i_2, \ldots, i_n) > (i_0; i_1, i_2, \ldots, i_n + j)$$

истинно, если n нечетно, и ложно, если n четно.

7. Пусть $i_0, i_1, i_2, \ldots, i_n$ и $j_0, j_1, \ldots, j_{n+1}$ — положительные целые числа. При каких условиях истинно неравенство

$$(i_0; i_1, i_2, \ldots, i_n) > (j_0; j_1, \ldots, j_n + j)?$$

8. Найдите квадратичные иррациональные числа, соответствующие бесконечным цепным дробям

a. $(1; 1, 1, 1, \ldots)$.

b. $(2; 1, 1, 1, \ldots)$.

c. $(2; 3, 1, 1, \ldots)$.

d. $(2; 2, 2, 2, \ldots)$.

9. Докажите, что $q_n/q_{n-1} = (c_n; c_{n-1}, \ldots, c_1)$ для $n \geq 1$. Найдите и докажите аналогичное разложение в цепную дробь для p_n/p_{n-1} в предположении, что $c_0 \geq 0$.

10. Разложите каждое из следующих чисел в бесконечную цепную дробь: $\sqrt{2}$, $\sqrt{2} - 1$, $\sqrt{2}/2$, $\sqrt{3}$.

11. Дано, что для двух иррациональных чисел подходящие дроби $p_0/q_0, p_1/q_1, \ldots$ совпадают по p_k/q_k включительно. Докажите, что их разложения в цепные дроби совпадают по c_k включительно.

12. Докажите, что

$$q_n|q_{n-1}\xi - p_{n-1}| + q_{n-1}|q_n\xi - p_n| = 1.$$

Раздел 2.3.1

1. Покажите, что для любого n существует простое p, такое, что $n < p \leq n! + 1$.

2. Докажите, что для проверки простоты n достаточно показать, что никакое $p \leq \sqrt{n}$ не делит n.

3. Пусть $p_1 = 2$, $p_2 = 3$, $\ldots$ — простые числа в порядке возрастания. Покажите, что

a. $p_{n+1} \leq 2p_n + 1$. [*Указание.* Используйте **постулат Бертрана**, который утверждает, что для любого $n > 1$ существует простое p, такое, что $n \leq p < 2n$; его доказательство можно найти в (Niven, Zuckerman, 1980).]

b. $p_{n+1} \leq p_1 + p_2 + \cdots + p_n$ для любого $n > 1$.

4. Покажите, что квадратный корень из простого числа никогда не является рациональным числом, т.е. $\sqrt{p} \neq r/s$.

5. Для каких значений n функция Эйлера $\phi(n)$ нечетна?
6. Если $\phi(n) = 2$, то что можно сказать о числе n?
7. Докажите, что для любой степени простого числа $\phi(p^\alpha) = p^\alpha - p^{\alpha-1} = p^\alpha(1 - 1/p)$.
8. Найдите $\phi(n)$ для всех n от 70 до 80.
9. Составьте список, содержащий все целые числа n, для которых $\phi(n) \leq 10$, и докажите, что ваш список полон.
10. Предположим, что $n = p \cdot q$, где p и q – два различных простых числа. Докажите, что знание двух простых чисел p, q эквивалентно знанию n и $\phi(n)$ в том смысле, что мы можем вычислить n и $\phi(n)$ по p и q и можем вычислить p и q по n и $\phi(n)$, используя только «малое» число арифметических операций и операцию извлечения квадратного корня из целого числа. Этот факт подчеркивался также в разделе 4.2.2.

Упр. 11–15 имеют дело с простыми числами в кольце гауссовых чисел $\mathbb{Z}[i]$. *(См. также упр. 7 разд. 2.2.2.)*

11. Докажите, что если z — гауссово целое число (т.е. $z \in \mathbb{Z}[i]$) и его норма $n(z) = p$ — простое число в $\mathbb{Z}$, то z — простое в $\mathbb{Z}[i]$. {Напомним, что если $z = x + y$, то $n(z) = x^2 + y^2 [= (x+iy)(x-iy)]$.} Проверьте, что $1+i$, $1+2i$ и $2-3i$ — простые числа в $\mathbb{Z}[i]$.
12. Покажите, что если z — простое число в $\mathbb{Z}[i]$, то существует простое число p в $\mathbb{Z}$, такое, что $z|p$ в $\mathbb{Z}[i]$.
13. Докажите, что если p — простое число в $\mathbb{Z}$, то либо p — простое в $\mathbb{Z}[i]$, либо $p = u \cdot v$, где u и v — сопряженные простые числа в $\mathbb{Z}[i]$. (Следовательно, разлагается ли p на множители в $\mathbb{Z}[i]$, зависит от того, может ли p быть записано в виде суммы двух квадратов целых чисел. Следующие два упражнения помогают нам определить их.)
14. Покажите, что если p — простое число в $\mathbb{Z}$ и $p \equiv 3 \pmod 4$, то p просто в $\mathbb{Z}[i]$.
15. (Ферма) Покажите, что если p — простое число в $\mathbb{Z}$ и $p \equiv 1 \pmod 4$, то $p = x^2 + y^2$, x, y — целые числа, т.е. p не является простым в $\mathbb{Z}[i]$. (*Указание.* Воспользуйтесь теоремой Вильсона из следующего раздела.)

См. также упр. 5(b) в упражнениях для разд. 2.3.4.

Раздел 2.3.2

1. Завершите доказательство теоремы 2.3.7.
2. Завершите доказательство теоремы 2.3.8.
3. Покажите, что если 5 не делит n, то 5 делит $n^8 - 1$.

4. Найдите $2^{47} \pmod{23}$; используйте теорему Ферма.

5. Покажите, что для любого n число $n^7 - n$ делится на 2, 3, 6 и 7.

6. Используя два способа, которые обсуждались в тексте, вычислите мультипликативные обратные к числам 7, 8 и 9 по модулю 19. [*Указание.* Используйте алгоритм **E**.]

7. Покажите, что мультипликативный обратный к $(m-1)$ в кольце $\mathbb{Z}_m$ равен $m-1$.

8. Пусть G — группа и a — ее элемент порядка s; покажите, что порядок элемента a^r равен $s/(r,s)$.

9. Найдите все примитивные корни по модулю 27.

10. Используйте теорему 2.2.4 для доказательства того, что система сравнений $x \equiv a \pmod{m_1}$ и $x \equiv b \pmod{m_2}$ имеет решение тогда и только тогда, когда $(m_1, m_2) \mid (b-a)$. Если решение существует, то оно единственно по модулю $[m_1, m_2]$, наименьшего общего кратного чисел m_1 и m_2. (Обобщите это утверждение на случай произвольного числа уравнений. Для доказательства единственности покажите, что $x \equiv x_0 \pmod{m_1}$ и $x \equiv x_0 \pmod{m_2}$ тогда и только тогда, когда $x \equiv x_0 \pmod{[m_1, m_2]}$.)

11. Докажите теорему 2.3.25.

12. (Никомах; см. также историческое замечание 6.) Решите следующую систему уравнений:

$$x \equiv 2 \pmod{3},$$
$$x \equiv 3 \pmod{5},$$
$$x \equiv 2 \pmod{7}.$$

13. **a.** Покажите, что существует бесконечно много чисел, свободных от квадратов.

b. Покажите, что всякое положительное целое число представляется единственным образом в виде произведения квадрата и числа, свободного от квадратов.

14. Докажите, что десятичное целое число делится на 3 тогда и только тогда, когда сумма его цифр делится на 3, и что оно делится на 9 тогда и только тогда, когда сумма его цифр делится на 9.

15. **a.** Пусть $m = (p_1)^{e_1} \cdot (p_2)^{e_2} \ldots (p_k)^{e_k}$. Докажите, что любое решение системы сравнений $x \equiv a \pmod{p_i^{e_i}}$, $i = 1, 2, \ldots, k$, является также решением сравнения $x \equiv a \pmod{m}$ и обратно. (Обобщите этот результат на сравнения высших степеней и покажите, что если $S(m)$ обозначает число решений сравнения $f(x) \equiv 0 \pmod{m}$, где m — данное выше число, то $S(m) = \prod_{1 \le i \le k} S(p_i^{e_i})$.)

b. Покажите, что $x \equiv a \pmod{m_i}$ для $i = 1, 2, \ldots, k$ тогда и только тогда, когда $x \equiv a \pmod{[m_1, m_2, \ldots, m_k]}$, где, как вы помните, $[\ldots]$ обозначает наименьшее общее кратное чисел, заключенных в скобки.

16. **a.** Сколько решений имеет сравнение

$$x^2 \equiv 1 \pmod{m},$$

если два решения x и x' считаются совпадающими, если $x \equiv x'$? (*Указание.* Сначала докажите, что если m имеет вид p^α или $2 \cdot p^\alpha$, $\alpha > 0$ и $p > 2$ для обоих случаев, то сравнение $x^2 \equiv 1 \pmod{m}$ имеет два решения $x \equiv 1 \pmod{m}$ и $x \equiv - \pmod{m}$. Затем докажите, что если m имеет вид 2^k, $k \geq 3$, то существуют четыре решения, $x \equiv \pm 1 \pmod{m}$ и $x \equiv 2^{k-1} \pm 1 \pmod{m}$. Что случится при $k = 1$ и $k = 2$? Наконец, воспользуйтесь тем, что $x^2 \equiv 1 \pmod{m}$ тогда и только тогда, когда $x^2 \equiv 1 \pmod{p_i^{e_i}}$ для всех простых чисел p_i, у которых $e_i > 0$ в полном разложении m на множители, и скомбинируйте полученные результаты, чтобы показать, что если m имеет ровно r различных простых делителей, то общее число решений равно 2^r, с корректировкой для четных m. То есть в общем случае точное число равно $2^{r+(8|m)+(4|n)-(2|m)}$, где $(k|m)$ равно 1, если $k|m$, и 0 в противном случае.)

b. С помощью рассуждений, аналогичных предложенным выше, докажите, что число решений сравнения $x^2 \equiv x \pmod{m}$ для любого целого m равно 2^r, если m имеет ровно r различных простых делителей.

c. Сколько 3-значных чисел удовлетворяет сравнению $x^2 \equiv x \pmod{1000}$? Обобщите свой ответ.

17. Покажите, что сравнение $x^2 \equiv -1 \pmod{p}$, где p — простое число, имеет решения тогда и только тогда, когда $p = 2$ или $p \equiv 1 \pmod{4}$. (*Указание.* Воспользуйтесь теоремой Вильсона и тем, что если $p \neq 2$ и p не сравнимо с 1 по модулю 4, то $p \equiv 3 \pmod{4}$.)

18. **a.** Докажите, что если p — простое число и $\gcd(a, p) = 1$, то сравнение $x^n \equiv a \pmod{p}$ имеет $\gcd(n, p-1)$ решений или не имеет решений в зависимости от того,

$a^{(p-1)/\gcd(n,p-1)} \equiv 1 \pmod p$ или $a^{(p-1)/\gcd(n,p-1)} \not\equiv 1 \pmod p$. (*Указание.* Воспользуйтесь малой теоремой Ферма, чтобы показать, что сравнение $x^n \equiv a \pmod p$ не имеет решений, если $a^{(p-1)/\gcd(n,p-1)} \not\equiv 1 \pmod p$. Для доказательства обратного утверждения воспользуйтесь следующей информацией: должен существовать образующий g по модулю p такой, что $g^j \equiv a \pmod p$, а значит, исходная проблема может быть сведена к сравнению $yn \equiv j \pmod{p-1}$; затем воспользуйтесь тем, что линейное сравнение общего вида $ax \equiv b \pmod m$ имеет $\gcd(a,m)$ решений по модулю m (теорема 2.3.22).)

b. *Критерий Эйлера.* Используйте (a), чтобы показать, что если p — нечетное простое число и $\gcd(a,p) = 1$, то сравнение $x^2 \equiv a \pmod p$ имеет два решения или не имеет ни одного в зависимости от того, $a^{(p-1)/2} \equiv 1 \pmod p$ или $a^{(p-1)/2} \equiv -1 \pmod p$. (Величина $a^{(p-1)/2} \pmod p$ играет важную роль в теории чисел; см. упр. 9 разд. 2.3.3, утверждающее, что ее значение равно значению символа Лежандра $\left(\frac{a}{p}\right)$, по определению равного 1, если a — квадратичный вычет по модулю p, и равного -1, если a — квадратичный невычет по модулю p. Существует также символ Якоби: для $q = q_1 q_2 \ldots q_s$, т.е. когда q равно произведению s не обязательно различных простых чисел, он определяется как $\left(\frac{a}{q}\right) = \prod_{1 \le j \le s} \left(\frac{a}{q_j}\right)$, где $\left(\frac{a}{q_j}\right)$ — символ Лежандра. Ясно, что $\left(\frac{a}{q}\right) = \pm 1$, но *неверно*, что из равенства $\left(\frac{a}{q}\right) = 1$ следует, что a является квадратичным вычетом по модулю q, например $\left(\frac{2}{9}\right) = 1$, но сравнение $x^2 \equiv 2 \pmod 9$ не имеет решений.)

19. Найдите наименьшее положительное число, дающее остаток 2 при делении на 11, остаток 3 при делении на 12 и остаток 4 при делении на 13.

20. Расширьте алгоритм возведения в степень **E** так, чтобы он вычислял $b^n \pmod m$ в общем случае (т.е. для любых значений n), и воспользуйтесь этим расширенным алгоритмом, чтобы вычислить $75^{38} \pmod{107}$.

21. Как указано в тексте, если $\gcd(a,p) = 1$, то a^{p-2} является обратным к a по модулю p. Предположим, что p — очень большое число. Скомбинируйте бинарный алгоритм возведения в степень с расширенным алгоритмом Евклида, чтобы получить эффективный метод вычисления $a^{-1} \pmod p$, когда

(a) a имеет столько же цифр, сколько и p, и (b) когда a гораздо меньше p.

22. Предположим, что 3-значное (десятичное) положительное целое число, дающее остаток 2 при делении на 11, остаток 9 при делении на 12 и остаток 11 при делении на 13, является делителем шестизначного натурального числа, дающего остаток 5 при делении на 11, 9 при делении на 12 и 8 при делении на 13. Найдите частное. (*Указание.* Имеется два возможных ответа.)

23. (Обращение теоремы Ферма.) Покажите, что если $a^{m-1} \equiv 1 \pmod{m}$ и $a^{(m-1)/p}$ не сравнимо с 1 по модулю m для всех простых p, таких, что $p|(m-1)$, то m — простое число. (*Указание.* Покажите, что если это условие выполнено, то числа $a^k \pmod{m}$ различны для $1 \leq k < m$.)

24. **a.** Пользуясь малой теоремой Ферма, докажите, что если a не делится на простое число m и $n \equiv k \pmod{m-1}$, то $a^n \equiv a^k \pmod{m}$.

b. Используйте (a), чтобы вычислить последнюю цифру по основанию 7 числа 3^{10000}.

25. Пользуясь греко-китайской теоремой об остатках, покажите, что ϕ-функция Эйлера мультипликативна, т.е. покажите, что $\phi(m \cdot n) = \phi(m) \cdot \phi(n)$, если только $\gcd(m, n) = 1$.

26. Пользуясь теоремой Эйлера (теорема 2.3.14), докажите, что если $\gcd(a, m) = 1$, то $a^n \equiv a^{n'} \pmod{m}$ тогда и только тогда, когда $n \equiv n' \pmod{\phi(m)}$. (Этот результат будет использован в разд. 4.2.2.)

27. **a.** Покажите, что в теореме Эйлера (теорема 2.3.14), если m — не простое число и не четное число вида $2p$, где p — простое число, то существует меньшая (чем $\phi(m)$) степень числа a, гарантированно дающая 1 по модулю m, а именно наименьшее общее кратное степеней, дающих $1 \mod p^\alpha$, α — наибольшее целое, такое, что $p^\alpha | m$. Например, пусть $m = 105$, где $105 = 3 \cdot 5 \cdot 7$; в этом случае $\phi(105) = 48$ и $a^{48} \equiv 1 \pmod{105}$ для некоторого a; заметим, что также $a^{12} \equiv 1 \pmod{105}$, потому что 12 — это наименьшее общее кратное чисел $3-1$, $5-1$ и $7-1$.

b. Пользуясь (a), вычислите $2^{10000} \mod 77$. Получите то же самое решение, пользуясь греко-китайской теоремой об остатках.

28. Найдите 3-значное (десятичное) число (две возможности), дающее остаток 4 при делении на 5, 7 или 13.

29. Рассмотрим систему сравнений

$$x \equiv a_1 \pmod{m_1},$$
$$x \equiv a_2 \pmod{m_2},$$

где m_1 и m_2 не являются взаимно простыми. Приведите пример чисел a_1 и a_2, для которых нет решения. Для каких a_1 и a_2 эта система имеет решение?

Раздел 2.3.3

1. Покажите, что если m — нечетное составное число, то имеет место следующее:

a. Если m делится на полный квадрат > 1, то m не является кармайкловым числом.

b. Если m свободно от квадратов, то m — кармайклово число тогда и только тогда, когда $(p-1)|(m-1)$ для каждого простого числа p, такого, что $p|m$.

(В качестве примера рассмотрите $m = 561 = 3 \cdot 11 \cdot 17$; 560 делится на $3-1$, $11-1$ и $17-1$.)

2. Используйте упр. 1, чтобы доказать, что кармайклово число должно быть произведением *не менее* трех различных простых чисел.

3. **a.** Найдите все основания b, по которым 15 — псевдопростое число (не включайте тривиальные основания ± 1.)

b. Покажите, что если и p, и $2p-1$ — простые числа и $n = p(2p-1)$, то n — псевдопростое число для 50% возможных оснований b, т.е. для всех b, которые являются квадратичными вычетами по модулю $2p-1$.

4. Пусть m — положительное нечетное составное число, и пусть $\gcd(b, m) = 1$.

a. Покажите, что если p — простое число, $p|m$ и m псевдопросто по основанию b, то $b^{m'-1} \equiv 1 \pmod{p}$, где $m' = m/p$.

b. Покажите, что никакое целое число вида $m = 3p$ ($p > 3$ — простое число) не может быть псевдопростым по основанию 2, 5 или 7.

c. Покажите, что никакое целое число вида $m = 5p$ ($p > 5$ — простое число) не может быть псевдопростым по основанию 2, 3 или 7.

d. Покажите, что 91 — наименьшее псевдопростое число по основанию 3.

5. Чему равно наименьшее псевдопростое число по основанию 2? по основанию 5?

6. Докажите, что если p — простое число, то p^2 псевдопросто по основанию b тогда и только тогда, когда $b^{p-1} \equiv 1 \pmod{p^2}$.

7. Пусть $m = pq$ — произведение двух различных простых чисел. Положим $d = \gcd(p-1, q-1)$. Докажите, что m псевдопросто по основанию b тогда и только тогда, когда $b^d \equiv 1 \pmod{m}$. Сколько, в терминах d, существует оснований, относительно которых m псевдопросто? Сколько существует оснований, относительно которых m псевдопросто, если $q = 2p+1$? Перечислите все такие основания в терминах p. (См. также упр. 19 ниже.)

8. Покажите, что если m псевдопросто по основанию b и если $\gcd(b-1, m) = 1$, то целое число $(b^m - 1)/(b-1)$ псевдопросто по основанию b. Приведите пример, показывающий, что это может быть не так, если мы опустим условие $\gcd(b-1, m) = 1$.

9. **a.** Воспользуйтесь малой теоремой Ферма, чтобы показать, что если m — нечетное простое число и $\gcd(b, m) = 1$, то

$$b^{(m-1)/2} \equiv \left(\frac{b}{m}\right) \pmod{m}. \qquad (*)$$

(См. определение символа $\left(\frac{b}{m}\right)$ в упр. 18 разд. 2.3.2.)

Мы покажем, что если m — нечетное составное число, то сравнение $(*)$ ложно по крайней мере для 50% всех b, для которых $\gcd(b, m) = 1$.

b. Покажите, что если сравнение $(*)$ истинно для b_1 и ложно для b_2, то оно ложно для произведения $b_1 b_2$. Используйте этот результат, чтобы доказать, что если сравнение $(*)$ ложно хотя бы для одного b, то число значений b, для которых оно ложно, по крайней мере так же велико, как число значений b, для которых оно истинно.

c. Если m делится на квадрат простого числа p, покажите, как найти целое число b, $\gcd(b, m) = 1$, такое, что $b^{(m-1)/2} \not\equiv \pm 1 \pmod{m}$.

d. Покажите, что если m — произведение различных простых чисел, p — одно из этих простых чисел и b таково, что $\left(\frac{b}{p}\right) = -1$ и $b \equiv 1 \pmod{m/p}$, то сравнение $(*)$ для b ложно. Затем покажите, что такое b всегда существует.

10. Если m — нечетное составное число и b — целое число такое, что $\gcd(b, m) = 1$ и имеет место сравнение $(*)$, то m называется *псевдопростым числом Эйлера по основанию* b.

Если m — псевдопростое число Эйлера по основанию b, то оно псевдопросто по основанию b. (Отметим, что обратное утверждение неверно; например, 91 псевдопросто по основанию 3, но $3^{45} \equiv 27 \pmod{91}$.)

11. Предположим, что n — положительное целое число, такое, что $6n+1$, $12n+1$ и $18n+1$ — простые числа. Докажите, что $m = (6n+1)(12n+1)(18n+1)$ — кармайклово число. (Существует также много кармайкловых чисел *не* такого вида.)

12. Проверьте, что все следующие числа кармайкловы: $1105 = 5 \cdot 13 \cdot 17$; $1729 = 7 \cdot 13 \cdot 19$; $2465 = 5 \cdot 17 \cdot 29$; $2821 = 7 \cdot 13 \cdot 31$; $6601 = 7 \cdot 23 \cdot 41$; $29341 = 13 \cdot 37 \cdot 61$; $172081 = 7 \cdot 13 \cdot 31 \cdot 61$; $278545 = 5 \cdot 17 \cdot 29 \cdot 113$.

13. **a.** Покажите, что для любого фиксированного простого числа r существует только конечное число кармайкловых чисел вида rpq (p, q — простые числа).

b. Найдите все кармайкловы числа вида $3pq$ (p, q — простые числа).

c. Найдите все кармайкловы числа вида $5pq$ (p, q — простые числа).

14. Покажите, что 561 — наименьшее кармайклово число.

15. Рассмотрим 561, наименьшее кармайклово число.

a. Найдите число оснований $b \in (\mathbb{Z}/561\mathbb{Z})^*$ (обратимые по модулю 561 элементы), для которых 561 — псевдопростое число Эйлера.

b. Найдите и перечислите основания, по которым 561 строго псевдопросто.

16. Покажите, что если $m = p^\alpha$, где $\alpha > 1$ и p — простое число, то m строго псевдопросто по основанию b тогда и только тогда, когда оно псевдопросто по основанию b.

17. Проверьте, что 65 строго псевдопросто по основанию 8 и по основанию 18, но не по основанию 14, которое равно произведению 8 и 18 по модулю 65.

18. Покажите, что если $m \equiv 3 \pmod{4}$, то m строго псевдопросто по основанию b тогда и только тогда, когда оно — псевдопростое число Эйлера по основанию b.

19. Покажите, что если мы найдем основание b, такое, что m псевдопросто, но *не* строго псевдопросто по основанию b, то мы можем быстро найти нетривиальный сомножитель числа

m. Объясните, как бороться с этим при выборе $m = pq$ в RSA-криптосистеме, обсуждаемой в разд. 4.2.2; см. также выше упр. 7.

Раздел 2.3.4

1. **a.** Докажите, что для любого целого числа b и любого положительного целого числа n число $b^n - 1$ делится на $b - 1$ и частное равно $b^{n-1} + b^{n-2} + \cdots + b^2 + b + 1$.

b. Заменив b на b^m, мы видим, что $b^{mn} - 1$ делится на $b^m - 1$ с соответствующим образом измененным частным. Чему равны два легко вычисляемых сомножителя числа $2^{21} - 1$?

2. Докажите, что если n нечетно, то $b^n + 1$ делится на $b + 1$ и частное равно $b^{n-1} - b^{n-2} + \cdots + b^2 - b + 1$.

3. **a.** Докажите, что если $2^n - 1$ — простое число (Мерсенна), то n — простое число, и что если $2^n + 1$ — простое число (Ферма), то n — степень числа 2. (*Указание.* Докажите это методом от противного, используя упр. 1 и 2.)

b. Используйте малую теорему Ферма (и несколько десятков умножений), чтобы доказать, что $2^{32}+1$ — *не* простое число.

4. Предположим, что $\gcd(b, n) = 1$ и что a и c — положительные целые числа. Докажите, что если $b^a \equiv 1 \pmod{n}$ и $b^c \equiv 1 \pmod{n}$, то $b^d \equiv 1 \pmod{n}$, где $d = \gcd(a, c)$.

5. Используйте упр. 4, чтобы доказать, что если p — простое число, делящее $b^n - 1$, то либо (i) $p|(b^d - 1)$ для некоторого *собственного* делителя d числа n, либо (ii) $p \equiv 1 \pmod{n}$. Если $p > 2$ и n нечетно, то в случае (ii) мы имеем $p \equiv 1 \pmod{2n}$.

Приведенные выше упражнения используются далее в упр. 6–9:

6. **a.** Разложите на множители $2^{15} - 1$ и $2^{30} - 1$.

b. Разложите на множители $2^{33} - 1$ и $2^{21} - 1$.

7. Разложите на множители $3^{15} - 1$ и $3^{24} - 1$.

8. Разложите на множители $5^{12} - 1$.

9. Разложите на множители $10^6 - 1$ и $10^8 - 1$.

Разложение на множители методом Ферма (обоснование метода, описанного в тексте).

10. **a.** Покажите, что если m — положительное нечетное число, то существует взаимно однозначное соответствие между разложениями m на множители вида $m = pq$, $p \geq q > 0$,

и представлениями m в виде $m = u^2 - v^2$, где u, v — неотрицательные целые числа. Это соответствие задается уравнениями

$$u = (p+q)/2, \quad v = (p-q)/2; \quad p = u+v, \quad q = u-v.$$

b. Используя (a), мы можем разложить m на множители, вычисляя для $k = 1, 2, \ldots$, числа $a_k = \lceil\sqrt{m}\rceil + k$ и проверяя, является ли $u^2 - n = v^2$ полным квадратом (где, разумеется $u := a_k$).

c. Пользуясь описанной в (b) процедурой, разложите на множители 91; 323.

Метод цепных дробей; этот метод отличается от обсуждаемого в тексте только тем, что величины a_k выбираются с применением цепных дробей (также здесь используются системы множителей). Он основан на двух следующих упражнениях.

11. Покажите, что если $x > 1$ — вещественное число, разложение которого в цепную дробь имеет подходящие дроби p_i/q_i, то $|p_i^2 - x^2 q_i^2| < 2x$ для всех i.

12. **a.** Покажите, что если m — положительное целое число, не являющееся полным квадратом, и p_i/q_i — подходящие дроби в разложении $\sqrt{m}$ в цепную дробь, то наименьший по абсолютной величине вычет $p^2 \pmod m$ меньше, чем $2\sqrt{m}$. (*Указание.* Воспользуйтесь упр. 11.)

b. Часть (a) является ключевой в методе цепных дробей; она утверждает, что, перебирая *числители* подходящих дробей в разложении $\sqrt{m}$ в цепную дробь, мы можем найти последовательность чисел p_i, квадраты которых имеют малые вычеты. (Отметим, что нам нужны только числители p_i, т.е. нам не нужно находить точное значение подходящей дроби.) Процедура разложения на множители аналогична процедуре, описанной в тексте, и детали мы оставляем читателю (просмотрите еще раз разд. 2.2.4).

c. Пользуясь методом цепных дробей, разложите на множители 899 и 1443.

Раздел 2.4

1. Докажите теорему 2.4.1.

2. Пусть d — наибольший общий делитель целых чисел m и n. Покажите, что полином $x^d - 1$ является наибольшим общим делителем полиномов $x^m - 1$ и $x^n - 1$ над любым полем.

3. Используя многомодульную арифметику с вектором оснований $\beta = [3, 5, 7]$, вычислите $a + b$, $a - b$, $a \cdot b$ и $b^{-1} \pmod{\beta}$, где $a = 19$ и $b = 23$.

4. Используя тот же вектор оснований $\beta = [3, 5, 7]$, найдите многомодульное представление следующих целых чисел как для симметричной системы остатков, так и для системы наименьших неотрицательных остатков: **(a)** 127; **(b)** -127; **(c)** 537; **(d)** -537.

5. Пусть $\beta = [7, 5, 3, 2]$ — вектор оснований. Каков знак числа $x = [6, 3, 1, 1]$?

Упражнения по программированию

Раздел 2.2.1

1. Свойство евклидовости утверждает, что для заданных a и ненулевого b существует единственное частное q и остаток r, такие, что $a = bq + r$, $0 \leq r < |b|$.

a. Напишите процедуру **QUO**, которая возвращает частное q.

b. Напишите процедуру **MOD**, которая возвращает наименьший неотрицательный остаток r.

c. Напишите процедуру **LAVMOD**, которая возвращает минимальный по абсолютной величине остаток r, такой, что $-|b|/2 < r \leq |b|/2$.

Раздел 2.2.2

1. Используя **EA**, напишите процедуру **GCDLCM**, которая для заданных a и b возвращает $d = \gcd(a, b)$ и $m = \operatorname{lcm}(a, b)$. Эта процедура должна возвращать значения $(0, a) = (a, 0) = |a|$ и $[0, a] = [a, 0] = 0$ для любого a. Выполните эту процедуру для

следующих входных данных:

a	b
18755	6727
4199	407
69	9453
1463	14098

Если возникнут трудности с переполнением (т.е. если максимальное представимое целое число равно $2^{15}-1=32767$), то выразите наименьшее общее кратное в виде $[a/(a,b)]b$ (например, $[21,35]=3\cdot 35$).

2. Выберите 500 пар случайных чисел в интервале от 1 до 500. Для каждой пары вычислите $\gcd(a,b)$, используя процедуру из предыдущего упражнения. Найдите в процентах долю пар, состоящих из взаимно простых чисел. Сравните полученное число (процентов) с $100\cdot(6/\varpi^2)$, т.е. определите, насколько эта доля близка к 0.60793 (Дирихле).

Раздел 2.2.3

1. Напишите подпрограмму **GLE** (линейное уравнение общего вида (General Linear Equation)), которая по заданным a, b, c, таким, что a и b не равны нулю одновременно, возвращает x, y и t, где

$$t=\begin{cases}0, & \text{если } ax+by=c \text{ имеет решение,}\\ 1, & \text{если это уравнение не имеет решений,}\end{cases}$$

и, если $t=0$, то x, y — решение с наименьшим положительным x. Эта подпрограмма должна сначала использовать **XEA**, чтобы найти решение уравнения $ax+by=(a,b)$, а затем использовать теорему 2.2.4. Выполните подпрограмму для следующих входных данных:

$168x-66y=42$, $\qquad 343x+407y=7$,
$426x-156y=128$, $\qquad 1463x+4235y=11$.

Раздел 2.2.4

1 (Лагранж, 1798). Пусть $p(x)=c_nx^n+\cdots+c_0$, $c_n>0$ — полином с целыми коэффициентами, не имеющий рациональных корней и имеющий в точности один вещественный корень

$\xi > 1$. Напишите процедуру нахождения (вместе с любой добавочной информацией, которую вы хотите получить) неполных частных (pq) (от слов *p*artial *q*uotiens) числа ξ, используя следующий алгоритм с экспоненциальным временем работы (который включает в основном только операции сложения; для его понимания необходимо овладеть материалом гл. 7):

1. [Инициализация] Положить $(pq) := 1$.

2. [$x := 1+x$] Для $i = 0, 1, \ldots, n-1$ (в таком порядке) и для $j = n-1, \ldots, i$ (в таком порядке) вычислить $c_j = c_{j+1}+c_j$. [Этот шаг основан на методе Руффини–Горнера, который обсуждается в разд. 3.1.2; корни полинома $p_{ti}(1+x)$ на единицу меньше, чем корни полинома $p_{ti}(x)$, где $p_{ti}(x)$ получается из исходного полинома $p(x)$ после ряда преобразований.]

3. [Продолжить?] Если $c_n + c_{n-1} + \cdots + c_0 < 0$, то положить $pq := pq + 1$ и перейти к шагу 2.

4. [$x := 1/x$] Выдать pq, значение следующего частичного приближения, заменить коэффициенты $(c_n, c_{n-1}, \ldots, c_0)$ на $(-c_0, -c_1, \ldots, -c_n)$ и перейти к шагу 1. [Корни функции $p_{ti}(1/1)$ взаимны (reciprocals) с корнями полинома $p_{ti}(x)$.]

Попробуйте выполнить этот алгоритм для полинома $p(x) = x^3 - 2$ и аналогичных полиномов.

Раздел 2.3.1

1. Используя рассмотренный в тексте метод решета, постройте вектор всех простых чисел, не превосходящих 100.

2. Найдите 20 наибольших простых чисел $< 2^{15}$, используя приведенный в тексте метод.

Раздел 2.3.2

1. Напишите процедуру **LCE** (**L**inear **C**ongruence **E**quation), которая по заданным ненулевым a, b и $m > 1$ возвращает x, n и t, где

$$t = \begin{cases} 0, & \text{если } ax \equiv b \pmod m \text{ имеет решение,} \\ 1, & \text{если уравнение не имеет решений,} \end{cases}$$

и, если $t = 0$, то x — наименьшее неотрицательное решение по модулю $n = m/(a, m)$; см. теорему 2.3.22. Процедура должна

использовать **GLE** (разд. 2.2.3). Выполните эту программу для следующих входных данных:

$36x \equiv -20 \pmod{16}$, $22x \equiv 253 \pmod{143}$,
$57x \equiv 148 \pmod{38}$, $35x \equiv 9973 \pmod{12}$.

2. Используя **LCE**, напишите процедуру **MODINV**, которая по заданным $a \neq 0$ и $m > 1$ возвращает x и t, где

$$t = \begin{cases} 0, & \text{если } ax \equiv 1 \pmod{m} \text{ имеет решение,} \\ 1, & \text{если это уравнение не имеет решений,} \end{cases}$$

и, если $t = 0$, то x — наименьшее неотрицательное решение по модулю m. Выполните программу для следующих входных данных:

$32x \equiv 1 \pmod{35}$, $74x \equiv 1 \pmod{128}$,
$119x \equiv 1 \pmod{12}$, $486x \equiv 1 \pmod{1033}$.

3. **a.** Реализуйте **GRAk** и используйте его для решения следующих систем:

(i)	(ii)	(iii)
$x \equiv -28 \pmod{15}$,	$x \equiv 24 \pmod{13}$,	$x \equiv 6 \pmod{23}$,
$x \equiv 18 \pmod{22}$,	$x \equiv 52 \pmod{17}$,	$x \equiv -50 \pmod{12}$,
	$x \equiv -1 \pmod{19}$,	$x \equiv 27 \pmod{31}$.

b. Модифицируйте алгоритм так, чтобы он возвращал наименьший по абсолютной величине остаток, **LAVGCRA**, и выполните его для указанных выше входов.

Раздел 2.3.4

1. Опишем процедуру получения разложения числа $n > 1$ на простые множители. Используя метод решета, построим таблицу простых чисел $\leq \sqrt{n}$ и проверим, делится ли n на простые числа 2, 3, ..., $\leq \sqrt{n}$. Если делителей нет, то n просто (см. также упр. 2 разд. 2.3.1). Если какое-либо p делит n, то найдем наибольшую степень e числа p, которая делит n. Запомним p, e и для нового частного $q = n/p^e$ проверим, делится ли оно на последующие простые числа $\leq \sqrt{n}$. Если новых делителей

не будет найдено, то либо $q = 1$ и все простые делители уже найдены, либо $\sqrt{n} < q < n$ и q — еще один простой делитель (n может иметь не более одного простого делителя $> \sqrt{n}$). Напишите процедуру **FACT**, находящую разложение на простые множители для чисел $n \leq 2^{15} - 1 = 32767$. Вам потребуется таблица простых чисел $< \lfloor \sqrt{2^{15}} \rfloor = 181$. Используйте эту процедуру для разложения чисел 9583, 9973, 16384, 17017 и 29957.

Исторические замечания и литература

1. Наше представление теории множеств «наивно», потому что мы предполагаем, что любое внушающее доверие описание определяет множество. В начале нашего столетия Бертран Рассел удивил логиков и математиков, выведя несколько логических противоречий в наивной теории множеств. Рассмотрим множество $y = \{x : x \notin x\}$, т.е. множество, содержащее все множества, которые не содержат себя в качестве элемента. Для наших целей не важно знать, существует ли множество, которое содержит себя в качестве элемента, нас заботит, что y может быть множеством всех множеств. Возникает вопрос, верно ли, что $y \in y$? Если $y \in y$, то $y \notin y$ по определению y. С другой стороны, если $y \notin y$, то $y \in y$. Оба случая, как $y \in y$, так и $y \notin y$, ведут к противоречию. Это — один из парадоксов Рассела, и он аналогичен парадоксу цирюльника: в деревне живет цирюльник, который бреет только тех мужчин, которые не бреются сами. Кто бреет цирюльника? Бреет себя цирюльник или не бреет, оба случая ведут к противоречию. Усилия развить состоятельную теорию множеств привели математиков к построению аксиоматической теории множеств, которая указывает точные способы, как может быть построено множество. Однако если не забираться слишком глубоко и использовать теорию множеств преимущественно как инструмент, то годится наивная теория множеств.

2. Леонардо Пизанский, широко известный как Фибоначчи, был одинокой звездой первой величины на темном математическом небосклоне средневековья. Он жил около 1200 г., много путешествовал по Аравии и с помощью своей «Книги абака» принес в Европу индоарабскую числовую систему и другие достижения Востока.

3. Интересно отметить, что в 1896 г., почти через сто лет после того,

как были сформулированы гипотезы Гаусса и Лежандра, Адамар и Валле Пуссен доказали «теорему о простых числах», пользуясь «аналитическими» методами, т.е. математическими средствами, не относящимися к области целых чисел. Первое доказательство без использования таких средств было получено в 1949 г. (Эрдёш).

4. Пьер Ферма сформулировал свою «малую теорему» в письме к другу Френикл де Бесси 18 октября 1640 г. Модулярная арифметика была изобретена и сформулирована Карлом Фридрихом Гауссом.

5. Кармайкловы числа были названы по имени американского математика Кармайкла (R.D. Carmichael), исследовавшего их свойства в 1909 г.

6. В литературе греко-китайская теорема об остатках называется просто «китайской теоремой об остатках» и приписывается Сунь-Цзы, жившему в первом[1)] веке нашей эры. В своей книге «Суань–Цзинь» («Математический трактат») Сунь-Цзы дает правило «тьятен» (= большое обобщение) определения натурального числа, дающего остатки 2, 3, 2 при делении на 3, 5, 7 соответственно. Однако греческий математик Никомах из Герасы, также живший в первом веке нашей эры, упоминает подобную задачу в своей книге «Введение в арифметику», а именно он вводит как игру метод для определения натурального числа по остаткам, полученным от деления этого числа на другие натуральные числа (Nicomachi Geraseni Pythagorei Introductionis Arithmeticae, Libri II, Lipsiae, 1866). Эти методы сходны, и мы отдаем должное обоим. (Следует также отметить, что Никомах был пифагорийским философом и что решето Эратосфена описано в его книге.)

Абрамов С.А. Некоторые оценки, связанные с алгоритмом Евклида, *Журнал выч. мат. и мат. физ.* **19**, 756–760, 1979.

Абрамов С.А., Рыбин С.И. Обобщение бинарного алгоритма вычисления наибольшего общего делителя целых чисел. В кн. *Вопросы математической логики и теории алгоритмов*, 34–37, ВЦ АН СССР, Москва, 1988.

Adleman L.M., Pomerance C., Rumely R.S. On distinguishing prime numbers from composite numbers, *Annals of Mathematics* **117**, 173–206, 1983.

Bradley G.H. Algorithm and bound for the greatest common divisor of n integers, *Communications of the ACM* **13**, 433–436, 1970.

[1)] По другим данным — в III в. н.э. — *Прим. ред.*

Childs L. *A concrete introduction to higher algebra.* Springer-Verlag, New York, 1979.

Cohen H., Lenstra H.W. Jr. *Primality testing and Jacobi sums.* Report 82-18, Mathematical Institut of the University of Amsterdam, Amsterdam, 1982.

Dickson L.E. *History of the theory of numbers.* Chelsea, New York, 1952.

Dirichlet P.G.L. *Abhandlungen der koeniglichen preussischen Akademie der Wissenschaften*, 1849, 69–83.

Dixon J.D. Asymptotically fast factorization of integers. *Mathematics of Computation* **36**, 255–260, 1981.

Dixon J.D. Factorization and primality tests. *American Mathematical Monthly* **91**, 333–352, 1984.

Dudley U. Formulas for primes. *Mathematics Magazine* **56**, 17–22, 1983.

Erdoes P. On a new method in elementary number theory which leads to an elementary proof of the prime number theorem. *Proceedings of the National Academy of Sciences* (USA) **35**, 374–384, 1949.

Goldstein L.J. A history of the prime number theorem. *American Mathematical Monthly* **80**, 599–614, 1973.

Gregory R.T., Krishnamurthy E.V. *Methods and Applications of Error-free Computation.* Springer-Verlag, New York, 1984. [Имеется перевод: Грегори Р., Кришнамурти Е. Безошибочные вычисления. Методы и приложения. — М.: Мир, 1988.]

Guy R.K. How to factor a number. Congressus Numerantium XVI, *Proceedings of the Fifth Manitoba Conference on Numerical Mathematics* (Winnipeg, 1976) pp. 49–89.

Knuth D. *The art of computer programming.* Vol. 2. *Seminumerical algorithms.* Addison-Wesley, Reading, MA, 1969. [Имеется перевод: Кнут Д. *Искусство программирования для ЭВМ.* Т. 2. — М.: Мир, 1977.]

Lagrange J.L. *Traité de la resolution des equations numeriques.* Paris, 1798.

Lamé Gabriel. Note sur la limite du nombre des divisions dans la recherche du plus grand commun diviseur entre deux nombres entiers. *Comptes Rendues de l'Académie des Sciences* (Paris) **19**, 867–870, 1844.

Lang S., Trotter H. Continued fractions for some algebraic numbers. *Journal fuer die reine und angewandte Mathematik* **255**, 112–134, 1972.

Legendre A.M. Théorie des nombres. Paris, 1798.

Lehman R.S. Factoring large integers. *Mathematics of Computation* **28**, 637–646, 1974.

LeVeque W.J. *Fundamentals of number theory.* Addison-Wesley, Reading, MA, 1977.

Levinson N. A motivated account of an elementary proof of the prime number theorem. *American Mathematical Monthly* **76**, 225–245, 1969.

Lipson J.D. *Elements of algebra and algebraic computing.* Addison-Wesley, Reading, MA, 1981.

Lucas E. *Théorie des nombres.* Blanchard, Paris, 1961.

Mills W.H. A prime representing function. *Bulletin of the American Mathematical Society* **53**, 604, 1947.

Morrison M.A., Brillhart J. A method of factoring and the factorization of F_7. *Mathematics of Computaion* **29**, 183–205, 1975.

Motzkin T.S. The Euclidean algorithm. *Bulletin of the American Mathematical Society* **55**, 1142–1146, 1949.

Niven I., Zuckerman H.S. *An introduction to the theory of numbers*, 4th ed. Wiley, New York, 1980.

Olds C.D. *Continued fractions.* Random House, New York, 1963.

Pomerance C. Recent developments in primality testing. *The Mathematics Intelligencer* **3**, 97–105, 1981.

Pratt V.R. Every prime has a succinct certificate. *SIAM Journal of Computing* **4**, 214–220, 1975.

Rabin M.O. Probabilistic algorithm for testing primality. *Journal of Number Theory* **12**, 128–138, 1980.

Richards I. Continued fractions without tears. *Mathematics Magazine* **54**, 163–171, 1981.

Schroeder M.R. *Number theory in science and communication.* Springer-Verlag, New York, 1984 and 1986 (2nd ed.)

Scott N.R. *Computer number systems and arithmetic.* Prentice-Hall, Englewood Cliffs, N.J., 1985.

Sims C.C. *Abstract algebra, a computational approach.* Wiley, New York, 1984.

Solovay R., Strassen V. A fast Monte Carlo test for primality. *SIAM Journal of Computing* **6**, 1977, 84–85; erratum ibid., **7**, 118, 1978.

Wilf H.S. *Algorithms and complexity.* Prentice-Hall, Englewood Cliffs, N.J., 1986.

Williams H.C. Primality testing on a computer. *Ars Combinatoria* **5**, 127–185, 1978.

Williams H.C. The influence of computers in the development of number theory. *Computers and Mathematics with Applications* **8**, 75–93, 1982.

Williams H.C. Factoring on a computer. *The Mathematics Intelligencer* **6**, 29–36, 1984.

Wunderlich M. A running time analysis of Brillhart's continued fraction factoring method. Number theory Cardondale 1979, *Lecture notes in Mathematics no.* 751. Springer-Verlag, Berlin, 1979, pp. 328–342.

Глава 3

Полиномы

В этой главе мы познакомимся с полиномами; мы обнаружим, что с многих точек зрения полиномиальная арифметика родственна арифметике целых чисел: для полиномов имеется некоторая версия алгоритма Евклида и даже греко-китайская теорема об остатках. И еще многое из того, что мы узнали о целых числах (системы вычетов и т.д.), можно обобщить на полиномы, и здесь имеется бесконечно много приложений. Наиболее важным приложением полиномов с целыми коэффициентами является построение полей Галуа, которые играют ключевую роль в вычислительных науках.

3.1

Основные понятия

В этом разделе мы формально вводим полиномы и исследуем различные алгоритмы выполнения операций над ними. Мы будем главным образом интересоваться полиномами над кольцом целых чисел и над (конечными) полями. Читатель должен помнить об этом при изучении материала, поскольку полиномы по-разному «ведут себя» над этими двумя различными алгебраическими структурами.

3.1.1. Основные сведения о полиномах

Пусть J — область целостности и x — неизвестная, т.е. x — это не независимый элемент области J, а просто формальный символ. Выражение вида

$$p(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0, \quad \text{или}$$
$$p(x) = c_0 x^n + c_1 x^{n-1} + \cdots + c_{n-1} x + c_n,$$

где $c_i \in J$, $i = 0, 1, \ldots, n$, называется *полиномом от x с коэффициентами из J* или *полиномом от x над J*; в этой книге мы будем использовать обе формы записи полиномов. Каждое выражение $c_i x^i$ называется *членом* степени i полинома $p(x)$. Если коэффициент при x^n не равен 0, то n — *степень* полинома; она обозначается $\deg[p(x)]$;

коэффициент при x^n называется *старшим* коэффициентом полинома $p(x)$ и обозначается $\mathrm{lc}[p(x)]$; если $\mathrm{lc}[p(x)] = 1$ (единичный элемент области J), то полином называется нормированным. При описании полиномов мы обычно будем использовать неизвестную, но в некоторых случаях, в частности при умножении или делении, полином будет представляется только своими коэффициентами, т.е. будет рассматриваться как кортеж коэффициентов.

Два полинома $p_1(x) = \sum_{0 \le i \le m} c_i x^i \neq 0$ и $p_2(x) = \sum_{0 \le i \le n} d_i x^i \neq 0$ с коэффициентами в J равны тогда и только тогда, когда $c_0 = d_0$, $c_1 = d_1, \ldots, c_i = d_i, \ldots$ для всех i. Для этих полиномов $p_1(x)$ и $p_2(x)$ определяем их сумму и произведение следующим образом:

$$p_1(x) + p_2(x) = \sum_{0 \le i \le m} c_i x^i + \sum_{0 \le i \le n} d_i x^i = \sum_{0 \le i \le \max(m,n)} (c_i + d_i) x^i,$$

$$p_1(x) \cdot p_2(x) = \Big(\sum_{0 \le i \le m} c_i x^i \Big) \Big(\sum_{0 \le j \le n} d_j x^j \Big) = \sum_{0 \le h \le m+n} \Big(\sum_{i+j=h} c_i d_j \Big) x^h.$$

Ясно, что и сумма, и произведение полиномов $p_1(x), p_2(x)$ представляют собой некоторый полином от x с коэффициентами из той же самой области целостности J. Мы можем теперь легко показать, что множество полиномов от x с коэффициентами из области целостности J *само является областью целостности, обозначаемой* $J[x]$. ($J[x]$ содержит как J, так и неизвестную x.) Чтобы в этом убедиться, рассмотрим полиномы $p_1(x)$ и $p_2(x)$, определенные выше; покажем, что $p_1(x) \cdot p_2(x) \neq 0$, где 0 обозначает нулевой полином $0 + 0x + 0x^2 + \ldots$. Предположим, что $\deg[p_1(x)] = m$ и $\deg[p_2(x)] = n$, так что $c_m \neq 0$ и $d_n \neq 0$. Отсюда следует, что $c_m d_n \neq 0$, поскольку c_m и d_n являются элементами области целостности J. Однако $c_m d_n$ — это коэффициент при x^{m+n} в произведении $p_1(x) \cdot p_2(x)$; следовательно, $p_1(x) \cdot p_2(x) \neq 0$, что означает отсутствие в $J[x]$ делителей нуля.

Следует заметить, что равенство полиномов, как оно определено выше, — это не то же самое, что равенство функций, несмотря на то что любой полином $p(x)$ из $J[x]$ определяет на J функцию путем подстановки, т.е. функция, определенная полиномом $p(x)$, отображает любой элемент j из J в $p(j)$. Однако, как вы помните, две функции равны в точности тогда, когда они имеют равные значения для каждого элемента из J, и возможно, что два полинома определяют одну и ту же функцию на J, не будучи равными как полиномы. Например, в кольце $\mathbb{Z}_2[x]$ полиномы $p_1(x) = x^3 - 1$ и $p_2(x) = x^5 - 1$ не равны как полиномы, но $p_1(0) = p_2(0) = 1$, $p_1(1) = p_2(1) = 0$, так что они

равны как функции на $\mathbb{Z}_2[x]$. В общем случае если J — область целостности с n элементами $a_0, a_1, \ldots, a_{n-1}$, где a_0, a_1 — нуль и единица соответственно, то полином

$$p(x) = x(x - a_1)(x - a_2) \ldots (x - a_{n-1}) \neq 0$$

равен нулю для всех x из J, и поэтому определяет функцию, равную функции, определенной нулевым полиномом. Мы далее докажем, что если область целостности J содержит бесконечное число элементов, например $J = \mathbb{R}$, то два полинома, определяющие одну и ту же функцию на J, должны быть равны как полиномы.

Тот факт, что мы можем ставить полиному в соответствие целое число ≥ 0, его степень, является чрезвычайно полезным, потому что это позволяет нам переносить на полиномы доказательства по индукции того же типа, что мы использовали для целых чисел. Ниже приводятся приложения.

Рассмотрим полином $p_2(x) \neq 0$ [мы говорим, что $p_2(x)$ — не *тождественный* нуль]. Полином $p_1(x)$ *делится* на полином $p_2(x)$, если существует полином $q(x)$, такой, что $p_1(x) = p_2(x) \cdot q(x)$; в этом случае мы пишем $p_2(x)|p_1(x)$. Степень полинома $p_2(x)$ не превосходит степени полинома $p_1(x)$, если, конечно, $p_1(x) \neq 0$; $p_2(x)$ называется *делителем* или *сомножителем* полинома $p_1(x)$. Отметим, что определенная так делимость не совпадает с введенной ранее делимостью целых чисел; например, $5|7$ имеет место, если $J = \mathbb{Q}$ и 5 и 7 рассматриваются как полиномы степени 0, в то время как целое число 5 не делит 7.

Теорема 3.1.1 (свойство евклидовости). Пусть J — область целостности; кроме того, пусть $p_1(x) = c_m x^m + c_{m-1}x^{m-1} + \cdots + c_1 x + c_0$ и $p_2(x) = d_n x^n + d_{n-1}x^{n-1} + \cdots + d_1 x + d_0 \neq 0$ — два полинома степеней m и n в кольце $J[x]$, и пусть d_n обратим в J. Тогда существуют единственные полиномы $q(x)$ и $r(x)$ в $J[x]$ (*частное* и *остаток* соответственно), такие, что

$$p_1(x) = p_2(x)q(x) + r(x), \quad \deg[r(x)] < \deg[p_2(x)].$$

Доказательство. Воспользуемся индукцией по степени делимого $p_1(x)$. Если $p_1(x) = 0$ или $\deg[p_1(x)] < \deg[p_2(x)]$, то положим $q(x) = 0$ и $r(x) = p_1(x)$. В противном случае, пусть $\deg[p_1(x)] = \deg[p_2(x)] + k$, $k \geq 0$, и образуем полином $p_1'(x) = p_1(x) - (c_m/d_n)x^k p_2(x)$. Тогда $\deg[p_1'(x)] < \deg[p_1(x)]$, поскольку мы уничтожили старшую степень неизвестной x. Если $p_1'(x) = 0$ или $\deg[p_1'(x)] < \deg[p_2(x)]$, то все

доказано, в противном случае по индукции $p_1'(x) = p_2(x)q_0(x) + r(x)$ для некоторых $q_0(x)$ и $r(x)$, таких, что $\deg[r(x)] < \deg[p_2(x)]$. Поэтому $p_1(x) = p_2(x)[q_0(x) + (c_m/d_n)x^k] + r(x)$, что доказывает существование полиномов $q(x), r(x)$. Ясно, что $q(x)$ и $r(x)$ – полиномы в кольце $J[x]$ и либо $r(x) = 0$, либо $\deg[r(x)] < \deg[p_2(x)]$. Для доказательства единственности предположим, что существует другая пара $q_1(x), r_1(x)$, такая, что $p_1(x) = p_2(x)q_1(x) + r_1(x)$, $\deg[r_1(x)] < \deg[p_2(x)]$. Тогда $r_1(x) = p_2(x)\{q_1(x) - q(x)\}$ и $p_2(x)|[r(x) - r_1(x)]$, что может иметь место, только если $r(x) - r_1(x) = 0$. Следовательно, $r(x) = r_1(x)$ и $q(x) = q_1(x)$. □

В гл. 5 мы обсуждаем, что случится, если d_n не является обратимым в кольце J. Заметим, что если J — поле, то для наличия свойства евклидовости или деления *достаточно*, чтобы полином-делитель $p_2(x)$ был отличен от нуля.

Мы имеем следующую процедуру, которая является хорошо известным синтетическим алгоритмом деления:

PDF. Полиномиальное деление над полем (**P**olynomial **D**ivision over a **F**ield)

Вход: $p_1(x) = \sum_{0\le i\le m} c_i x^i$ и $p_2(x) = \sum_{0\le i\le n} d_i x^i$ над полем, $m \ge n \ge 0$ и $d_n \ne 0$. (Этот алгоритм будет работать и над областью целостности J при условии, что d_n обратим в J.)

Выход: $q(x) = \sum_{0\le i\le m-n} q_i x^i$ и $\sum_{0\le i\le n-1} r_i x^i$, обладающие свойством евклидовости (теорема 3.1.1).

1. [Основной цикл] Для k от $m-n$ до 0 выполнять $\{q_k := c_{n+k}/d_n$; для j от $n+k-1$ до k выполнять $c_j := c_j - q_k d_{j-k}\}$.

2. [Выход] Вернуть q_i, $i = 0, 1, 2, \ldots, m-n$, коэффициенты полинома $q(x)$, вычисленного на шаге 1, и r_i, $i = 0, 1, 2, \ldots, n-1$, коэффициенты полинома $r(x)$, где $r_i = c_i$ (c_i также вычисляются на шаге 1).

Анализ времени работы алгоритма ***PDF***. Во все время работы алгоритма операции сложения, умножения и деления коэффициентов выполняются в поле (арифметика с одинарной точностью), и поэтому каждая операция выполняется за время ~ 1.

Шаг 1, основной цикл, очевидно доминирует во времени работы алгоритма. Этот цикл, очевидно, выполняется $(m-n+1)$ раз и при каждом выполнении имеет место одно деление и пересчитывается n

коэффициентов. Поэтому шаг 1 выполняется за время $O[n(m-n+1)]$ и

$$t_{\mathbf{PDF}}[p_1(x), p_2(x)] = O[n(m-n+1)].$$

То есть $t_{\mathbf{PDF}}[p_1(x), p_2(x)]$ равняется времени, которое необходимо для вычисления произведения $p_2(x)q(x)$ над полем (точно так же, как в случае деления целых чисел).

Пример. Рассмотрим полиномы $p_1(x) = 7x^5 + 4x^3 + 2x + 1$ и $p_2(x) = x^3 + 2x + 1$ с целыми коэффициентами. Поскольку старший коэффициент полинома $p_2(x)$ равен 1, мы можем применить **PDF** и получить

$$\begin{array}{rrrr|l}
7x^5 & +4x^3 & & +2x+1 & x^3+2 \\
\cline{5-5}
7x^5 & & +14x^2 & & 7x^2+4 \\
\cline{1-3}
 & 4x^3 & -14x^2 & +2x+1 & \\
 & 4x^3 & & +8 & \\
\cline{2-4}
 & & -14x^2 & +2x-7 &
\end{array}$$

Если мы будем выписывать только коэффициенты, то получим следующую таблицу:

$$\begin{array}{rrrrrr|l}
7 & 0 & 4 & 0 & 2 & 1 & 1\,0\,0\,2 \\
\cline{6-6}
7 & 0 & 0 & 14 & & & 7\,0\,4 \\
\cline{1-4}
 & 0 & 4 & -14 & 2 & 1 & \\
 & 0 & 0 & 0 & 0 & & \\
\cline{2-5}
 & & 4 & -14 & 2 & 1 & \\
 & & 4 & 0 & 0 & 8 & \\
\cline{3-6}
 & & & -14 & 2 & -7 &
\end{array}$$

Пусть J — область целостности, и рассмотрим $p(x)$ из $J[x]$. Если $\alpha \in J$, то можно разделить $p(x)$ на $x-\alpha$ (это может быть выполнено, поскольку коэффициент при x обратим) и получить

$$p(x) = (x-\alpha)q(x) + r(x), \quad \deg[r(x)] < \deg(x-\alpha) = 1,$$

т.е. $r(x)$ — константа из кольца J. Мы говорим, что $\alpha \in J$ — *корень* или *нуль* полинома $p(x)$, если $p(\alpha) = 0$.

Имеет место следующая

Теорема 3.1.2. Пусть J — область целостности, $p(x) \in J[x]$ и $\alpha \in J$. Тогда α — корень полинома $p(x)$, если и только если $(x-\alpha)|p(x)$.

Доказательство очевидно из приведенных выше рассуждений.

Следствие 3.1.3. Пусть J — область целостности, $p(x) \in J[x]$ и $\alpha \in J$. Тогда, если мы разделим $p(x)$ на $(x-\alpha)$, то остаток равен $p(\alpha)$.

Пример. Рассмотрим полином $x^2 - x - 2 \in \mathbb{Z}[x]$, имеющий корни -1 и 2. Тогда, очевидно, $(x+1)|(x^2-x-2)$ и $(x-2)|(x^2-x-2)$; более того, если мы разделим $x^2 - x - 2$ на $x - 3$, то получим в остатке $4 = (3)^2 - (3) - 2$. В следующем разделе мы увидим, как эффективно вычислять значение полинома в данной точке.

Если α — корень полинома $p(x)$ и $p(x) = (x-\alpha)^m q(x)$, $m \geq 1$, $q(\alpha) \neq 0$, то m называется *кратностью корня* α; если $m = 1$, то α называется *простым* корнем.

Имеет место следующая

Теорема 3.1.4. Пусть J — область целостности и $p(x) \neq 0$ — полином из $J[x]$. Если степень полинома $p(x)$ равна n, то $p(x)$ имеет не более n корней с учетом кратностей. Эти корни лежат в J или в большей области.

Доказательство. Предположим, что $\alpha_1, \alpha_2, \ldots, \alpha_m$ — различные корни полинома $p(x)$ в J или в большей области. (Случай кратных корней мы оставляем читателю.) Мы будем доказывать по индукции, что $p(x)$ делится на $(x-\alpha_1)(x-\alpha_2)\ldots(x-\alpha_m)$. Для $m = 1$ это выполняется по теореме 3.1.2. Пусть предположение индукции верно для $m-1$; тогда $p(x)$ можно представить в виде $p(x) = (x-\alpha_1)(x-\alpha_2)\ldots(x-\alpha_{m-1})q(x)$, для некоторого $q(x) \in J[x]$. Однако при $x = a_m$ мы получаем $(a_m-\alpha_1)(a_m-\alpha_2)\ldots(a_m-\alpha_{m-1})q(a_m) = 0$; поскольку все корни различны, отсюда следует, что $q(a_m) = 0$. Снова пользуясь теоремой 3.1.2, получаем $q(x) = (x-a_m)r(x)$, и, следовательно, $p(x) = (x-\alpha_1)(x-\alpha_2)\ldots(x-\alpha_{m-1})(x-\alpha_m)r(x)$; более того, из последнего выражения видно, что m не может быть больше, чем n, поскольку при $m = n$ полином $r(x)$ является константой, и это завершает доказательство теоремы.

Следующий пример показывает, что будет с теоремой 3.1.4, если J *не* является областью целостности.

Пример. Рассмотрим полином $x^2 - 1$ с коэффициентами из кольца $\mathbb{Z}_8$, содержащего делители нуля. Этот полином имеет четыре корня, а именно 1, -1 (или 7), 3 и 5.

Следствие 3.1.5. Пусть J — область целостности и $p(x) \in J[x]$. Если степень полинома $p(x)$ равна n и $p(x)$ имеет больше, чем n, корней, то $p(x) = 0$.

Теперь можно доказать следующую теорему.

Теорема 3.1.6. Пусть J — область целостности с единицей. Если она имеет бесконечное число элементов, то два различных полинома $p(x)$ и $q(x)$ из $J[x]$ всегда определяют различные полиномиальные функции.

Доказательство. Разность двух данных полиномов $p(x) - q(x) = d(x)$ определяет функцию $d_f(x)$, которая является разностью полиномиальных функций $p_f(x)$ и $q_f(x)$, определенных полиномами $p(x)$ и $q(x)$ соответственно. Если функции $p_f(x)$ и $q_f(x)$ равны, то их разность должна быть равна 0 для любого x из J. Однако из этого следует, что каждый элемент кольца J будет корнем полинома $d(x)$, т.е. если $\deg[d(x)] = n_d$, то $d(x)$ имеет больше, чем n_d, корней, откуда следует, что $d(x) = 0$. Поэтому $p(x) = q(x)$. □

3.1.2. Метод Руффини–Горнера

В этом разделе мы рассмотрим два эффективных алгоритма, которые помогут нам (1) вычислить значение полинома $p(x)$ в данной точке $x = \alpha$ и (2) вычислить новый полином $p(y)$, где $x = \alpha + y$.

Рассмотрим область целостности $J[x]$ и полином

$$p(x) = c_0 + c_1 x + \cdots + c_n x^n$$

из этой области, *значение* которого в точке $x = \alpha$ мы хотим вычислить. Очевидно, наиболее прямой способ достигнуть этого состоит в том, чтобы заменить x на α в выписанном полиномиальном выражении и вычислять каждый член отдельно. Заметим, что это довольно трудоемкий процесс. Существует, однако, другой способ, а именно можно использовать метод Руффини–Горнера. [Он хорошо известен как *метод Горнера*, но Руффини опередил Горнера на 15 лет; см. статью Кайори (Cajori, 1911).]

Метод Руффини–Горнера для эффективного вычисления $p(\alpha)$ работает следующим образом. Положим $p_0 := c_n$; умножив это равенство на α и прибавив c_{n-1}, получим $p_1 := \alpha p_0 + c_{n-1}$. Затем, умножив p_1 на α и прибавив c_{n-2}, получим $p_2 := \alpha p_1 + c_{n-2}$ и т.д. То есть рекурсивная схема этого процесса такова:

$$\begin{cases} p_0 := c_n, \\ p_k := \alpha p_{k-1} + c_{n-k}, \quad k > 0, \end{cases} \tag{**RH**}$$

и мы получаем *вложенную* форму

$$p(\alpha) = c_0 + \alpha\{c_1 + \alpha[c_2 + \cdots + \alpha(c_{n-1} + \alpha c_n)\ldots]\}.$$

Анализ времени работы ***метода Руффини–Горнера***. Мы рассмотрим следующие два случая:

Случай a. Коэффициенты полинома $p(x)$ и точка α принадлежат (конечному) полю. В этом случае мы имеем n операций сложения и умножения с одинарной точностью, где $n = \deg[p(x)]$; поэтому $p(\alpha)$ вычисляется за время $O(n)$.

Случай b. Коэффициенты полинома $p(x)$ и точка α принадлежат кольцу целых чисел. В этом случае мы имеем n сложений и умножений длинных целых чисел, и, как мы отмечали в разд. 1.2, время умножений доминирует над временем сложений; поэтому мы предполагаем, что только умножаем n различных членов на α, где $n = \deg[p(x)]$. Если μ — значение наибольшего члена, полученного при выполнении метода Руффини–Горнера, то, очевидно,

$$t_{\textbf{R-H}}[p(x), \alpha] = O(nL(\alpha)L(\mu)].$$

Чтобы оценить μ, положим $d = |p(x)|_\infty$ [наибольший по абсолютному значению коэффициент $p(x)$] и рассмотрим «наихудший» возможный случай, когда все коэффициенты полинома $p(x)$ равны d; тогда μ получается из следующей схемы:

$$\begin{array}{cccccc} d & d & d & \ldots & d \\ d & d(\alpha+1) & d(\alpha^2+\alpha+1) & \ldots & d(\alpha^n+\alpha^{n-1}+\cdots+\alpha+1) \end{array}$$

То есть наибольший член, получающийся в течение вычислений, — это значение $p(x)$ в точке $x = \alpha$, и $\mu = d(\alpha^n + \alpha^{n-1} + \cdots + \alpha + 1)$. Однако $\mu = d(\alpha^n + \alpha^{n-1} + \cdots + \alpha + 1) \le d[(n+1)\alpha^n]$ и, следовательно,

$$\begin{aligned} t_{\textbf{R-H}}(p(x), \alpha) &= O\{nL(\alpha)L[d(n+1)\alpha^n]\} \\ &= O(nL(\alpha)\{L(d) + L(n+1) + nL(\alpha)\}). \end{aligned}$$

Однако во всех практических приложениях $L(n+1) = 1$, и, поскольку $L(d) + nL(\alpha) \le nL(\alpha)L(d) + 1$, мы имеем

$$t_{\textbf{R-H}}[p(x), \alpha] = O\{n^2 L^2(\alpha) L[|p(x)|_\infty]\}.$$

Пример. Вычислим значение полинома $p(x) = x^3 - 7x + 7$ в точке $\alpha = 3$, работая с целыми числами. Пользуясь методом Руффини–Горнера, получаем $p(3) = 13$.

$$\begin{array}{cccc} 1 & 0 & -7 & 7 \\ 1 & 3 & 2 & \mathit{13} \end{array}$$

Мы действуем следующим образом. В первой строке выписываем все коэффициенты полинома $p(x)$, включая нулевые; старший коэффициент стоит слева. Вторая строка получается так: первый (крайний левый) элемент второй строки — это старший коэффициент полинома $p(x)$. Этот первый элемент умножаем на α, прибавляем к произведению второй элемент (первой строки) и записываем сумму как второй элемент второй строки; в нашем примере $\alpha = 3$, и мы имеем $1 \cdot 3 + 0 = 3$. В общем случае, чтобы вычислить «следующий» элемент второй строки, умножаем последний вычисленный элемент второй строки на α и прибавляем к произведению «следующий» коэффициент из первой строки. Таким образом, продолжая наш пример, мы имеем $3 \cdot 3 - 7 = 2$ и $2 \cdot 3 + 7 = 13$, что дает $p(3)$.

Заметим, что $p(3) = 13$, последний элемент второй строки в приведенном примере, — остаток, полученный при делении $x^3 - 7x + 7$ на $x - 3$. Частное $q(x)$ этого деления также вычисляется в упомянутой выше схеме и получается из остальных элементов второй строки; а именно $q(x) = x^2 + 3x + 2$. Этот метод известен также как *синтетический алгоритм деления.*

Рассмотрим теперь в области целостности $J[x]$ следующий полином от x:

$$p_n(x) = \sum_{0 \le i \le n} c_i x^{n-i} = c_0 x^n + c_1 x^{n-1} + \cdots + c_n, \tag{RH1}$$

из которого мы хотим получить другой полином от y, где $x = \alpha + y$. Очевидно, что полином от y может быть вычислен с использованием теоремы о разложении Тейлора, согласно которой $p(\alpha + y) = \sum_{0 \le i \le n} [p^{(i)}(\alpha)/i!] y^i$. Однако мы можем действовать лучше; заметим, что эта подстановка дает уравнение от y с коэффициентами b_i, как показывают следующие соотношения:

$$\begin{aligned} \sum_{0 \le i \le n} c_i x^{n-i} &= \sum_{0 \le i \le n} c_i (y + \alpha)^{n-i} \\ &= \sum_{0 \le i \le n} b_i y^{n-i} = \sum_{0 \le i \le n} b_i (x - \alpha)^{n-i}. \end{aligned} \tag{RH2}$$

Используя первое и последнее выражения в (RH2), мы покажем, что коэффициенты b_i преобразованного полинома могут быть получены последовательным применением синтетического алгоритма деления.

Мы можем написать

$$p_n(x) = \sum_{0 \le i \le n} c_i x^{n-i} = (x - \alpha) p_{n-1}(x) + r_n, \tag{RH3}$$

где $p_{n-1}(x)$ — полином степени $n-1$, а r_n — коэффициент b_n в (RH2). Если мы выразим $p_{n-1}(x)$ в виде

$$p_{n-1}(x) = a_0 x^{n-1} + a_1 x^{n-2} + \cdots + a_{n-1}, \tag{RH4}$$

подставим в (RH3) и приравняем коэффициенты при одинаковых степенях x в обеих частях равенства, то получим

$$a_0 = c_0, \quad a_j = c_j + \alpha a_{j-1}, \quad j = 1, 2, \ldots, n, \tag{RH5}$$

что является синтетическим алгоритмом деления, рассмотренным выше. Заметим, что последний коэффициент a_n — в точности остаток r_n в (RH3), или коэффициент b_n в (RH2).

Дальнейшее применение того же самого процесса к $p_{n-1}(x)$ дает

$$p_{n-1}(x) = (x-\alpha)p_{n-2}(x) + r_{n-1}, \tag{RH6}$$

где $p_{n-2}(x)$ — полином степени $n-2$. Сочетая (RH3) и (RH6), мы получаем

$$p_n(x) = (x-\alpha)^2 p_{n-2}(x) + r_{n-1}\cdot(x-\alpha) + r_n.$$

Если процесс повторить n раз, то получим

$$p_n(x) = r_0\cdot(x-\alpha)^n + r_1\cdot(x-\alpha)^{n-1} + \cdots + r_n,$$

где коэффициенты r_i — это коэффициенты b_i, появляющиеся как остатки при каждом применении алгоритма (RH5). В частности, при $\alpha = 1$ этот алгоритм не требует умножений, поэтому преобразование $x = y + 1$ можно выполнить очень эффективно. Мы оставляем читателю в качестве упражнения показать, что для вычисления $p(y)$, где $x = \alpha + y$, необходимо время

$$O\{n^3 L^2(\alpha) + n^2 L(\alpha) L[|p(x)|_\infty]\}.$$

Пример. Рассмотрим полиномиальное уравнение $p(x) = x^3 - 7x + 7 = 0$, для которого мы хотим вычислить полином $p(1+y)$. Повторным применением синтетического алгоритма деления преобразованный полином от y может быть получен двумя следующими способами:

Руффини (1804)

1	0	-7	7	
	1	1	-6	*1*
		1	2	*-4*
			1	*3*
				1

Горнер (1819)

1	0	-7	7
1	1	-6	*1*
1	2	*-4*	
1	*3*		
1			

В обоих случаях преобразованное уравнение от y имеет вид $y^3 + 3y^2 - 4y + 1 = 0$. (В каждом случае читайте коэффициенты последнего столбца снизу вверх.)

3.1.3. Интерполяция над полем

Пусть теперь J — поле, и рассмотрим совокупность $n+1$ «пробных» точек $(a_i, b_i) \in J \times J,\ i = 1, 2, \ldots, n+1$ и a_i различны. Тогда задача интерполяции над J состоит в том, чтобы найти полином $p(x) \in J[x]$, такой, что

$$p(a_i) = b_i, \quad i = 1, 2, \ldots, n+1.$$

Задача интерполяции имеет важное значение во многих областях математики, и в этом разделе мы рассмотрим два метода ее решения. (В гл. 6 мы воспользуемся интерполяцией для получения сомножителей полинома.)

Начнем с интерполяции Лагранжа.

Теорема 3.1.7. Пусть $(a_i, b_i) \in J \times J,\ J$ — поле, $i = 1, 2, \ldots, n+1$ и a_i различны. Тогда существует единственный полином $p(x) \in J[x]$ степени $\leq n$, такой, что $p(a_i) = b_i,\ i = 1, 2, \ldots, n+1$.

Доказательство. (Существование.) Для доказательства существования $p(x)$ воспользуемся *интерполяционной формулой Лагранжа*

$$p(x) = \sum_{1 \leq i \leq n+1} b_i L_i(x),$$

где

$$L_i(x) = \frac{(x-a_1)\ldots(x-a_{i-1})(x-a_{i+1})\ldots(x-a_{n+1})}{(a_i-a_1)\ldots(a_i-a_{i-1})(a_i-a_{i+1})\ldots(a_i-a_{n+1})}.$$

Проверкой убеждаемся, что $L_i(a_j) = 0$ для $i \neq j$ благодаря сомножителю $(a_j - a_j)$ в числителе $L_i(x)$, в то время как $L_i(a_i) = 1$; следовательно, $p(a_i) = b_i$, что и требовалось. Более того, по построению $\deg[p(x)] \leq n$.

(Единственность.) Предположим, что существует другой полином $p'(x) \in J[x]$, такой, что $p'(a_i) = p(a_i) = b_i,\ i = 1, 2, \ldots, n+1$. Тогда полином $p'(x) - p(x)$ имеет, очевидно, $n+1$ корней, но его степень $\leq n$. Из теоремы 3.1.4 видно, что это может быть только в случае, когда $p'(x) - p(x) = 0$. Следовательно, $p'(x) = p(x)$.

Пример. Рассмотрим «пробные» точки $a_1 = 0$, $b_1 = 0$, $a_2 = 1$, $b_2 = 1$, $a_3 = -1$, $b_3 = 1$, $a_4 = 3$, $b_4 = 9$, и пусть $J = \mathbb{R}$. Заметим, что значения

b_i не обязаны быть все различными, и в примере встречаются одинаковые. Пользуясь формулой Лагранжа и выполняя вычисления в $\mathbb{R}$, мы получаем следующий полином $p(x) \in \mathbb{R}[x]$:

$$\begin{aligned} p(x) =& 0\left[\frac{(x-1)(x+1)(x-3)}{(0-1)(0+1)(0-3)}\right] + 1\left[\frac{(x-0)(x+1)(x-3)}{(1-0)(1+1)(1-3)}\right] \\ &+ 1\left[\frac{(x-0)(x-1)(x-3)}{(-1-0)(-1-1)(-1-3)}\right] + 9\left[\frac{(x-0)(x-1)(x+1)}{(3-0)(3-1)(3+1)}\right] \\ =& x^2. \end{aligned}$$

Проверка: $p(0) = 0,\ p(1) = 1,\ p(-1) = 1,\ p(3) = 9.$

Пример. Вычислим $p(x) \in \mathbb{Z}_{11}[x]$, такой, что $p(0) = 3,\ p(1) = 2,\ p(2) = 1$ и $p(3) = 2$. Работая с множеством неотрицательных целых чисел, получаем

$$\begin{aligned} p(x) =& 3\left[\frac{(x-1)(x-2)(x-3)}{(0-1)(0-2)(0-3)}\right] + 2\left[\frac{(x-0)(x-2)(x-3)}{(1-0)(1-2)(1-3)}\right] \\ &+ 1\left[\frac{(x-0)(x-1)(x-3)}{(2-0)(2-1)(2-3)}\right] + 2\left[\frac{(x-0)(x-1)(x-2)}{(3-0)(3-1)(3-2)}\right] \\ =& 3\cdot 5^{-1}(x^3+5x^2+5) + (x^3+6x^2+6x) \\ &+ 9^{-1}(x^3+7x^2+3x) + 2\cdot 6^{-1}(x^3+8x^2+2x) \\ =& 4x^3+10x^2+7x+3. \end{aligned}$$

Проверка: $p(0) = 3, p(1) = 2, p(2) = 1, p(3) = 2$ над $\mathbb{Z}_{11}$.

Посмотрим теперь, как решать задачу интерполяции с использованием греко-китайской теоремы об остатках.

Из следствия 3.1.3 мы знаем, что если разделить $p(x) \in J[x]$ на $(x-a)$, $a \in J$, то в остатке получим $p(a)$, т.е.

$$p(x) = (x-a)q(x) + p(a).$$

Поэтому $(x-a)|(p(x)-p(a))$, а следовательно,

$$p(a) = b \quad \text{тогда и только тогда, когда} \quad p(x) \equiv b \pmod{(x-a)}.$$

Если рассматривать b как полином-константу, то видно, что мы ввели в $J[x]$ *отношение сравнимости* по модулю полинома. Сравнимость по модулю полинома $m(x)$ обладает свойствами, аналогичными свойствам сравнимости $\bmod m$ для целых чисел, и все факты

относительно сравнимости, полученные для целых чисел, остаются справедливыми для сравнимости по модулю $m(x)$.

Учитывая, что мы хотим найти $p(x) \in J[x]$, такой, что

$$p(x) \equiv b_i \pmod{(x - a_i)}, \quad i = 1, 2, \ldots, n+1,$$

легко убеждаемся, что задача интерполяции — это частный случай греко-китайской задачи об остатках над $J[x]$.

Теперь, конечно, модули — это линейные полиномы, но поскольку все a_i различны, то различны и полиномы $(x - a_i)$; следовательно, они удовлетворяют ограничению быть попарно взаимно простыми. (См. теорему 6.2.12 и окружающий ее текст для формального распространения греко-китайской теоремы об остатках на полиномы. Также аналогично случаю целых чисел два полинома взаимно просты, если любой их наибольший общий делитель — константа; см. также следующий раздел.)

Следующая теорема покажет нам, как приспособить к нашим потребностям греко-китайский алгоритм, представленный в гл. 2.

Теорема 3.1.8. Пусть дано поле J, $p(x) \in J[x]$ и $a \in J$; тогда

a. $p(x) \pmod{(x - a)} = p(a)$.

b. Если $p(a) \neq 0$, то $p^{-1}(x) \pmod{(x - a)} = p^{-1}(a)$.

Доказательство.

a. Доказательство вытекает из следствия 3.1.3, согласно которому, чтобы найти остаток от деления $p(x)$ на $(x - a)$, достаточно просто вычислить значение $p(x)$ в точке $x = a$.

b. По определению мультипликативного обратного нам нужен единственный полином $v(x)$ степени $< 1[= \deg(x - a)]$, такой, что

$$p(x)v(x) \equiv 1 \pmod{(x - a)}.$$

Поскольку $\deg[v(x)] < 1$, $v(x)$ — константа; эта константа равна $p^{-1}(a)$, потому что

$$p(x) \cdot p^{-1}(a) \equiv \{p(x) \pmod{(x - a)}\}p^{-1}(a) \pmod{(x - a)}, \text{ или}$$
$$[p(x) \pmod{(x - a)}]p^{-1}(a) = p(a)p^{-1}(a) = 1.$$

□

Представляем теперь греко-китайский алгоритм, приспособленный к задаче интерполяции.

GCRAn-Interpolation. Интерполяция на основе греко-китайского алгоритма ()

Вход: $(a_i, b_i) \in J \times J,\ i = 1, 2, \ldots, n+1$, значения a_i различны и J — поле.

Выход: $p(x) \in J[x]$, такой, что $p(a_i) = b_i,\ i = 1, 2, \ldots, n+1$.

1. [Инициализация] $m(x) := 1;\ p(x) := b_1$.
2. [Основной цикл] Для i от 1 до n выполнять $\{m(x) := m(x) \cdot (x - a_i);\ c := m^{-1}(a_{i+1});\ q := [b_{i+1} - p(a_{i+1})] \cdot c;\ p(x) := p(x) + q \cdot m(x)\}$.
3. [Выход] Вернуть $p(x)$.

Анализ времени работы алгоритма ***GCRAn-Interpolation***. Поскольку упомянутый алгоритм используется исключительно над (конечными) полями, сложение и умножение коэффициентов выполняется за время ~ 1; более того, в этом случае расширенный алгоритм Евклида для целых чисел вычисляет мультипликативный обратный данного числа за время ~ 1.

Мы ясно видим, что шаг 2 доминирует над временем выполнения нашего алгоритма. Более того, при i-й итерации шага 2 мы имеем следующие четыре операции:

1. $m(x) := m(x)(x - a_i)$. После этого умножения полином $m(x)$ имеет степень i; очевидно, это умножение выполняется за время $O(i)$.
2. $c := m^{-1}(a_{i+1})$. Как мы знаем, вычисление над полем значения полинома степени i в данной точке требует времени $O(i)$; поскольку обращение выполняется за время ~ 1, ясно, что $c := m^{-1}(a_{i+1})$ вычисляется за время $O(i)$.
3. $q := [b_{i+1} - p(a_{i+1})]c$. Полином $p(x)$ в этой точке имеет степень $\leq i$, и, следовательно, $p(a_{i+1})$ вычисляется за время $O(i)$; это верхняя граница времени, необходимого для вычисления q, поскольку вычитание и умножение требуют время ~ 1.
4. $p(x) := p(x) + qm(x)$. Это выполняется за время $O(i)$.

Из приведенных рассуждений мы видим, что i-я итерация шага 2 выполняется за время $O(i)$, поэтому n итераций шага 2 выполняются за время

$$O\left(\sum_{0 \leq i \leq n} i\right) = O\left[\frac{n(n+1)}{2}\right] = O(n^2),$$

что является временем выполнения алгоритма **GCRAn-Interpolation**.

Пример. Вычислим $p(x) \in \mathbb{Z}_{11}[x]$, такой, что $p(0) = 3$, $p(1) = 2$, $p(2) = 1$, $p(3) = 0$. Применяя описанный алгоритм и работая с множеством неотрицательных целых чисел, мы получаем следующую таблицу:

i	a_i	b_i	$m(x)$	c	q	$p(x)$
1	0	3	1	–	–	3
2	1	2	x	1	10	$10x + 3$
3	2	1	$x^2 + 10x$	6	0	$10x + 3$
4	3	0	$x^3 + 8x^2 + 2x$	2	0	$10x + 3$

Таким образом, $p(x) = 10x + 3$ — решение данной задачи интерполяции. Проверка: $p(0) = 3,\ p(1) = 2,\ p(2) = 1$ и $p(3) = 0$ в $\mathbb{Z}_{11}$.

Однако если бы последняя «пробная» точка была $a_4 = 3$, $b_4 = 2$ вместо $a_4 = 3$, $b_4 = 0$, то на шаге 2 мы бы получили $q = (2 - 0)2 = 4$ и $p(x) = 10x + 3 + 4(x^3 + 8x^2 + 2x) = 4x^3 + 10x^2 + 7x + 3$, что и было бы решением нашей задачи. Проверка: $p(0) = 3$, $p(1) = 2$, $p(2) = 1$ и $p(3) = 2$ в $\mathbb{Z}_{11}$.

Читателю следует обратить внимание на то, что точно так же, как в случае алгоритма **GCRAk**, решение для $(n + 1)$-точечной задачи интерполяции строится следующим образом:

$$\begin{aligned} p(x) =& q_1 + q_2(x - a_1) + q_3(x - a_1)(x - a_2) \\ &+ q_4(x - a_1)(x - a_2)(x - a_3) \\ &+ \cdots + q_{n+1}(x - a_1)(x - a_2)\ldots(x - a_n), \end{aligned}$$

где $q_1 = b_1 \in J$ и q_i, $i > 1$, вычисляется на $(i - 1)$-й итерации описанного алгоритма. С другой стороны, интерполяционная формула Лагранжа дает

$$p(x) = \sum_{1 \le i \le n+1} b_i \prod_{1 \le j \le n+1, j \ne i} \frac{x - a_j}{a_i - a_j}.$$

Сравнивая две различные формы полинома $p(x)$, замечаем, что только первая форма *расширяема* в том смысле, что присоединить новую «пробную» точку можно, выполнив еще одну итерацию алгоритма **GCRAn-Interpolation**. С другой стороны, с добавлением новой пробной точки все коэффициенты полинома Лагранжа должны быть вычислены заново.

3.1.4. Вычисления, использующие схему: (вычисление значений)–интерполяция

Для наших дальнейших рассуждений нам понадобится следующее. Полином $p(x)$ в $J[x]$, J — область целостности, называется *простым*, если из условия $p(x) = p_1(x)p_2(x)$, следует, что или $p_1(x)$, или $p_2(x)$ является обратимым. [Полином $p(x)$ называется обратимым, если существует другой полином $q(x)$, такой, что $p(x)q(x) = 1$.] Полином $p(x)$ в $J[x]$, J — область целостности, называется *неприводимым*, если из условия $p(x) = p_1(x)p_2(x)$, следует, что или $p_1(x)$, или $p_2(x)$ является полиномом нулевой степени, т.е. принадлежит J. Очевидно, что всякий простой полином является неприводимым, но обратное утверждение в общем случае неверно. Заметим, что если J — поле, то все обратимые полиномы являются полиномами нулевой степени (константами); значит, в этом случае, для того чтобы полином $p(x)$ был прост, необходимо, чтобы хотя бы один из полиномов $p_1(x)$, и $p_2(x)$ был константой, т.е. над полем понятия неприводимого и простого полинома совпадают.

Пример.

1. $3x^2 + 3$ неприводим над $\mathbb{R}$.
2. $3x^2 + 3 = 3(x^2 + 1)$ не будет простым над $\mathbb{Z}$, поскольку ни 3, ни $x^2 + 1$ не являются обратимыми в кольце $\mathbb{Z}[x]$, однако этот полином неприводим над $\mathbb{Z}$.
3. $x^2 + 1$ неприводим над $\mathbb{Z}$, $\mathbb{Q}$ или $\mathbb{R}$, но приводим над $\mathbb{C}$, $x^2 + 1 = (x + i)(x - i)$.
4. $x^2 - 2$ неприводим над $\mathbb{Q}$, поскольку $\sqrt{2} \notin \mathbb{Q}$.

Точно как же, как мы определили $\equiv_m$ в $\mathbb{Z}$, мы можем теперь определять отношение эквивалентности в $J[x]$, где J — поле.

Пусть $m(x) = x^n + \cdots + \mu_2 x^2 + \mu_1 x + \mu_0$ — нормированный полином в $J[x]$ степени $n > 0$. Тогда для любого $p(x) \in J[x]$ мы имеем

$$p(x) = m(x)q(x) + r(x), \quad \deg[r(x)] < \deg[m(x)].$$

Остаток $r(x)$ в приведенных соотношениях обозначается $r_{m(x)}[p(x)]$ или, эквивалентно, $p(x)$ $[\mathrm{mod}\ m(x)]$. Если $p(x)$ $[\mathrm{mod}\ m(x)] = 0$, то $m(x)$ делит без остатка $p(x)$ и мы пишем $m(x)|p(x)$.

Определим теперь отношение эквивалентности $\mathrm{mod}\, m(x)$ или $\equiv_{m(x)}$ в кольце $J[x]$ следующим образом:

$p_1(x) \equiv_{m(x)} p_2(x)$ тогда и только тогда, когда $m(x)|[p_1(x) - p_2(x)]$.

Читатель должен проверить, что это — отношение эквивалентности. Мы затем определяем $J[x]/\equiv_{m(x)}$ или просто $J[x]_{m(x)} = \{p(x) \in J[x] : \deg[p(x)] < \deg[m(x)]\}$; каждый класс эквивалентности, обозначаемый $[p(x)]_{m(x)}$ или просто $[p(x)]$, имеет единственный представитель в $J[x]_{m(x)}$, а именно $r_{m(x)}[p(x)]$, $\deg[r(x)] < \deg[m(x)]$. Определим сложение и умножение классов эквивалентности формулами

$$[p_1(x)]_{m(x)} + [p_2(x)]_{m(x)} = [p_1(x) + p_2(x)]_{m(x)},$$
$$[p_1(x)]_{m(x)} \cdot [p_2(x)]_{m(x)} = [p_1(x) \cdot p_2(x)]_{m(x)}.$$

Легко показать, что определенные таким образом операции превращают $J[x]_{m(x)}$ в кольцо (аналог теоремы 2.3.11.).

Мы можем рассматривать J как подмножество кольца $J[x]_{m(x)}$, отождествляя $a \in J$ с $[a]_{m(x)}$. [Это может быть сделано, потому что если $[a]_{m(x)} = [b]_{m(x)}$ для a, b в J, то полином $m(x)$ должен делить $a - b$; однако $\deg[m(x)] \geq 1$, так что полином $a - b$ должен быть нулевым, следовательно, $a = b$ в J. Таким образом функция, которая отображает a из J на $[a]_{m(x)}$ из $J[x]_{m(x)}$, взаимно однозначна, и мы можем отождествлять a с $[a]_{m(x)}$.] Следовательно, каждый элемент кольца $J[x]_{m(x)}$ — это элемент кольца J плюс $J[x]$-кратное элемента $[x]_{m(x)}$. Если мы положим $\alpha = [x]_{m(x)}$, то каждый элемент кольца $J[x]_{m(x)}$ имеет вид $m_{n-1}\alpha^{n-1} + \cdots + m_2\alpha^2 + m_1\alpha + m_0$, где m_i принадлежит J для всех i; это то же самое, что сказать, что каждый элемент кольца $J[x]_{m(x)}$ — это единственная J-линейная комбинация элементов $1, \alpha, \alpha^2, \ldots, \alpha^{n-1}$. Поэтому удобно представлять элементы кольца $J[x]_{m(x)}$ как полиномы кольца $J[x]$, вычисленные в точке $\alpha = [x]_{m(x)}$, где $m(\alpha) = 0$, поскольку $m(\alpha) = [m(x)]_{m(x)} = [0]_{m(x)}$.

Если мы представляем элементы кольца $J[x]_{m(x)}$ (также обозначаемого $J[\alpha]$) как полиномы от α, то сложение и умножение в $J[\alpha]$ совпадают со сложением и умножением значений полиномов, вычисленных в точке α. Более того, при умножении в $J[\alpha]$ полезно иметь таблицу, выражающую степени α между n и $2n-2$ в терминах полиномов от α степени $< n$. (Значение этого факта мы оценим позже.)

Пример. Если $J = \mathbb{R}$, то нормированный полином $m(x) = x^2 + 1$ неприводим в $\mathbb{R}[x]$. Из предыдущего обсуждения следует, что каждый элемент кольца $\mathbb{R}[x]_{(x^2+1)}$ имеет вид $a + b[x]_{(x^2+1)}, a, b \in \mathbb{R}$. Если мы положим $i = [x]_{(x^2+1)}$, то заметим, что

$$\begin{aligned} i^2 &= [x]^2_{(x^2+1)} = [x^2]_{(x^2+1)} = [x^2 + 1 - 1]_{(x^2+1)} \\ &= [x^2 + 1]_{(x^2+1)} - [1]_{(x^2+1)} = -1. \end{aligned}$$

Тогда сложение и умножение — те же, что и для полиномов; например,

$$(a+ib)(c+id) = ac + i(ad+bc) + i^2bd.$$

Чтобы выразить этот результат в виде полинома от i степени ≤ 1, используем то, что $i^2 = -1$, и получим

$$(a+ib)(c+id) = ac - bd + i(ad+bc).$$

Более того, заметим, что $\mathbb{R}[x]_{(x^2+1)}$ — поле, потому что если a или $b \neq 0$, то

$$(a+ib)\left(\frac{a-ib}{a^2+b^2}\right) = 1.$$

Поэтому $\mathbb{R}[x]_{(x^2+1)}$ выглядит точно так же, как $\mathbb{C}$.

Пример. Рассмотрим нормированный полином $m(x) = x^2 + x + 1$, который неприводим над $\mathbb{Z}_2$, и пусть $J[x] = \mathbb{Z}_2[x]$. Тогда множество $\mathbb{Z}_2[x]/\equiv_{m(x)}$ содержит элементы $\{[0],[1],[x],[1+x]\}$. Все операции на этих элементах выполняются по модулю $m(x)$, т.е. все они выполняются по правилам кольца $Z_2[x]$, а результаты заменяются их остатками от деления на $m(x)$. Так,

$$[x+1]\cdot[x+1]+[x] = (x+1)(x+1)+x = (x^2+1)+x = (x^2+x+1) = [0].$$

Заметим также, что

$$[x]\cdot[x+1] = (x^2+x) = (x^2+x+1)+1 = [1],$$

так что $[x]^{-1} = [x+1]$ в $\mathbb{Z}_2[x]/\equiv_{m(x)}$. Таким образом, мы построили поле с 4 элементами.

Теорема 3.1.9. Если J — поле, то $J[x]_{m(x)}$ — коммутативное кольцо с единицей. Оно является полем тогда и только тогда, когда полином $m(x)$ неприводим в $J[x]$.

Доказательство. Доказательство аналогично доказательству теоремы 2.3.11. Например, тот факт, что $J[x]_{m(x)}$ удовлетворяет аксиомам коммутативного кольца с единицей, следует из того, что $J[x]$ удовлетворяет тем же самым аксиомам.

Теперь, чтобы показать, что $J[x]_{m(x)}$ является полем, если полином $m(x)$ неприводим, мы должны проверить, что ненулевые элементы обратимы. Пусть $p(x)$ – любой ненулевой полином степени $< n = \deg[m(x)]$; тогда $p(x)$ и $m(x)$ взаимно просты и, как мы увидим

в следующем разделе, существуют полиномы $f(x)$ и $g(x)$, такие, что $p(x)g(x) + m(x)f(x) = 1$. Тогда, вычисляя значение последнего выражения в точке α, где $\alpha = [x]_{m(x)}$ и $m(\alpha) = 0$, получаем $p(\alpha)g(\alpha) = 1$ в $J[x]_{m(x)}$, и, значит, $p(\alpha)$ обратим в $J[x]_{m(x)}$. Поскольку это верно для любого ненулевого элемента кольца $J[x]_{m(x)}$, это кольцо является полем. Доказательство обратного утверждения оставляется читателю в качестве упражнения. □

Из приведенных рассуждений видно, что мы имеем способ построения новых полей. Мы просто берем неприводимый полином $m(x)$ в $J[x]$ и рассматриваем $J[x]_{m(x)}$ — вот вам и поле.

Поле вида $J[x]_{m(x)}$ называется *простым расширением поля* J, и ниже мы увидим некоторые приложения таких полей.

Как мы уже видели при изучении целых чисел, если мы хотим вычислить окончательный результат, res, выражения с целыми аргументами, часто легче воспользоваться окольным путем; то же самое верно для вычисления значений выражений с полиномиальными аргументами. Предположим, например, что мы хотим вычислить окончательный результат $\mathrm{res}(x)$ выражения $e[i_1(x), i_2(x), \ldots, i_h(x)]$ над $J[x]$, где J — *поле* и $i_1(x), \ldots, i_h(x) \in J[x]$. Если работа над $J[x]$ «трудна», то мы можем работать над полями $J[x]_{m_k(x)}$ для различных k, где $m_k(x) = x - a_k$, очевидно, — неприводимый полином, $a_k \in J$. В этом случае, однако, мы должны знать границу для $\deg[\mathrm{res}(x)]$; например, если мы знаем, что $\deg[\mathrm{res}(x)] \leq n$, то индекс k пробегает значения $1, 2, \ldots, n+1$. Более того, заметим, что в результате нашего выбора полиномов $m_k(x)$ поле $J[x]_{m_k(x)}$ совпадает с J для любого k, и, таким образом, вычисления не представляют труда.

Окольный подход работает следующим образом. Для $k = 1, 2, \ldots$ $\ldots, n+1$ мы сначала вычисляем $i_j(a_k)$, $j = 1, 2, \ldots, h$, а затем при условии, что мы выбрали точки a_k так, что выражения e_{m_k} определены, получаем $\mathrm{res}_{m_k} = b_k = e_{m_k}[i_1(a_k), \ldots, i_h(a_k)]$. Окончательный результат $\mathrm{res}(x)$ получается тогда при помощи интерполяции по выбранным точкам (a_k, b_k), $k = 1, 2, \ldots, n+1$, и удовлетворяет условию $\mathrm{res}(a_k) = b_k$, $k = 1, 2, \ldots, n+1$. (См. рис. 3.1.1.)

Пример. Работая над $\mathbb{Z}_{11}[x]$, вычислим произведение полиномов $p_1(x) = 5x + 2$ и $p_2(x) = 8x^2 + 3$, используя схему, представленную на рис. 3.1.1. Заметим, что степень полинома $p(x) = p_1(x) \cdot p_2(x)$ меньше или равна 3, следовательно, мы должны вычислять значения в 4 точках, т.е. $k = 1, 2, 3, 4$. Полагая $a_1 = 0$, $a_2 = 1$, $a_3 = 2$ и $a_4 = 3$ и вычисляя $\{p_1(x) \pmod{(x - a_k)}\}\{p_2(x) \pmod{(x - a_k)}\}$, $k = 1, 2, 3, 4$,

Рис. 3.1.1.

Окольная схема вычисления выражения, использующая схему: (вычисление значения)–интерполяция.

над $\mathbb{Z}_{11}$, получаем следующие точки:

$$p_1(0)\cdot p_2(0) = 2\cdot 3 = 6 = b_1,$$
$$p_1(1)\cdot p_2(1) = 7\cdot 11 = 0 = b_2,$$
$$p_1(2)\cdot p_2(2) = 1\cdot 2 = 2 = b_3,$$
$$p_1(3)\cdot p_2(3) = 6\cdot 9 = 10 = b_4.$$

Имея 4 пробные точки, применяем алгоритм **GCRAn-Interpolation**.

i	a_i	b_i	$m(x)$	c	q	$p(x)$
1	0	6	1	–	–	6
2	1	0	x	1	5	$5x+6$
3	2	2	x^2+10x	6	4	$4x^2+x+6$
4	3	10	x^3+8x^2+2x	2	7	$7x^3+5x^2+4x+6$

и получаем $p(x) = 7x^3+5x^2+4x+6 = (5x+2)(8x^2+3)$, что совпадает с ответом, полученным ранее.

Проверка: $p(0)=6,\ p(1)=0,\ p(2)=2,\ p(3)=10$ над $\mathbb{Z}_{11}$.

Следует отметить, что мы могли воспользоваться алгоритмом **GCRAn-Interpolation**, потому что работали в поле. Наш подход должен быть несколько изменен, если J — только область целостности, и это — тема последующего обсуждения.

В этом случае мы хотим вычислять окончательный результат res(x) выражения $e[i_1(x), i_2(x), \ldots, i_h(x)]$ над $\mathbb{Z}[x]$, где res(x),

$i_1(x), \ldots, i_h(x) \in \mathbb{Z}[x]$. Следует заметить, что если $p(x) = \sum_{0 \le i \le n} c_i x^i \in Z[x]$, то

$$r_m[p(x)] = \sum_{0 \le i \le n} r_m(c_i) x^i,$$

где $r_m(c)$ для целого числа c обозначает остаток от деления c на m. Предположим теперь, что все коэффициенты res_i полинома $\mathrm{res}(x)$ удовлетворяют неравенствам $0 \le \mathrm{res}_i \le m_1 m_2 \ldots m_j$, где $m_1, \ldots, m_j$ суть j попарно взаимно простых модулей. Тогда для вычисления $\mathrm{res}(x)$ окольным методом можно сделать следующее. Для $k = 1, 2, \ldots, j$ при условии, что e_{m_k} определены, получаем $\mathrm{res}_{m_k}(x) = e_{m_k}(r_{m_k}[i_1(x)], \ldots, r_{m_k}[i_h(x)])$ над $\mathbb{Z}_{m_k}[x]$ и затем решаем греко-китайскую полиномиальную задачу об остатках

$$\mathrm{res}(x) \equiv res_{m_k}(x) \pmod{m_k}, \quad k = 1, 2, \ldots, j,$$

для наименьшего неотрицательного решения. [Если мы знаем границу для абсолютных значений коэффициентов res_i полинома $\mathrm{res}(x)$, то будем искать наименьшее по абсолютной величине решение.]

Пример. Мы хотим вычислить $p(x) = p_1(x) \cdot p_2(x)$, где $p_1(x) = 5x + 2$ и $p_2(x) = 8x^2 + 3$ над $\mathbb{Z}[x]$. Мы отмечаем, что все коэффициенты произведения $p(x)$ не превосходят 40, и, поскольку $3 \cdot 5 \cdot 7 > 40$, мы будем использовать только эти три модуля. Для $m_1 = 3$ мы получаем

$$r_3[p_1(x)] \cdot r_3[p_2(x)] = (2x + 2)(2x^2) = x^3 + x^2.$$

Для $m_2 = 5$ произведение равно

$$r_5[p_1(x)] \cdot r_5[p_2(x)] = (2)(3x^2 + 3) = x^2 + 1.$$

Наконец, для $m_3 = 7$ мы имеем

$$r_7[p_1(x)] \cdot r_7[p_2(x)] = (5x + 2)(x^2 + 3) = 5x^3 + 2x^2 + x + 6.$$

Итак, мы знаем, как выглядит $p(x) \pmod{m_k}$ для различных модулей. Чтобы восстановить сам $p(x)$, мы должны решать систему

$$\begin{aligned} p(x) &\equiv x^3 + x^2 \pmod 3, \\ p(x) &\equiv x^2 + 1 \pmod 5, \\ p(x) &\equiv 5x^3 + 2x^2 + x + 6 \pmod 7. \end{aligned}$$

Ясно, что $p(x) = ax^3 + bx^2 + cx + d$, где коэффициенты получаются решением следующих 4 систем:

$$\begin{aligned}
&a \equiv 1 \pmod 3, \quad b \equiv 1 \pmod 3, \quad c \equiv 0 \pmod 3, \quad d \equiv 0 \pmod 3,\\
&a \equiv 0 \pmod 5, \quad b \equiv 1 \pmod 5, \quad c \equiv 0 \pmod 5, \quad d \equiv 1 \pmod 5,\\
&a \equiv 5 \pmod 7, \quad b \equiv 2 \pmod 7, \quad c \equiv 1 \pmod 7, \quad d \equiv 6 \pmod 7.
\end{aligned}$$

Применяя 4 раза греко-китайский алгоритм, получаем $a = 40$, $b = 16$, $c = 15$ и $d = 6$. Другими словами, $p(x) = 40x^3 + 16x^2 + 15x + 6$, ответ правильный.

Обобщая сказанное выше, мы видим, что существуют приложения, в которых сам окончательный результат является полиномом, задаваемым в виде решения системы полиномиальных сравнений; т.е.

$$\begin{aligned}
p(x) &\equiv p_1(x) \pmod{m_1},\\
&\dots\\
p(x) &\equiv p_k(x) \pmod{m_k},
\end{aligned}$$

где p_i, $i = 1, 2, \dots, k$, — это полиномы $p_i = p_{0i} + p_{1i}x + p_{2i}x^2 + \dots + p_{ni}x^n \in \mathbb{Z}[x]$.

Чтобы восстановить $p(x)$, заметим, что для полиномов $q(x) = \sum_{0 \le j \le n} a_j x^j$ и $w(x) = \sum_{0 \le j \le n} d_j x^j$ из $\mathbb{Z}[x]$ и $m \in \mathbb{Z}^+$

$$q(x) \equiv w(x) \pmod m, \quad \text{если и только если} \quad a_j \equiv d_j \pmod m,$$

поскольку $m|[q(x) - w(x)]$ тогда и только тогда, когда $m|(a_j - d_j)$, $j = 0, 1, 2, \dots, n$.

Из сделанных замечаний следует, что если модули m_i (данных систем сравнений) попарно взаимно просты и мы полагаем $M = m_1 m_2 \dots m_k$, то система полиномиальных сравнений имеет два однозначно определенных решения, одно из которых — решение с наименьшими неотрицательными коэффициентами, а второе — решение с наименьшими по абсолютной величине коэффициентами. Любое из этих решений получается $n + 1$-кратным применением **GCRAk** (греко-китайского алгоритма, обсуждавшегося при изучении целых чисел). Ясно, что время работы этого процесса — $O[nL^2(m_1 m_2 \dots m_k)]$.

3.2

Наибольшие общие делители полиномов над полем

В этом разделе мы ограничиваем наш анализ полиномами над полем. Глава 5 целиком посвящена вычислению наибольших общих делителей полиномов над кольцом целых чисел — гораздо более захватывающему сюжету (большой интерес представляет книга Нетто (Netto, 1896)).

3.2.1. Делимость полиномов

Мы начнем со следующего определения:

Определение 3.2.1. *Евклидова область* — это область целостности J вместе с функцией «степени» (или «порядка») $d : J \setminus 0 \to \mathbb{N}$, такой, что

1. $d(p_1 p_2) \geq d(p_1) \quad (p_1, p_2 \neq 0)$.
2. Для любых элементов p_1 и p_2 из J ($p_2 \neq 0$) в J существуют элементы q и r, обладающие евклидовым свойством $p_1 = p_2 q + r$, $d(r) < d(p_2)$ или $r = 0$.

Пример (евклидовы области). $J = \mathbb{Z}$ с $d(p) = |p|$ — евклидова область, потому что для $p_1, p_2 \in Z$, $p_2 \neq 0$, существуют q, r, такие, что

$$p_1 = p_2 q + r, \quad 0 \leq r < |p_2|.$$

Отметим, что если $r \neq 0$, то $r - p_2$ также обладает свойством евклидовости; единственность здесь получается требованием неотрицательности r. Кроме того, если J — *поле*, то $J[x]$ с $d[p(x)] = \deg[p(x)]$ — евклидова область, потому что в предыдущем разделе мы видели, что для любых $p_1(x), p_2(x) \in J[x]$, $p_2(x) \neq 0$, в $J[x]$ существуют единственные полиномы $q(x), r(x)$, такие, что

$$p_1(x) = p_2(x)q(x) + r(x), \quad \deg[r(x)] < \deg[p_2(x)].$$

Пример (неевклидовы области). $J = \mathbb{Q}$, поле рациональных чисел, с $d(p) = |p|$ не является евклидовой областью, потому что $d[5 \cdot (1/5)] = d(1) < d(5)$ и первое условие определения 3.2.1 не выполняется. Кольцо $\mathbb{Z}[x]$ с $d[p(x)] = \deg[p(x)]$ также не является евклидовой областью, потому что если мы разделим, например, $7x^5 + 4x^3 + 2x + 1$ на $5x^3 + 2$, то частное не принадлежит кольцу $\mathbb{Z}[x]$.

Определение 3.2.2. Пусть J — область целостности и $p_1(x)$, $p_2(x) \in J[x]$, $p_2(x) \neq 0$. Полином $p_h(x)$ из $J[x]$ называется *наибольшим общим делителем* полиномов $p_1(x)$ и $p_2(x)$, что обозначается $p_h(x) = \gcd[p_1(x), p_2(x)]$, если выполняются следующие условия:

a. $p_h(x)|p_1(x)$ и $p_h(x)|p_2(x)$.

b. Если $q(x)|p_1(x)$ и $q(x)|p_2(x)$, то $\deg[q(x)] \leq \deg[p_h(x)]$ и $q(x)|p_h(x)$.

Ниже мы сосредоточим наше внимание на полиномах от одной переменной. Изучение полиномов от многих переменных мы опускаем, потому что с помощью техники вычисления значений и интерполяции, которую мы излагали в предыдущем разделе, оно сводится к изучению полиномов от одной переменной.

Мы можем найти наибольший общий делитель двух полиномов $p_1(x)$, $p_2(x)$ в $J[x]$, где J — область целостности и $p_2(x) \neq 0$, используя несколько раз теорему о делимости (теорема 3.1.1). Соответствующий процесс назван *алгоритмом Евклида для полиномов* и работает следующим образом:

$$\begin{aligned}
&p_1(x) = p_2(x)q_1(x) + p_3(x), && \deg[p_3(x)] < \deg[p_2(x)],\\
&p_2(x) = p_3(x)q_2(x) + p_4(x), && \deg[p_4(x)] < \deg[p_3(x)],\\
&\ldots && \ldots\\
&p_{h-2}(x) = p_{h-1}(x)q_{h-2}(x) + p_h(x), && \deg[p_h(x)] < \deg[p_{h-1}(x)],\\
&p_{h-1}(x) = p_h(x)q_{h-1}(x) + 0.
\end{aligned}$$

Поскольку $\deg[p_i(x)] < \deg[p_{i-1}(x)]$ для $i = 3, 4, \ldots, h$, гарантировано, что эта последовательность делений заканчивается после самое большее $\deg[p_2(x)]$ шагов.

Теорема 3.2.3. Пусть J — область целостности и $p_1(x), p_2(x) \in J[x]$, $p_2(x) \neq 0$. В описанном выше алгоритме Евклида для полиномов последний отличный от нуля остаток $p_h(x)$ — это наибольший общий делитель полиномов $p_1(x)$ и $p_2(x)$.

Доказательство. Из приведенных выше рассуждений видно, что любой делитель полиномов $p_2(x)$ и $p_3(x)$ является также делителем полинома $p_1(x)$, а любой делитель $p_1(x)$ и $p_2(x)$ — делителем $p_3(x)$. Поэтому общие делители полиномов $\{p_1(x), p_2(x)\}$ — также общие делители полиномов $\{p_2(x), p_3(x)\}$, и, следовательно, $\gcd[p_1(x), p_2(x)] = \gcd[p_2(x), p_3(x)]$. Продолжая таким же образом, имеем:

$$\gcd[p_1(x), p_2(x)] = \gcd[p_2(x), p_3(x)] = \cdots = \gcd[p_{h-1}(x), p_h(x)] = p_h(x).$$

□

Последовательность остатков полиномов, полученная при выполнении алгоритма Евклида, называется *последовательностью полиномиальных остатков* (PRS).

Следует, однако, заметить, что бессмысленно (в общем случае) говорить о «единственном» наибольшем общем делителе двух полиномов, поскольку в алгебраической системе J может быть много обратимых элементов, т.е. если $p_h(x)$ — наибольший общий делитель полиномов $p_1(x)$ и $p_2(x)$, то им является и $ap_h(x)$, если a – обратимый элемент, и, обратно, если $p_h(x)$ и $p_m(x)$ — два наибольших общих делителя одних и тех же полиномов $p_1(x), p_2(x)$, то $p_h(x) = ap_m(x)$ для некоторого обратимого элемента a.

Мы будем говорить, что два полинома $p(x)$ и $q(x)$ *ассоциированы*, если каждый из них является скалярным кратным другого. Любой полином ассоциирован ровно с одним нормированным полиномом; поэтому, когда $p_h(x)$ нормирован, мы можем говорить о единственном наибольшем общем делителе. (Именно это мы имеем в виду, когда несколько небрежно пользуемся терминологией.)

Стоит упомянуть, что в $\mathbb{Z}$ наибольший общий делитель двух целых чисел не единствен, если мы определяем его как «наибольший по абсолютной величине»; например, числа 6 и 9 будут иметь два наибольших по абсолютной величине общих делителя: 3 и -3.

Два полинома в $J[x]$ называются *взаимно простыми*, если любой их наибольший общий делитель — обратимая константа из J. В этом случае мы будем говорить, что единичный элемент кольца J — их наибольший общий делитель.

Ниже мы исследуем алгоритм Евклида для полиномов над полем — относительно простая процедура. Напротив, вычисление наибольшего общего делителя полиномов $p_1(x)$ и $p_2(x)$ в $\mathbb{Z}[x]$ может значительно усложняться главным образом потому, что $\mathbb{Z}[x]$ — *не* евклидова область. При попытке вычислять полиномиальный gcd в $\mathbb{Z}[x]$ коэффициенты полиномов в последовательности остатков могут становиться очень большими, и это замедляет вычисления. Мы будем в гл. 5 исследовать способы обхода этой трудности.

3.2.2. Алгоритм Евклида для полиномов над полем

Пусть теперь J — *поле*, и пусть $p_1(x)$ и $p_2(x) \neq 0$ — два полинома в $J[x]$. Как мы уже видели, повторным применением алгоритма деления **PDF**, описанного в разд. 3.1.1, можно легко вычислить наибольший общий делитель полиномов $p_1(x)$ и $p_2(x)$.

Если $p_h(x)$ — наибольший общий делитель полиномов $p_1(x)$ и $p_2(x)$, то ясно, что $p_h(x)$ — делитель каждого полинома из множества

$$p_1(x)v(x) + p_2(x)u(x),$$

где $v(x)$ и $u(x)$ — произвольные полиномы из $J[x]$. Возникает вопрос, принадлежит ли этому множеству сам $p_h(x)$, т.е. можно ли найти два полинома $f(x)$ и $g(x)$ в $J[x]$, такие, что

$$p_1(x)g(x) + p_2(x)f(x) = p_h(x).$$

Имеет место следующая

Теорема 3.2.4. Пусть J — поле, и рассмотрим полиномы $p_1(x)$, $p_2(x) \neq 0 \in J[x]$. Если $p_h(x) = \gcd[p_1(x), p_2(x)]$, то в $J[x]$ существуют два полинома $u(x), v(x)$, такие, что

$$p_1(x)v(x) + p_2(x)u(x) = p_h(x). \qquad \text{(F)}$$

Доказательство. Из всех полиномов вида (F), не равных тождественно нулю, выберем полином наименьшей степени и обозначим его $p_h(x)$. Если $p_h(x)$ не делит $p_1(x)$, то по теореме 3.1.1 мы имеем $p_1(x) = p_h(x)q(x) + r(x)$, $r(x) \neq 0$ и $\deg[r(x)] < \deg[p_h(x)]$. Но тогда полином $r(x) = p_1(x) - p_h(x)q(x) = p_1(x) - \{p_1(x)v(x) + p_2(x)u(x)\}q(x) = p_1(x)\{1 - v(x)q(x)\} - p_2(x)\{u(x)q(x)\}$ имеет вид (F), в противоречие с выбором $p_h(x)$. □

Следствие 3.2.5. Необходимым и достаточным условием, чтобы два полинома $p_1(x)$ и $p_2(x)$ из $J[x]$, J — поле, были взаимно просты, является существование двух полиномов $v(x), u(x)$, таких, что

$$p_1(x)v(x) + p_2(x)u(x) = 1.$$

Полиномы $u(x)$ и $v(x)$ в теореме 3.2.4 не единственны. Действительно, если $u(x) = f(x)$ и $v(x) = g(x)$ удовлетворяют требованиям теоремы, то им удовлетворяют и полиномы

$$u(x) = f(x) - t(x)p_1(x), \quad v(x) = g(x) + t(x)p_2(x),$$

где $t(x)$ — произвольный полином из $J[x]$. (Проверьте это непосредственной подстановкой.) Поэтому можно выбрать $u(x)$ и $v(x)$ произвольно высокой степени. Однако для их степеней имеются ограничения снизу.

Теорема 3.2.6. Пусть J — поле, и рассмотрим полиномы $p_1(x)$ и $p_2(x)$ из $J[x]$. Если $p_h(x) = \gcd[p_1(x), p_2(x)]$, то в $J[x]$ существуют два единственных полинома $f(x)$ и $g(x)$, степени которых меньше степеней полиномов $p_1(x)$ и $p_2(x)$ соответственно, такие, что

$$p_1(x)g(x) + p_2(x)f(x) = p_h(x).$$

Доказательство. Для конструктивного доказательства см. ниже расширенный алгоритм Евклида. □

Все приведенные выше результаты справедливы также для полиномов с коэффициентами из области целостности с единицей при условии, что можно применять алгоритм **PDF**.

XEA-P. Расширенный алгоритм Евклида для полиномов над полем (E**x**tended **E**uclidean **A**lgorithm for **P**olynomials over a Field)

Вход: $p_1(x), p_2(x) \in J[x]$, $p_2(x) \neq 0$, $m = \deg[p_1(x)] \geq \deg[p_2(x)] = n$; J — поле.

Выход: $p_h(x), f(x), g(x) \in J[x]$, такие, что $\deg[f(x)] < \deg[p_1(x)] - \deg[p_h(x)]$, $\deg[g(x)] < \deg[p_2(x)] - \deg[p_h(x)]$ и $p_h(x) = \gcd[p_1(x), p_2(x)] = p_1(x)g(x) + p_2(x)f(x)$.

1. [Инициализация] $[p_0(x), p_1(x)] := [p_1(x), p_2(x)]$;
$[g_0(x), g_1(x)] := (1, 0)$;
$[f_0(x), f_1(x)] := (0, 1)$.

2. [Основной цикл] Пока $p_1(x) \neq 0$ выполнять
$\{q(x) := \mathbf{PDF}[p_0(x), p_1(x)]$;
$[p_0(x), p_1(x)] := [p_1(x), p_0(x) - p_1(x)q(x)]$;
$[g_0(x), g_1(x)] := [g_1(x), g_0(x) - g_1(x)q(x)]$;
$[f_0(x), f_1(x)] := [f_1(x), f_0(x) - f_1(x)q(x)]\}$.

3. [Выход] Вернуть $[p_h(x), g(x), f(x)] := [p_0(x), g_0(x), f_0(x)]$.

Анализ времени работы алгоритма ***XEA-P***. Ясно, что время работы этого алгоритма доминируется временем выполнения шага 2.

Поскольку мы работаем в поле, первое выполнение алгоритма **PDF** требует времени $O[n(m - n + 1)]$, где $m = \deg[p_1(x)]$, $n = \deg[p_2(x)]$, $m \geq n$, и $m - n = \deg[q(x)]$; кроме того, первое выполнение каждого из полиномиальных умножений $p_1(x)q(x)$, $g_1(x)q(x)$ и $f_1(x)q(x)$ также происходит за время $O[n(m - n + 1)]$, и оно явно доминирует время выполнения каждого из соответствующих полиномиальных вычитаний. Поэтому время первого выполнения шага 2 равно $O[n(m-n+1)]$, что доминирует и время всех его последующих выполнений (проверьте это).

Итак, мы можем сказать, что в худшем случае каждое выполнение шага 2 происходит за время $O[n(m-n+1)]$, а поскольку может быть не более n выполнений этого шага, мы имеем

$$t_{\textbf{XEA-P}}[p_1(x), p_2(x)] = O[n^2(m-n+1)].$$

Пример. Рассмотрим поле $\mathbb{Z}_{11}$ и полиномы $p_1(x) = 7x^5 + 4x^3 + 2x + 1$ и $p_2(x) = 5x^3 + 2$ над этим полем. Применяя **XEA-P** к $p_1(x)$ и $p_2(x)$, получаем следующую таблицу (работая с множеством неотрицательных целых чисел):

Итерация	$q(x)$	$p_0(x)$	$p_0(x)$	$g_0(x)$	$g_0(x)$	$f_0(x)$	$f_1(x)$
0	—	$7x^7+4x^3+2x+1$	$5x^3+2$	1	0	0	1
1	$8x^2+3$	$5x^3+2$	$6x^2+2x+6$	0	1	1	$3x^2+8$
2	$10x+4$	$6x^2+2x+6$	$9x$	1	$x+7$	$3x^2+8$	$3x^3+10x^2+8x+2$
3	$8x+10$	$9x$	6	$x+7$	$3x^2+8$	$3x^3+10x^2+8x+2$	$9x^4+4x^2+3x+10$
4	$7x$	6	0	$3x^2+8$	—	$9x^4+4x^2+3x+10$	—

Проверка: $6 = (7x^5 + 4x^3 + 2x + 1)(3x^2 + 8) + (5x^3 + 2)(9x^4 + 4x^2 + 3x + 10)$.

Эти вычисления показывают, что 6 — наибольший общий делитель двух исходных полиномов. Как мы уже отмечали, каждый отличный от нуля элемент поля обратим, следовательно, любое ненулевое кратное наибольшего общего делителя — также наибольший общий делитель. Поэтому удобно разделить результат на его старший коэффициент и именно полученный нормированный полином назвать *наибольшим общим делителем* двух данных полиномов. Соответственно таким наибольшим общим делителем в приведенном выше примере будет 1, а не 6.

Если $m(x)$ — неприводимый полином в $\mathbb{Z}_p[x]$, p — простое число, то мы можем воспользоваться приведенным выше алгоритмом так же, как при работе с целыми числами, чтобы вычислить обратный к полиному $p(x) \neq 0$ в $\mathbb{Z}_p[x]_{m(x)}$, где $\deg[p(x)] < \deg[m(x)]$. Мы просто применяем **XEA-P** к $m(x)$ и $p(x)$ и получаем полиномы $f(x)$ и $g(x)$, такие, что

$$m(x)f(x) + p(x)g(x) = 1.$$

Тогда, вычисляя значение последнего выражения в точке α, где $\alpha = [x]_{m(x)}$ и $m(\alpha) = 0$, мы получаем $p(\alpha)g(\alpha) = 1$ в $\mathbb{Z}_p[x]_{m(x)}$, откуда следует, что $p(\alpha)$ обратим в $Z_p[x]_{m(x)}$.

Рассмотрим теперь другой пример, и сконцентрируем наше внимание на росте коэффициентов членов последовательности полиномиальных остатков.

Пример. Рассмотрим полиномы $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$. Применяя алгоритм Евклида над рациональными числами, получаем такие последовательности:

$$\begin{aligned}
&p_1(x) = x^3 - 7x + 7, && \\
&p_2(x) = 3x^2 - 7, && q_1(x) = (1/3)x, \\
&p_3(x) = (-14/3)x + 7, && q_2(x) = (-9/14)x - 27/28, \\
&p_4(x) = -1/4, && q_3(x) = (56/3)x - 28, \\
&p_5(x) = 0. &&
\end{aligned}$$

Как и прежде, $1 = \gcd[p_1(x), p_2(x)]$.

Рост коэффициентов последовательности полиномиальных остатков может быть минимизирован, если каждый член, как только он получен, нормируется. В этом случае мы получаем

$$\begin{aligned}
&p_1(x) = x^3 - 7x + 7, && \\
&p_2(x) = x^2 - 7/3, && q_1(x) = x, \\
&p_3(x) = x - 3/2, && q_2(x) = x + 3/2, \\
&p_4(x) = 1, && q_3(x) = x - 3/2, \\
&p_5(x) = 0. &&
\end{aligned}$$

Действительно, мы видим, что цель достигнута, но ценой вычислений gcd целых чисел на каждом шаге, чтобы максимально редуцировать дроби.

Из этого примера видно, что использовать арифметику рациональных чисел для вычисления последовательности полиномиальных остатков нецелесообразно; с одной стороны, число требуемых для максимального редуцирования коэффициентов вычислений gcd целых чисел слишком велико и, с другой стороны, отказ от редукции ведет в стремительному росту выражения.

3.2.3. Неприводимые сомножители полиномов

Неприводимые полиномы играют ту же роль, что и простые числа в теории разложения на множители целых чисел, поэтому исследуем некоторые их свойства. Мы должны знать, какие полиномы неприводимы в $J[x]$, когда $J = \mathbb{C}$, $J = \mathbb{R}$, $J = \mathbb{Q}$ и $J = \mathbb{Z}$; случай $J = \mathbb{Z}_p$, p — простое число, будет рассмотрен отдельно в разд. 3.3 и гл. 6. (В

гл. 6 мы также исследуем, как вычислять неприводимые сомножители данного полинома с целыми коэффициентами, — весьма нелегкая задача.)

Мы знаем, что поле $\mathbb{C}$ было изобретено, чтобы включать корни неприводимых полиномов из $\mathbb{R}[x]$; мы видели, что $i = \sqrt{-1}$ — корень полинома $x^2 + 1$. Из следующей теоремы мы заключаем, что единственные отличные от констант неприводимые полиномы в $\mathbb{C}[x]$ — полиномы степени 1.

Основная теорема алгебры. Каждый полином $p(x)$ из $\mathbb{C}[x]$ степени ≥ 1 имеет корень в $\mathbb{C}$.

Доказательство. Имеются различные доказательства этой теоремы, принадлежащие разным знаменитым математикам. Эти доказательства можно найти в большинстве учебников по алгебре [см., например, (Childs, 1979)]. □

С помощью основной теоремы алгебры мы можем теперь легко определять неприводимые полиномы в $\mathbb{R}[x]$. Однако отметим, что знание того, какие полиномы в $\mathbb{R}[x]$ (или в $\mathbb{C}[x]$) неприводимы, не облегчает разложение данного полинома.

Теорема 3.2.7. Отличный от константы полином $p(x)$ из $\mathbb{R}[x]$ неприводим, если и только если либо **(a)** $p(x)$ имеет степень 1, либо **(b)** $p(x) = ax^2 + bx + c$ и $b^2 - 4ac < 0$.

Доказательство. Очевидно, что любой полином степени 1 неприводим.

Предположим теперь, что $p(x) = ax^2 + bx + c$. Тогда $p(x)$ разлагается в $\mathbb{C}[x]$ на множители:

$$p(x) = a\left[x + \frac{b + \sqrt{b^2 - 4ac}}{2a}\right]\left[x + \frac{b - \sqrt{b^2 - 4ac}}{2a}\right].$$

Мы знаем, что если $b^2 - 4ac < 0$, то корни полинома $p(x)$ комплексные, а значит, $p(x)$ неприводим в $\mathbb{R}[x]$. Таким образом, мы доказали, что два типа полиномов, которые по нашему утверждению неприводимы в $\mathbb{R}[x]$, действительно неприводимы.

Докажем теперь обратное утверждение теоремы. Предположим, что $p(x)$ — неприводимый в $\mathbb{R}[x]$ полином, $\deg[p(x)] > 1$. Следовательно, у $p(x)$ нет вещественных корней. Однако по основной теореме алгебры у него есть комплексный корень $\rho = a + ib$, где a, b — вещественные числа, $b \neq 0$. Образуем теперь полином второй степени

$t(x) = (x-\rho)(x-\bar{\rho})$, где $\bar{\rho} = a - ib$, и, разделив $p(x)$ на $t(x)$, получим $p(x) = t(x)q(x) + r(x)$, где $q(x), t(x) \in \mathbb{R}[x], \deg[r(x)] < \deg[t(x)]$. Ясно, что $r(x)$ — полином степени ≤ 1, т.е. $r(x) = cx + d$. Рассматриваем равенство $p(x) = t(x)q(x) + r(x)$ как равенство функций на $\mathbb{C}$ и полагаем $x = \rho$. Получаем $0 = p(\rho) = t(\rho)q(\rho) + r(\rho)$; $t(\rho) = 0$ по построению и, значит, $r(\rho) = 0$. Однако, поскольку ρ — не вещественное число, равенство $r(\rho) = 0$ может иметь место, только если $c = d = 0$, а тогда $r(x) = 0$. Таким образом, $p(x) = t(x)q(x)$; значит, $p(x)$ не является неприводимым, если $q(x)$ — не константа. Но если $q(x)$ — константа, то $p(x)$ — неприводимый полином степени 2, что и требовалось доказать. □

Сформулируем теперь некоторые результаты, имеющие место для полиномов над полем.

Теорема 3.2.8. Пусть J — поле и $p_1(x), p_2(x) \in J[x]$. Если неприводимый полином $m(x) \in J[x]$ делит произведение $p_1(x)p_2(x)$, то $m(x)$ должен делить или $p_1(x)$, или $p_2(x)$.

Доказательство. Если хотя бы один из полиномов $p_1(x)$ и $p_2(x)$ тождественно равен нулю, то результат очевиден. Если $p_1(x)p_2(x) \neq 0$, то предположим, что $m(x)$ не делит $p_1(x)$, и покажем, что $m(x)|p_2(x)$. По предположению $\gcd[m(x), p_1(x)] = 1$, и по следствию 3.2.5 существуют полиномы $v(x)$ и $u(x)$, такие, что

$$m(x)u(x) + p_1(x)v(x) = 1.$$

Умножая на $p_2(x)$, получаем

$$p_2(x)m(x)u(x) + p_2(x)p_1(x)v(x) = p_2(x).$$

Поскольку $m(x)$ делит левую часть уравнения, мы заключаем, что $m(x)|p_2(x)$. □

Теорема 3.2.9. (теорема о разложении на простые множители для полиномов). Пусть J — поле и $p(x) \in J[x]$, $\deg[p(x)] > 0$. Тогда полином $p(x)$ может быть однозначно разложен в произведение неприводимых нормированных полиномов над $J[x]$, т.е. $p(x) = cp_1(x)p_2(x)\ldots p_k(x)$, $p_i(x) \in J[x]$, $i = 1, 2, \ldots, k$ и $c \in J$. Разложение является единственным с точностью до порядка сомножителей.

Доказательство. Доказательство проведем индукцией по степени полинома $p(x)$. Очевидно, что если $\deg[p(x)] = 1$, то теорема верна, поскольку полином $p(x)$ неприводим. Предположим, что $p(x)$ имеет два

разложения на простые множители, т.е. $p(x) = cp_1(x)p_2(x)\ldots p_k(x) = dq_1(x)q_2(x)\ldots q_j(x)$. Согласно теореме 3.2.8, $p_1(x)$ делит некоторый $q_i(x)$. Поскольку и $p_1(x)$, и $q_i(x)$ нормированы и неприводимы, мы имеем $p_1(x) = q_i(x)$. Поэтому мы можем написать

$$p_1'(x) = \frac{p(x)}{p_1(x)} = cp_2(x)\ldots p_k(x) = dq_1(x)\ldots q_{i-1}(x)q_{i+1}(x)\ldots q_j(x).$$

По предположению индукции разложение на простые множители полинома $p_1'(x)$ единственно с точностью до порядка сомножителей, т.е. каждый $p_i(x)$ равен некоторому $q_k(x)$, и наоборот. Следовательно, разложение на простые множители полинома $p(x)$ должно также быть единственным с точностью до порядка сомножителей. □

Так же как для целых чисел, мы можем записать разложение на множители полинома $p(x)$ в $\mathbb{R}[x]$ в виде

$$p(x) = [p_1(x)]^{e_1}[p_2(x)]^{e_2}\ldots[p_k(x)]^{e_k}.$$

Если какое-либо из чисел e_i больше единицы, то мы будем говорить, что у полинома $p(x)$ есть *кратный сомножитель*. Например, $p(x) = (x-1)^3(x+1)$ имеет кратный сомножитель, а $p(x) = (x-1)(x+1)$ таковых не имеет. В первом случае мы говорим, что у $p(x)$ есть *кратный* корень в J, а во втором — что $p(x)$ имеет только *простые* корни. Мы видим, таким образом, что разложение на множители в $\mathbb{R}[x]$ или в $\mathbb{C}[x]$ эквивалентно нахождению корней полинома. Имеет место следующая

Теорема 3.2.10. (Гаусс). Пусть J — область целостности и $p(x) \in J[x]$, $\deg[p(x)] > 0$. Тогда полином $p(x)$ может быть единственным образом разложен в произведение неприводимых нормированных полиномов над $J[x]$ при условии, что каждый элемент в кольце J может быть единственным образом разложен в произведение неразложимых элементов.

Доказательство. Доказательство достаточно длинное, и мы его опускаем; детали см. в книге (Sims, 1984, pp. 229–234). □

Следствием теоремы 3.2.10 является тот факт, что $\mathbb{Z}[x]$ — *область с однозначным разложением на множители*, хотя она не является евклидовой.

В качестве примера области, не являющейся областью с однозначным разложением на множители, рассмотрим область целостности,

состоящую из чисел вида $a+b\sqrt{-5}$, a, b – целые числа. (Проверьте, что это действительно область целостности.) Число 21 имеет два разложения на неразложимые множители: $21 = 3\cdot 7 = (1+2\sqrt{-5})(1-2\sqrt{-5})$.

Сформулируем общий результат.

Теорема 3.2.11. Если J — евклидова область, то она является областью с однозначным разложением на множители, т.е. каждый ее отличный от нуля элемент или обратим, или может быть представлен в виде конечного произведения неразложимых элементов.

Доказательство. Доказательство достаточно длинное, и мы его опускаем; детали см. в книге (Sims, 1984, pp. 214–221). □

Мы продолжаем обсуждение неприводимых над $\mathbb{Q}[x]$ полиномов. В отличие от $\mathbb{R}[x]$ или $\mathbb{C}[x]$, где мы могли явно описать все неприводимые полиномы, в $\mathbb{Q}[x]$ мы можем дать только некоторые достаточные критерии неприводимости (по причинам, которые станут понятными ниже). Основной пункт нашего обсуждения состоит в том, что разложение на множители в $\mathbb{Q}[x]$ — это то же самое, что разложение на множители в $\mathbb{Z}[x]$.

Пусть $p(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0$ — полином с рациональными коэффициентами. Мы можем умножить $p(x)$ на t, наименьшее общее кратное знаменателей коэффициентов, и получить полином $tp(x) = s(x)$ с целыми коэффициентами. Поскольку полиномы $p(x)$ и $s(x)$ ассоциированы, $s(x)$ неприводим в $\mathbb{Q}[x]$, если и только если неприводим $p(x)$. Поэтому, изучая полиномы в $\mathbb{Q}[x]$, мы можем всегда предполагать, что их коэффициенты — целые числа.

Мы называем полином $p(x) \in \mathbb{Q}[x]$ *примитивным*, если его коэффициенты — целые числа и их наибольший общий делитель равен 1. Тогда любой полином в $\mathbb{Q}[x]$ с целыми коэффициентами ассоциирован с примитивным полиномом. [Чтобы в этом убедиться, рассмотрим полином $p(x)$ в $\mathbb{Q}[x]$ с целыми коэффициентами, и пусть d — наибольший общий делитель его коэффициентов. Тогда $(1/d)p(x)$ — все еще полином с целыми коэффициентами, но теперь наибольший общий делитель его коэффициентов — единица. Следовательно, $p(x)$ и $(1/d)p(x)$ ассоциированы в $\mathbb{Q}[x]$.]

Теорема 3.2.12. Произведение двух примитивных полиномов из $\mathbb{Q}[x]$ снова является примитивным полиномом.

Доказательство. Ясно, что произведение двух полиномов с целыми коэффициентами — снова полином с целыми коэффициентами.

Пусть $p(x)$ и $q(x)$ — два примитивных полинома. По определению примитивности для любого простого числа p

$$p(x) \neq 0 \pmod{p}, \quad q(x) \neq 0 \pmod{p}.$$

Поэтому

$$p(x)q(x) \neq 0 \pmod{p}$$

для любого простого p, откуда следует, что никакое простое число не делит все коэффициенты полинома $p(x)q(x)$. Следовательно, наибольший общий делитель коэффициентов полинома $p(x)q(x)$ равен 1 и полином $p(x)q(x)$ примитивен. □

Теорема 3.2.13. (Гаусс). Пусть $p(x)$ — полином из $\mathbb{Q}[x]$ с целыми коэффициентами. Если $p(x) = q(x)r(x)$ в $\mathbb{Q}[x]$, то $p(x) = q_1(x)r_1(x)$, где $q_1(x)$ и $r_1(x)$ — полиномы с целыми коэффициентами, ассоциированные с $q(x)$ и $r(x)$ соответственно.

Доказательство. Без потери общности предположим, что $p(x)$ примитивен и $p(x) = q(x)r(x)$ в $\mathbb{Q}[x]$. Тогда из приведенных рассуждений видно, что существуют рациональные числа a и b, такие, что $a \cdot q(x)$ и $b \cdot r(x)$ — примитивные полиномы. По предыдущей теореме полином $a \cdot b \cdot q(x)r(x) = a \cdot b \cdot p(x)$ примитивен. Но примитивен и $p(x)$, и, пользуясь тем, что если r — рациональное число, такое, что $r \cdot p(x)$ и $p(x)$ — примитивные полиномы, то $r = 1$ или -1, получаем $p(x) = \pm a \cdot b \cdot q(x)r(x)$. Для завершения доказательства положим $q_1(x) = \pm a \cdot q(x)$ и $r_1(x) = b \cdot r(x)$. □

Мы говорим, что полином в $\mathbb{Z}[x]$ *неприводим*, если он не разлагается в произведение двух полиномов степеней ≥ 1 с целыми коэффициентами. В силу теоремы 3.2.13 мы видим, что полином неприводим в $\mathbb{Z}[x]$, если и только если он неприводим как полином в $\mathbb{Q}[x]$. Следующая теорема помогает нам в вопросе о неприводимости.

Теорема 3.2.14. Если $p(x) = c_n x^n + \cdots + c_1 x + c_0$ — полином в $\mathbb{Z}[x]$ и r/s — его корень, такой, что $(r, s) = 1$, то $s|c_n$ и $r|c_0$.

Доказательство. Так как r/s — корень полинома $p(x)$, то $c_n(r^n/s^n) + \cdots + c_1(r/s) + c_0 = 0$. Умножая на s^n, получаем $c_n r^n + c_{n-1} r^{n-1} s + \cdots + c_1 r s^{n-1} + c_0 s^n = 0$, откуда $c_n r^n = \lambda s$ для некоторого $\lambda \in \mathbb{Z}$. Следовательно, $s|c_n r^n$, и поскольку $(r, s) = 1$, то $s|c_n$. Аналогично $c_0 s^n = \mu r$ для некоторого $\mu \in \mathbb{Z}$, а так как $(r, s) = 1$, то $r|c_0$. □

Пример. Единственные возможные рациональные корни полинома $x^2 - 6x + 8$ суть $x = 1,\ -1,\ 2,\ -2,\ 4,\ -4,\ 8,\ -8$, поскольку они являются единственными делителями числа 8; действительно, два корня суть 2 и 4.

Заметим, что если r/s — корень полинома $p(x) = 0$, то $sx - r$ — линейный сомножитель полинома $p(x)$. Полезен следующий критерий для обнаружения неприводимых полиномов в $\mathbb{Z}[x]$.

Теорема 3.2.15. (критерий Эйзенштейна, 1850). Пусть $p(x) = c_0x^n + c_1x^{n-1} + c_2x^{n-2} + \cdots + c_{n-1}x + c_n$ — полином из $\mathbb{Z}[x]$. Если существует простое число p, такое, что p *не* делит c_0 и *делит* остальные целые коэффициенты $c_1, c_2, \ldots, c_n$, но p^2 не делит c_n, то полином $p(x)$ неприводим.

Доказательство. Будем доказывать от противного. Предположим, что $p(x) = (a_0x^j + a_1x^{j-1} + a_2x^{j-2} + \cdots + a_{j-1}x + a_j) \cdot (b_0x^k + b_1x^{k-1} + b_2x^{k-2} + \cdots + b_{k-1}x + b_k) = a_jb_k + (a_{j-1}b_k + a_jb_{k-1})x + \cdots + a_0b_0x^{j+k}$, где $j + k = n$; далее, положим $a_0 = b_0 = 1$. Тогда, поскольку $a_jb_k = c_n$, ровно одно из чисел a_j, b_k делится на p; пусть, например, $p|a_j$ и $(b_k, p) = 1$. Кроме того, так как $p|c_{n-1}$, где $c_{n-1} = a_{j-1}b_k + a_jb_{k-1}$, то $p|a_jb_{k-1}$ и $p|a_{j-1}b_k$; однако, поскольку $(b_k, p) = 1$, мы будем иметь $p|a_{j-1}$. Таким же образом мы покажем, что p делит $a_{j-2}, a_{j-3}, \ldots, a_1$, а также коэффициент a_0, который равен 1. Однако это противоречие, и, следовательно, полином $p(x)$ неприводим. □

Теорема 3.2.15 верна также, когда коэффициенты полинома $p(x)$ принадлежат области целостности, которая является областью с однозначным разложением на множители.

Пример. Согласно теореме 3.2.15, полином $x^n - 2$ неприводим при любом n.

Теорема 3.2.15 показывает, что в $\mathbb{Q}[x]$ имеются неприводимые полиномы любой степени. Заметим также, что имеются полиномы, к которым критерий Эйзенштейна неприменим; например, для полинома $p(x) = x^2 - 6x + 7$ критерий Эйзенштейна абсолютно бесполезен, но тем не менее полином $p(x)$ неприводим.

Пример. В $\mathbb{Q}[x]$ полином $p(x) = x^3 - 2$ неприводим. В $\mathbb{R}$ он имеет корень, а именно $2^{1/3}$, и

$$x^3 - 2 = (x - 2^{1/3})(x^2 + 2^{1/3}x + 4^{1/3}),$$

где второй сомножитель имеет два комплексных корня.

Другой тест неприводимости состоит в редукции полинома по модулю m, т.е. вычисляем $p^1(x) \equiv p(x) \pmod{m}$, где m не делит старший коэффициент полинома $p(x)$, и проверяем $p^1(x)$ на неприводимость. [Заметим, что $\deg[p^1(x)] = \deg[p(x)]$.] Проверка неприводимости полинома $p^1(x)$ в $\mathbb{Z}_m[x]$ — конечная задача (поскольку имеется только конечное число возможных делителей); в гл. 6 мы детально разбираем метод разложения полиномов в конечных полях.

Имеет место следующая

Теорема 3.2.16. Если $p^1(x) \equiv p(x) \pmod{m}$ для некоторого m, не делящего старший коэффициент полинома $p(x)$, и полином $p^1(x)$ неприводим в $Z_m[x]$, то $p(x)$ неприводим в $\mathbb{Q}[x]$.

Доказательство. Предположим, что $p(x) = q(x)r(x)$, где $q(x)$ и $r(x)$ — примитивные полиномы с целыми коэффициентами. Тогда для любого m, не делящего старший коэффициент полинома $p(x)$, мы имеем $p^1(x) = q^1(x)r^1(x)$ в $\mathbb{Z}_m[x]$, так что $p^1(x)$ разлагается на множители. □

3.2.4. Разложение полиномов на свободные от квадратов множители

Полином $p(x)$ называется *свободным от квадратов*, если не существует полинома $q(x)$ положительной степени, такого, что $q^2(x)|p(x)$. Процесс нахождения свободных от квадратов сомножителей данного полинома широко используется в математике. Среди его приложений — разложение полиномов на множители, разложение на простейшие дроби и интегрирование рациональных функций; более того, в гл. 7 мы увидим, что решение полиномиального уравнения с кратными корнями может быть сведено к решению одного или нескольких уравнений, имеющих только простые корни, и эти уравнения — свободные от квадратов сомножители исходного уравнения.

Пусть J — произвольная числовая область; определим $p'(x) = D[p(x)]$, производную полинома $p(x) \in J[x]$, следующими двумя правилами: (1) $D(ax^n) = anx^{n-1}$ для $a \in J$ и $n \geq 0$; следует отметить, что $D(ax^n) = 0$, если $J = \mathbb{Z}_n$, и (2) $D[p(x) + q(x)] = D[p(x)] + D[q(x)]$. Напомним, что для производных имеет место известное правило произведения $D[p(x) \cdot q(x)] = p(x)D[q(x)] + D[p(x)]q(x)$.

Теорема 3.2.17. Пусть J — *область с однозначным разложением на множители* характеристики нуль, и пусть $p(x)$ — примитивный отличный от константы полином в $J[x]$. Пусть $p(x) =$

$[p_1(x)]^{e_1}[p_2(x)]^{e_2}\ldots[p_n(x)]^{e_n}$ — однозначное разложение полинома $p(x)$ на неприводимые сомножители и $p'(x)$ — его производная. Тогда

$$\gcd[p(x),p'(x)] = [p_1(x)]^{e_1-1}[p_2(x)]^{e_2-1}\ldots[p_n(x)]^{e_n-1}.$$

Доказательство. Пусть $q(x) = \prod_{2\le i\le n}[p_i(x)]^{e_i}$ и $r(x) = \gcd[p(x), p'(x)]$. Тогда $p(x) = q(x)[p_1(x)]^{e_1}$ и

$$p'(x) = [p_1(x)]^{e_1}q'(x) + e_1[p_1(x)]^{e_1-1}p_1'(x)q(x),$$

откуда следует, что $[p_1(x)]^{e_1-1}|r(x)$. Покажем методом от противного, что $[p_1(x)]^{e_1}$ не делит $r(x)$. Предположим, что $[p_1(x)]^{e_1}|r(x)$; тогда $[p_1(x)]^{e_1}|p'(x)$, откуда мы заключаем, что $[p_1(x)]^{e_1}$ делит $e_1[p_1(x)]^{e_1-1}p_1'(x)q(x)$. После сокращений в последнем соотношении мы получаем $p_1(x)|e_1p_1'(x)q(x)$; однако, поскольку полиномы $p_i(x)$ взаимно просты, $\gcd[p_1(x),q(x)] = 1$ и, стало быть, $p_1(x)|e_1p_1'(x)$. Это и есть нужное противоречие, поскольку из $p_1(x)|e_1p_1'(x)$ следует, что $\deg[p_1(x)] \le \deg[p_1'(x)]$. Таким образом степень $p_1(x)$ в $r(x)$ равна $e_1 - 1$, и из соображений симметрии мы получаем $r(x) = [p_1(x)]^{e_1-1}\ldots[p_n(x)]^{e_n-1}$, что и требовалось доказать. □

Из теоремы 3.2.17 мы заключаем, что если $\gcd[p(x),p'(x)] = 1$, то $p(x)$ не имеет кратных сомножителей, и наоборот. Справедливо также

Следствие 3.2.18. Простые корни полинома не являются корнями его производной.

Следствие 3.2.19. Пусть J — поле и $p(x)$ — неприводимый полином в $J[x]$, который делит $s(x) \in J[x]$. Тогда $[p(x)]^2|s(x)$, если и только если $p(x)|s'(x)$.

Доказательство. Поскольку $p(x)|s(x)$, мы можем записать $s(x) = p(x)q(x)$ и, следовательно, $s'(x) = p'(x)q(x) + p(x)q'(x)$. Значит, если $[p(x)]^2|s(x)$, то $p(x)|q(x)$, и очевидно, что $p(x)|s'(x)$. Обратно, если $p(x)|s'(x)$, то $p(x)|p'(x)q(x)$ и по теореме 3.2.8 $p(x)$ делит или $p'(x)$, или $q(x)$. Однако $\deg[p'(x)] < \deg[p(x)]$ и, следовательно, $p(x)|q(x)$, откуда вытекает, что $[p(x)]^2|s(x)$. □

Мы теперь готовы обсуждать алгоритм разложения на свободные от квадратов множители. Пусть $p(x)$ — примитивный полином положительной степени от одной переменной, определенный на J, области с однозначным разложением на множители. [Как мы видели, выбирая $p(x)$ примитивным, мы не ограничиваем общности.] Предположим, что $p(x) = [p_1(x)]^{e_1}[p_2(x)]^{e_2}\ldots[p_n(x)]^{e_n}$ — разложение $p(x)$

на неприводимые множители $p_i(x)$ положительной степени, так что $e_i > 0$ для каждого i, и пусть $e = \max(e_1, \ldots, e_n)$; для $1 \le i \le e$ положим

$$J_i = \{j : e_j = i\}, \quad s_i(x) = \prod_{j \in J_i} p_j(x).$$

Тогда, очевидно,

$$p(x) = \prod_{1 \le i \le e} [s_i(x)]^i,$$

что называется *разложением полинома $p(x)$ на свободные от квадратов множители*. [Заметим, что некоторые из полиномов $s_i(x)$ могут быть равны 1; $s_1(x)$ — это произведение всех линейных множителей, соответствующих простым корням, $s_2(x)$ — произведение всех сомножителей, соответствующих двойным корням, и т.д.] Полиномы $s_i(x)$ — это *свободные от квадратов сомножители* полинома $p(x)$; мы можем найти их с помощью теоремы 3.2.17, заметив, что

$$r(x) = \gcd[p(x), p'(x)] = \prod_{1 \le i \le n} [p_i(x)]^{e_i - 1} = \prod_{1 \le i \le e} [s_i(x)]^{i-1}$$

[$s_1(x)$ здесь не присутствует]. Тогда наибольший свободный от квадратов делитель полинома $p(x)$ равен

$$t(x) = p(x)/r(x) = \prod_{1 \le i \le n} p_i(x) = \prod_{1 \le i \le e} s_i(x),$$

и, следовательно,

$$v(x) = \gcd[r(x), t(x)] = \prod_{2 \le i \le e} s_i(x).$$

Поэтому $s_1(x) = t(x)/v(x)$, т.е. первый свободный от квадратов сомножитель полинома $p(x)$ может быть вычислен с помощью дифференцирования, вычисления gcd и деления. Повторяя процесс с $r(x)$ вместо $p(x)$, мы можем вычислить $s_2(x)$ как первый свободный от квадратов сомножитель полинома $r(x)$ и в конечном счете получить все свободные от квадратов сомножители полинома $p(x)$. Итак, мы имеем следующий алгоритм:

PSQFF. Разложение полиномов на свободные от квадратов множители (**P**olynomial **Sq**uare**f**ree **F**actorization)

Вход: $p(x)$ — примитивный полином положительной степени от одной переменной над областью J характеристики нуль с однозначным разложением на множители.

Выход: Полиномы $s_i(x)$ и число e, такие, что $p(x) = \prod_{1 \le i \le e} [s_i(x)]^i$ — разложение полинома $p(x)$ на свободные от квадратов множители.

1. [Инициализация] $r(x) := \gcd[p(x), p'(x)]; t(x) := p(x)/r(x); j := 1$.

2. [Конец?] Если $\deg[r(x)] = 0$, то $\{e := j;\ s_j(x) = t(x);$ выход;$\}$.

3. [Вычисление $s_j(x)$] $v(x) := \gcd[r(x), t(x)]; s_j(x) := t(x)/v(x)$.

4. [Обновление] $r(x) := r(x)/v(x);\ t(x) := v(x);\ j := j + 1;$ перейти к шагу 2.

Анализ времени работы алгоритма ***PSQFF***. Ясно, что время работы этого алгоритма доминируется вычислениями gcd, имеющими место на шаге 3.

Если $n = \deg[p(x)]$, то n ограничивает число выполнений цикла (состоящего из шагов 2, 3, и 4), в котором находится шаг 3. Кроме того, время, необходимое для вычисления $r(x) := \gcd[p(x), p'(x)]$ на шаге 1, — это верхняя граница времени каждого вычисления gcd на шаге 3. Имеют место:

Случай a. Полином $p(x)$ принадлежит $J[x]$, где J — поле. В этом случае $\gcd[p(x), p'(x)]$ вычисляется за время $O(n^2)$, и так как имеется n выполнений шага 3, то

$$t_{\mathbf{PSQFF}}[p(x)] = O(n^3).$$

Случай b. Полином $p(x)$ принадлежит $J[x]$, где $J = \mathbb{Z}$. В этом случае, как мы увидим в гл. 5, $\gcd[p(x), p'(x)]$ вычисляется за время $O\{n^5 L^2[|p(x)|_\infty]\}$, и снова, поскольку имеется n выполнений шага 3,

$$t_{\mathbf{PSQFF}}[p(x)] = O\{n^6 L^2[|p(x)|_\infty]\}.$$

Дополнительную информацию об алгоритме разложения на свободные от квадратов множители можно найти в книгах (Wang, Trager, 1979) и (Yun, 1976). Мы завершаем этот раздел примером, в котором неявно используется информация из гл. 5, а именно как вычислять $\gcd[p(x), p'(x)]$ в $\mathbb{Z}[x]$.

Пример. Найдем свободные от квадратов сомножители полинома $p(x) = x^5 - x^4 - 2x^3 + 2x^2 + x - 1$. Применяя алгоритм **PSQFF**, получаем при первом проходе: $r(x) = x^3 - x^2 - x + 1$, $t(x) = x^2 - 1$, $v(x) = x^2 - 1$ и $s_1(x) = 1$, что указывает на отсутствие линейных сомножителей. При втором проходе мы имеем $r(x) = x - 1$, $t(x) = x^2 - 1$, а значит, $v(x) = x - 1$ и $s_2(x) = x + 1$; это означает, что $(x+1)^2$ —

сомножитель исходного полинома. В начале третьего и последнего прохода мы имеем $r(x) = 1, t(x) = x-1$ и на шаге 2 видим, что степень полинома $r(x)$ равна нулю; следовательно, $s_3(x) := t(x) = x - 1$, т.е. $(x-1)^3$ — также сомножитель исходного полинома. Таким образом, $p(x) = (x+1)^2(x-1)^3$.

3.3

Поля Галуа $GF(p^r)$

Мы уже сталкивались с конечными полями порядка p, где p — простое число, например с классами вычетов по модулю p. Однако в многочисленных приложениях нам понадобятся числовые поля порядка p^r, и в этом разделе мы узнаем, как их построить и как производить в них вычисления. Сейчас читателю целесообразно заново просмотреть материал разд. 2.3.2, в частности теоремы с 2.3.17 по 2.3.21, и также разд. 3.1.4.

3.3.1. Основные факты о конечных полях

Из определения мы уже знаем, что каждое поле — область целостности; обратное утверждение в общем случае неверно, в чем можно убедиться на примере кольца $\mathbb{Z}$. Однако для конечных полей имеет место

Теорема 3.3.1. Каждая конечная область целостности — поле.

Доказательство. Пусть J — конечная область целостности. Если a, b — два элемента из J, $a \neq b$, то для всех ненулевых элементов c в J по правилу сокращения $ac \neq bc$ (см. также теорему 2.3.8). Поэтому $cJ = J$ и $cd = 1$ для некоторого $d \in J$, а это означает, что каждый ненулевой элемент области J имеет в ней мультипликативный обратный, и, значит, J — поле. □

Следующие две теоремы непосредственно вытекают из теорем гл. 2. Значения q будут определены позже теоремой 3.3.6; оно *не может* быть произвольным.

Теорема 3.3.2. Если F — поле из q элементов и a — любой его ненулевой элемент, то $a^{q-1} = 1$.

Доказательство. Ненулевые элементы поля F образуют абелеву группу порядка $q-1$ относительно умножения (см. также теорему 2.3.17). □

Следствие 3.3.3. Если F — поле из q элементов, то любой элемент $a \in F$ удовлетворяет уравнению $x^q - x = 0$.

Доказательство. Из теоремы 3.3.2 мы знаем, что все ненулевые элементы поля F удовлетворяют уравнению $x^{q-1} - 1 = 0$; нулевой элемент поля, 0, удовлетворяет уравнению $x = 0$. Поэтому все элементы поля удовлетворяют уравнению $x(x^{q-1} - 1) = x^q - x = 0$. □

Теорема 3.3.4. Пусть F — поле из q элементов и a — его произвольный ненулевой элемент. Если n — порядок элемента a, то $n|(q-1)$.

Доказательство. Если n не делит $q-1$, то мы можем найти k и r, такие, что $q-1 = kn+r$, где $0 < r < n$. Тогда $a^{q-1} = a^{kn+r} = a^{kn}a^r = (a^n)^k a^r = 1$; следовательно, $a^r = 1$, поскольку $a^{q-1} = a^n = 1$. Это, однако, невозможно, так как $0 < r < n$ и n — наименьшее число, такое, что $a^n = 1$. Поэтому $n|(q-1)$. □

В разд. 3.1.4 мы уже видели, что, отправляясь от $J[x] = \mathbb{Z}_2[x]$, мы построили новое поле, $\mathbb{Z}_2[x]/\equiv_{m(x)}$, где $m(x) = x^2 + x + 1$ — неприводимый полином над $\mathbb{Z}_2$. (Предположим временно, что для любого n существуют неприводимые над $\mathbb{Z}_p$ полиномы степени n; этот факт мы докажем позже в этом же разделе.) Это новое поле, которое является множеством классов полиномов по модулю $m(x)$, содержит четыре элемента $\{[0],[1],[x],[1+x]\}$ и обозначается также $GF(2^2) = GF(4)$. Кроме того, мы можем представлять элементы поля $GF(4)$ как полиномы с коэффициентами из $\mathbb{Z}_p$, вычисленные в точке $\alpha = [x]_{m(x)}$, рассматриваемой как корень полинома $m(x)$, т.е. это новое поле построено добавлением к $\mathbb{Z}_2$ единственного элемента, и, значит, является *простым расширением поля* $\mathbb{Z}_2$ и обозначается $\mathbb{Z}_2[\alpha]$. Имеет место следующая

Теорема 3.3.5. Пусть p — простое число и $m(x)$ — неприводимый полином степени r в поле $\mathbb{Z}_p[x]$. Тогда классы вычетов $\mathbb{Z}_p[x]/\equiv_{m(x)}$ образуют поле из p^r элементов, содержащее $\mathbb{Z}_p$ и корень полинома $m(x)$.

Доказательство. Доказательство фактически было дано при обсуждении примера в разд. 3.1.4. □

Теорема 3.3.6. Пусть F — поле из q элементов. Тогда $q = p^r$, где p — простое, а r — натуральное числа.

Доказательство. По определению F имеет единичный элемент относительно умножения; мы обозначим его 1. Очевидно, что $1 + 1$ принадлежит F, и мы обозначаем этот элемент 2. Мы продолжаем таким образом: $2 + 1 = 3 \in F$ и т.д., и после конечного числа шагов нам встречается элемент, который мы уже видели.

Следуя аргументам, представленным в конце разд. 2.1.3 (еще раз просмотрите их), предположим, что

$$\sum_{1 \le i \le k'} 1 = \sum_{1 \le i \le k''} 1,$$

где $k' < k''$; следовательно, $\sum_{1 \le i \le k'' - k'} 1 = 0$. Значит, должно существовать *наименьшее* целое число λ, такое, что $\sum_{1 \le i \le \lambda} 1 = 0$; аналогично рассуждениям из теоремы 2.3.12 заключаем, что λ — простое число p и, таким образом, F_p — подполе поля F, изоморфное полю $\mathbb{Z}_p$ (покажите это).

Определим линейную независимость множества элементов из поля F с коэффициентами из поля F_p очевидным образом (см. приложение в конце книги). Среди всех линейно независимых подмножеств поля F пусть $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ — подмножество с максимальным числом элементов. Если α — элемент поля F, то $\{\alpha, \alpha_1, \alpha_2, \ldots, \alpha_r\}$ — линейно зависимое множество, т.е. имеются коэффициенты $x_1, x_2, \ldots, x_r$, такие, что α — линейная комбинация элементов $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$. Очевидно, имеется p^r различных линейных комбинаций элементов $\{\alpha_1, \alpha_2, \ldots, \alpha_r\}$, и это доказывает теорему. □

Обратное к этому предложению см. в теореме 3.3.19.

Следствие 3.3.7. Если F — конечное поле, то оно имеет характеристику p для некоторого простого $p > 0$ и, таким образом, содержит подполе, изоморфное полю $\mathbb{Z}_p$.

Как мы знаем из разд. 2.3.2, групповой элемент a является примитивным элементом, или примитивным корнем, если его степени $1, a, a^2, \ldots$ пробегают все элементы группы (см. также теорему 2.3.20); примеры также исследовались в разд. 2.3.2. Следующая теорема гарантирует нам, что в каждом конечном поле есть примитивный корень.

Теорема 3.3.8 (теорема о примитивном корне). Пусть F — поле из q элементов. Тогда существует элемент $a \in F$, такой, что **(i)** каждый ненулевой элемент поля F является степенью элемента a, и **(ii)** порядок элемента a равен $q-1$.

Доказательство. Из теоремы 3.3.4 нам известно, что порядок элемента a делит $q-1$. Идея доказательства состоит в изучении множества O порядков элементов поля F. Очевидно, что O — некоторое множество целых чисел $\leq q-1$, и доказательство будет закончено, когда мы покажем, что $q-1$ принадлежит O. Это так, потому что тогда $a^{q-1}=1$ для некоторого $a \in F$ и никакая меньшая степень a не равна 1. То есть будет доказана часть (ii) теоремы. Часть (i) тогда легко следует из того, что все степени $1, a, a^2, \ldots, a^{q-2}$ различны, и, таким образом, они пробегают все ненулевые элементы поля F. Подробности можно найти в книгах (Berlekamp, 1968) или (Childs, 1979). □

Теорема 3.3.8 не дает нам никакой формулы для нахождения примитивного корня в поле $\mathbb{Z}_p$, p — простое число. Фактически никакой такой формулы не существует. Известно, однако, что если p — простое число вида $p=4q+1$, где q — простое число, то 2 — примитивный корень поля $\mathbb{Z}_p$. Таким образом, 2 — примитивный корень полей $\mathbb{Z}_p$ для $p=5, 13$, и т.д. Примитивные корни в конечном поле легко можно найти с помощью следующего метода [обнаруженного в работе (Albert, 1958)].

Нахождение примитивного корня в конечном поле. В конечном поле $GF(q)$ элемент a является примитивным корнем, если и только если

$$a^{(q-1)/d_i} \neq 1 \pmod{q}$$

для всех простых делителей $d_1, d_2, \ldots, d_r$ числа $q-1$.

Обобщением теоремы 3.3.5 получается следующая

Теорема 3.3.9. Пусть F — поле и $p(x)$ — (нормированный) полином из $F[x]$, $\deg[p(x)] \geq 1$. Тогда существует содержащее F поле K, такое, что в $K[x]$ полином $p(x)$ разлагается в произведение линейных сомножителей.

Доказательство. Доказательство индукцией по $\deg[p(x)] = n$ мы оставляем читателю в качестве упражнения. □

Поле K, определенное в теореме 3.3.9, называется *полем расщепления* для $p(x)$. Например, согласно основной теореме алгебры, $\mathbb{C}$ — поле расщепления для любого полинома из $\mathbb{Q}[x]$.

Пример. В $\mathbb{Q}[x]$ полином $p(x) = x^3 - 2$ неприводим. Он имеет корень в $\mathbb{R}$, а именно $2^{1/3}$, но $\mathbb{R}$ не является полем расщепления для $p(x)$, потому что

$$x^3 - 2 = (x - 2^{1/3})(x^2 + 2^{1/3}x + 4^{1/3}),$$

а второй сомножитель имеет два комплексных корня.

Пусть F — поле и K — поле, содержащее F. Предположим, что $\alpha \in K$ — корень некоторого ненулевого полинома $m(x) \in F[x]$. Тогда мы говорим, что элемент α *алгебраичен над* F. (Числа, не являющиеся алгебраическими, называются *трансцендентными*; примеры трансцендентных над $\mathbb{Q}$ чисел — e и ϖ.)

Теорема 3.3.10. Пусть элемент $\alpha \in K$ алгебраичен над F. Тогда существует единственный нормированный неприводимый полином $m(x) \in F[x]$, такой, что α является его корнем, и каждый полином $p(x) \in F[x]$ с корнем α делится на $m(x)$.

Доказательство. Используя принцип полного упорядочения множества степеней полиномов из $F[x]$ с корнем α, мы заключаем, что имеется ненулевой нормированный полином $m(x)$ наименьшей степени, для которого α является корнем. Методом от противного покажем, что полином $m(x)$ неприводим. То есть предположим, что $m(x) = a(x)b(x)$, где $\deg[a(x)] < \deg[m(x)]$ и $\deg[b(x)] < \deg[m(x)]$. Тогда $0 = m(\alpha) = a(\alpha)b(\alpha)$, и, поскольку K не содержит делителей нуля, или $a(\alpha) = 0$, или $b(\alpha) = 0$. Однако это и есть требуемое противоречие, поскольку α является теперь корнем полинома степени, меньшей, чем $\deg[m(x)]$. Поэтому полином $m(x)$ неприводим. Наконец, пусть $p(x)$ — любой полином в $F[x]$ с корнем α. Применяя теорему о делении, получаем $p(x) = m(x)q(x) + r(x)$, где $\deg[r(x)] < \deg[m(x)]$. Поскольку $p(\alpha) = 0$, то $r(\alpha) = 0$, и $r(x)$ — нулевой полином и, следовательно, $m(x)|p(x)$. □

Полином $m(x)$, определенный в теореме 3.3.10, называется *минимальным полиномом элемента* α *над* F, потому что из доказательства мы знаем, что это полином минимальной степени в $F[x]$ с корнем α.

В гл. 4 нам понадобится нахождение минимальных полиномов. Следующая теорема утверждает, что они существуют, и показывает, как их находить.

Теорема 3.3.11. Пусть $K = F[\alpha]$ — простое расширение поля F, где минимальный полином $m(x)$ элемента α над F имеет степень r.

Тогда, если β — произвольный элемент поля K, то β алгебраичен над F и минимальный полином элемента β над F имеет степень $\leq r$.

Доказательство. Мы знаем, что каждый элемент поля $K = F[\alpha]$ — полином от α степени $\leq r-1$. То же верно для любой степени элемента β. Чтобы найти минимальный полином элемента β над F, будем искать ненулевое решение уравнения

$$c_r\beta^r + c_{r-1}\beta^{r-1} + \cdots + c_1\beta + c_0 = 0.$$

Заменяя степени элемента β эквивалентными им полиномиальными выражениями от α и собирая коэффициенты при одинаковых степенях α, мы получаем систему n уравнений от $r+1$ неизвестных. Мы используем затем результат из линейной алгебры (см. приложение в конце книги), что любая система r однородных линейных уравнений от $r+1$ неизвестных имеет ненулевое решение, чтобы найти в $F[x]$ ненулевой полином степени $\leq r$ с корнем β. □

Пример. Пусть $m(x) = x^3 + x + 1$ — неприводимый над $\mathbb{Z}_2$ полином, и рассмотрим $K = \mathbb{Z}_2[x]/\equiv_{m(x)} = \mathbb{Z}_2[\alpha] = GF(2^3)$. Тогда $\alpha^3+\alpha+1 = 0$ и поле K состоит из полиномов от α степени ≤ 2 с коэффициентами в $\mathbb{Z}_2$. Чему равен минимальный полином элемента $\beta = \alpha + 1$ над $\mathbb{Z}_2$?

Чтобы найти ненулевое решение уравнения $c_3\beta^3 + c_2\beta^2 + c_1\beta^1 + c_0 = 0$, выразим степени β в виде полиномов от $(\alpha+1)$ и получим $c_3(\alpha+1)^3+c_2(\alpha+1)^2+c_1(\alpha+1)+c_0 = 0$. Пользуясь тем, что $\alpha^3 = \alpha+1$ (проверьте это), и собирая коэффициенты при одинаковых степенях α, получаем систему

$$\begin{aligned} c_2+c_1+c_0 &= 0,\\ c_1 &= 0,\\ c_3+c_2 &= 0, \end{aligned}$$

которую решаем в $\mathbb{Z}_2$. Решением будет $c_1 = 0$ и $c_3 = c_2 = c_0$, и поэтому минимальный полином элемента $\beta = \alpha + 1$ над $\mathbb{Z}_2$ равен $x^3 + x^2 + 1$.

Некоторые сокращения при выполнении вычисления вручную читатель может найти в книге (Berlekamp, 1968, pp. 112–117).

Следующий результат играет существенную роль.

Теорема 3.3.12. Для данных простого числа p и натурального числа r все конечные поля из $q = p^r$ элементов изоморфны.

Доказательство. Пусть F — конечное поле из q элементов. Тогда из теоремы 3.3.2 нам известно, что порядок произвольного ненулевого элемента $\alpha \in F$ делит $q-1$, т.е. $a^{q-1} = 1$ для $\alpha \neq 0$. Умножая последнее равенство на α, получаем $\alpha^q - \alpha = 0$, что выполняется и для $\alpha = 0$. Поэтому $\alpha_1, \alpha_2, \ldots, \alpha_q$, все элементы поля F, являются корнями полинома $x^q - x$. Следовательно, поскольку полином $x^q - x$ делится на $(x - \alpha_i)$, $i = 1, 2, \ldots, q$, он должен делиться на $\prod_{1 \leq i \leq q}(x - \alpha_i)$. Из равенства степеней следует, что $x^q - x = \prod_{1 \leq i \leq q}(x - \alpha_i)$. Из следствия 3.3.7 видно, что поле F получается из $\mathbb{Z}_p$ присоединением всех корней полинома $x^q - x$ и поэтому определено единственным образом с точностью до изоморфизма. □

Как результат теорем 3.3.5 и 3.3.12 мы имеем такие следствия:

Следствие 3.3.13. Любое конечное поле изоморфно простому расширению поля $\mathbb{Z}_p$ для некоторого простого числа p.

Следствие 3.3.14. Если q не является степенью простого числа p, то не существует конечного поля из q элементов.

Доказательство. Если F — конечное поле, то оно изоморфно $\mathbb{Z}_p[x]_{m(x)}$ для некоторого неприводимого полинома $m(x)$ степени r. Тогда $\mathbb{Z}_p[x]_{m(x)}$ имеет p^r элементов, и то же верно для F. □

Поскольку имеется по существу только одно конечное поле с p^r элементами, принято обозначать его $GF(p^r)$. Следующие теоремы весьма полезны.

Теорема 3.3.15. В любом поле характеристики p выполняется соотношение $(x - a)^p = x^p - a^p$.

Доказательство. Раскрывая скобки, получаем $(x-a)^p = x^p + \sum_{k=1}^{p}\{p(p-1)\ldots(p-k+1)/k!\}(-a)^k x^{p-k}$. Доказательство теперь немедленно следует из того, что коэффициенты делятся на p. □

В качестве следствия теоремы 3.3.15 мы получаем, что в поле характеристики p ни у одного элемента порядок не является кратным p. Этот результат обобщается следующим образом.

Теорема 3.3.16. Пусть F — поле характеристики p. Тогда для всех r

$$\left(\sum_{1 \leq i \leq k} a_i\right)^{p^r} = \left(\sum_{1 \leq i \leq k} a_i^{p^r}\right).$$

Доказательство. Доказательство — индукцией сначала по r для частного случая $k = 2$, а затем по k для произвольного r. Детали мы оставляем читателю. □

Из теорем 3.3.15 и 3.3.16 вытекает

Следствие 3.3.17. Пусть $q(x)$ — произвольный полином из $\mathbb{Z}_p[x]$, а α — один из его корней. Тогда α^p — также корень полинома $q(x)$.

Доказательство. Из $\sum_i q_i\alpha^i = 0$ легко следует, что $0 = \left(\sum_i q_i\alpha^i\right)^p = \sum_i (q_i\alpha^i)^p = \sum_i q_i^p\alpha^{ip} = \sum_i q_i^p(\alpha^p)^i = \sum_i q_i(\alpha^p)^i$. □

Если $m(x)$ — неприводимый полином в $\mathbb{Z}_p[x]$ и α — один из его корней, то α^{p^k} — также корень полинома $m(x)$ для любого натурального числа k, т.е. $m(\alpha^{p^k}) = [m(\alpha)]^{p^k} = 0$. Однако, поскольку степень полинома $m(x)$ конечна, начиная с некоторого места, последовательность $\alpha, \alpha^p, \alpha^{p^2}, \ldots$ должна повторяться. Следующая теорема утверждает, что $m(x)$, неприводимый полином из $\mathbb{Z}_p[x]$, не имеет кратных корней ни в каком расширении поля.

Теорема 3.3.18. Пусть $m(x)$ — неприводимый полином в $\mathbb{Z}_p[x]$ степени r, и пусть K — поле, содержащее $\mathbb{Z}_p$. Если $\alpha \in K$ — корень полинома $m(x)$, то все корни $m(x)$ суть $\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{r-1}}$.

Доказательство. Из замечаний, предшествующих теореме, мы видим, что $\alpha, \alpha^p, \alpha^{p^2}, \ldots$ суть все корни полинома $m(x)$.

Пусть n — наименьшее натуральное число, с которого последовательность начинает повторяться, т.е. пусть $\alpha^{p^n} = \alpha$[1]. Тогда $\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}$ — все различные корни полинома $m(x)$. [Иначе из $a^{p^i} = \alpha^{p^j}$ мы получаем $(\alpha^{p^i})^{p^{n-j}} = (\alpha^{p^j})^{p^{n-j}} = \alpha$, откуда $\alpha^{p^{n-(j-i)}} = \alpha$, что противоречит минимальности n.] Поэтому $m(x)$ имеет по крайней мере n корней, $n \leq r$.

Для завершения доказательства пусть $m_1(x) = (x-\alpha)(x-\alpha^p)\ldots(x-\alpha^{p^{n-1}})$. Так как $(\alpha^{p^{n-1}})^p = \alpha$, то $[m_1(x)]^p = m_1(x^p)$ и α — корень полинома $m_1(x)$. Таким образом, $m_1(x)$ принадлежит $\mathbb{Z}_p[x]$ и по теореме 3.3.10 полином $m(x)$ должен делить $m_1(x)$; поскольку $n \leq r$, мы имеем $m(x) = m_1(x)$. □

Докажем теперь обращение теоремы 3.3.6.

---

[1] Существование такого n вытекает из следствия 3.3.3 и теоремы 3.3.6. — *Прим. перев.*

Теорема 3.3.19. Для степени простого числа $q = p^r$, $r > 0$, существует одно, и с точностью до изоморфизма ровно одно, конечное поле из q элементов. Эти элементы — корни полинома $x^q - x$.

Доказательство. Рассмотрим полином $p(x) = x^q - x \in \mathbb{Z}_p[x]$. Тогда по теореме 3.3.9 существует поле расщепления K для полинома $p(x)$, т.е. в $K[x]$ полином $p(x)$ может быть представлен в виде произведения линейных сомножителей.

Пусть F — подмножество поля K, состоящее из всех корней полинома $x^q - x$ в K, т.е. F состоит из всех элементов $a \in K$, таких, что $a^q = a$. Покажем, что F — искомое конечное поле.

Для этого нам нужно показать, что F содержит $q = p^r$ элементов и что оно — поле. По теореме 3.2.17 $\gcd[p(x), p'(x)] = 1$ [поскольку мы находимся в $\mathbb{Z}_p$ и производная полинома $p(x)$ равна -1], и, следовательно, у $p(x)$ нет в K кратных корней. Поэтому в K имеется $q = p^r$ различных корней полинома $p(x)$ и F содержит p^r элементов. Чтобы показать, что F — поле, мы должны доказывать, что если $a, b \in F$, то то же верно и для $a + b$, $a \cdot b$, $-a$ и a^{-1}. Детали мы оставляем читателю. (*Указание.* Воспользуйтесь теоремой 3.3.16.) □

Мы теперь в состоянии доказать следующий результат.

Теорема 3.3.20. Для любого r в $Z_p[x]$ существует неприводимый полином степени r.

Доказательство. Пусть F — конечное поле из p^r элементов. Тогда по следствию 3.3.13 поле F изоморфно $\mathbb{Z}_p[x]_{m(x)}$ для некоторого неприводимого полинома $m(x) \in \mathbb{Z}_p[x]$. Поэтому поле $\mathbb{Z}_p[x]_{m(x)}$ также имеет p^r элементов, и степень $m(x)$ равна r. □

Теперь ясно, что любой неприводимый полином $m(x)$ степени r в $\mathbb{Z}_p[x]$ имеет корень в любом поле F из p^r элементов. Это верно, потому что поле F изоморфно $\mathbb{Z}_p[x]_{m(x)}$, в котором $m(x)$ имеет корень, а именно $[x]_{m(x)}$. Поэтому $m(x)$ имеет корень в F. Дополнительная информация получается из следующей теоремы, значение которой будет продемонстрировано в гл. 6.

Теорема 3.3.21. Полином $p(x) = x^{p^r} - x$ равен произведению всех неприводимых полиномов в $\mathbb{Z}_p[x]$, степени которых делят r.

Доказательство. Пусть $m(x)$ — неприводимый полином степени r над $\mathbb{Z}_p$, и рассмотрим поле F, полученное присоединением к $\mathbb{Z}_p$ корня α полинома $m(x)$; очевидно, F содержит p^r элементов. В силу теоремы 3.3.10 мы знаем, что для любого полинома $q(x)$ равенство

$q(\alpha) = 0$ справедливо, если и только если $q(x)$ делится на $m(x)$. Кроме того, по следствию 3.3.3 каждый элемент поля F — корень полинома $p(x)$ в F; в частности, $p(\alpha) = 0$, следовательно, $p(x)$ делится на $m(x)$. Однако то же верно и в случае $\deg[m(x)] = d$ и $d|r$, потому что если $r = d \cdot e$, то $\alpha^{p^d} = \alpha$, и, применяя это соотношение e раз, мы видим, что α — корень полинома $p(x)$.

Для завершения доказательства покажем методом от противного, что если s не делит r, то ни один неприводимый над $\mathbb{Z}_p$ полином степени s не делит $p(x)$. Предположим, что $\deg[m(x)] = s$ и что $m(x)|p(x)$. Рассмотрим поле F, получающееся из $\mathbb{Z}_p$ присоединением к нему корня α полинома $m(x)$. Мы можем считать α примитивным корнем в поле F, т.е. p^s является минимальным из натуральных чисел n, таких, что $\alpha^n = \alpha$. Поскольку $m(x)|p(x)$, элемент α является также корнем полинома $p(x)$, т.е. $\alpha^{p^r} = \alpha$. Пусть $r = ks + q$, где q — остаток от деления r на s. Тогда $\alpha = \alpha^{p^r} = \alpha^{p^{ks+q}} = \alpha^{p^{ks}p^q} = \alpha^{p^{ksp^q}} = \alpha^{p^q}$, поскольку индукцией по k легко показать, что $\alpha^{p^{ks}} = \alpha^{p^s p^{(k-1)s}} = \alpha^{p^s p^{(k-1)s}} = \alpha^{p^{(k-1)s}} = \alpha$. Если $0 < q < s$, то полученное выше соотношение $\alpha = \alpha^{p^q}$ противоречит предположению о примитивности корня α. □

Пример. Рассмотрим поле Галуа $GF(2^4) = \mathbb{Z}_2[x]_{m(x)}$, где $m(x) = x^4 - x - 1$, и выразим $x^{16} - x$ в виде произведения неприводимых полиномов в $\mathbb{Z}_2[x]$. (Заметим, что $+1 = -1$ в $GF(2)$.) Ясно, что $m(x) = x^4 - x - 1$ — сомножитель с корнями $\alpha, \alpha^2, \alpha^4, \alpha^8$ (по теореме 3.3.18), т.е. $x^4 - x - 1 = (x-\alpha)(x-\alpha^2)(x-\alpha^4)(x-\alpha^8)$ (см. табл. 3.3.1 для проверки последнего равенства). Другие очевидные сомножители — полиномы x и $(x-1)$.

Возьмем теперь степень элемента α, которая еще не рассматривалась, а именно, α^3. Тогда другим сомножителем будет полином с корнями $\alpha^3, \alpha^6, \alpha^{12}, \alpha^9$, т.е. полином $(x-\alpha^3)(x-\alpha^6)(x-\alpha^{12})(x-\alpha^9)$. Сомножитель $(x-\alpha^9)$ появляется потому, что последний полином имеет корень α^{12}, а значит, будет также иметь корень $(\alpha^{12})^2 = \alpha^{24} = \alpha^{15}\alpha^9 = \alpha^9$. Затем рассмотрим α^5 и еще один сомножитель — полином $(x-\alpha^5)(x-\alpha^{10})$. Наконец, последний сомножитель — это полином $(x-\alpha^7)(x-\alpha^{14})(x-\alpha^{13})(x-\alpha^{11})$. Сомножитель $(x-\alpha^{13})$ появляется, потому что последний полином имеет корень α^{14}, следовательно, он будет также иметь корень $(\alpha^{14})^2 = \alpha^{28} = \alpha^{15}\alpha^{13} = \alpha^{13}$. Аналогично сомножитель $(x-\alpha^{11})$ появляется, потому что α^{13} — корень и $(\alpha^{13})^2 = \alpha^{26} = \alpha^{15}\alpha^{11} = \alpha^{11}$. Поэтому

$$x^{16} - x = x(x-1)(x^4-x-1)(x^4-x^3-x^2-x-1)(x^4-x^3-1)(x^2-x-1).$$

Таблица 3.3.1

Три различных представления элементов поля $GF(2^4)$

Степенное представление	Полиномиальное представление	Векторное представление
0	0	(0000)
$\alpha^0 = 1$	1	(0001)
α	α	(0010)
α^2	α^2	(0100)
α^3	α^3	(1000)
α^4	$\alpha + 1$	(0011)
α^5	$\alpha^2 + \alpha$	(0110)
α^6	$\alpha^3 + \alpha^2$	(1100)
α^7	$\alpha^3 + \alpha + 1$	(1011)
α^8	$\alpha^2 + 1$	(0101)
α^9	$\alpha^3 + \alpha$	(1010)
α^{10}	$\alpha^2 + \alpha + 1$	(0111)
α^{11}	$\alpha^3 + \alpha^2 + \alpha$	(1110)
α^{12}	$\alpha^3+\alpha^2+\alpha+1$	(1111)
α^{13}	$\alpha^3 + \alpha^2 + 1$	(1101)
α^{14}	$\alpha^3 + 1$	(1001)

3.3.2. Построение полей Галуа $\boldsymbol{GF(2^r)}$

Мы представим теперь метод построения полей Галуа $GF(2^r)$ для любого r, исходя из двоичного поля $GF(2)$; $GF(2)$ называется *основным* полем, и характеристика $GF(2^r)$ равна 2. Отметим, что арифметика в $GF(2)$ совпадает с арифметикой в $\mathbb{Z}_2$, и мы уже знаем, как работать с элементами в этом поле, а также с полиномами над $\mathbb{Z}_2$. Также мы говорим, что неприводимый полином $p(x)$ степени r *c-примитивен*, если наименьшее целое число n, для которого $p(x)|(x^n - 1)$, — это $n = 2^r - 1$; в теории кодирования используют для таких полиномов термин *примитивный*, но, как мы отмечали, мы резервируем этот термин для полиномов с взаимно простыми коэффициентами. c-примитивные полиномы нелегко обнаружить и для них даются таблицы.

Мы начинаем с двух элементов 0 и 1 из $GF(2)$ и символа α. Определяем умножение «$\cdot$» для элементов из $GF(2)$ и α следующим обра-

зом: $0 \cdot \alpha = \alpha \cdot 0 = 0$, $1 \cdot \alpha = \alpha \cdot 1 = \alpha$; затем полагаем $\alpha^2 = \alpha \cdot \alpha$, $\alpha^3 = \alpha^2 \cdot \alpha = \alpha \cdot \alpha \cdot \alpha$ и т.д. Заметим, что из данного определения следует $0 \cdot \alpha^k = \alpha^k \cdot 0 = 0$, $1 \cdot \alpha^k = \alpha^k \cdot 1 = \alpha^k$ и $\alpha^j \cdot \alpha^k = \alpha^{j+k}$.

Таким образом, мы построили множество $F' = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^k, \ldots\}$, на элементах которого определили операцию умножения. Затем мы налагаем на элемент α некоторое условие, так что F' содержит только 2^r элементов и операция умножения замкнута в F'. Пусть $p(x)$ есть c-примитивный полином степени r над $GF(2)$, и предположим, что $p(\alpha) = 0$. Так как $p(x)|(x^{2^r-1}-1)$, то $x^{2^r-1}-1 = p(x)q(x)$ для некоторого $q(x)$ над $GF(2)$, и мы получаем $\alpha^{2^r-1} - 1 = p(\alpha)q(\alpha) = 0$, или $\alpha^{2^r-1} = 1$. Таким образом условие $p(\alpha) = 0$ превращает множество F' в новое множество F, которое является конечным и содержит элементы $\{0, 1, \alpha, \alpha^2, \ldots, \alpha^{2^r-2}\}$, которые все различны, т.е.

$$F = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^{2^r-2}\}.$$

Как и прежде, F^* обозначает ненулевые элементы множества F.

Покажем теперь, что F^* — (мультипликативная) группа относительно умножения, замкнутой в F^* операции. Прежде всего заметим, что элемент 1 — единичный элемент. Затем возьмем два элемента α^i и α^j из F^* и рассмотрим их произведение $\alpha^i \cdot \alpha^j = \alpha^{i+j}$. Если $i + j < 2^r - 1$, то это элемент из F^* и ничего делать не нужно; однако если $i + j \geq 2^r - 1$, то $i + j = (2^r - 1) + v$, где $0 \leq v < 2^r - 1$, и $\alpha^i \cdot \alpha^j = \alpha^{i+j} = \alpha^{(2^r-1)+v} = \alpha^v$ — элемент из F^*. Более того, каждый элемент $\alpha^i \in F^*$, $0 < i < 2^r - 1$, имеет мультипликативный обратный, равный $\alpha^{(2^r-1)-i}$. Поэтому F^* — коммутативная группа относительно умножения порядка $2^r - 1$.

Теперь определим на F операцию сложения «+» так, что F образует коммутативную группу относительно «+», и в процессе ее определения мы получим также другой способ представления элементов из F.

Заметим, что если мы разделим x^i на $p(x)$, $0 \leq i < 2^r - 1$, то получим $x^i = p(x)q_i(x) + v_i(x)$, $\deg[v_i(x)] < r$ над $GF(2)$, т.е. $v_i(x) = v_{ir-1}x^{r-1} + \cdots + v_{i2}x^2 + v_{i1}x + v_{i0}$. Поскольку $\gcd[x^i, p(x)] = 1$, $p(x)$ не делит x^i, и поэтому $v_i(x) \neq 0$ для любого i. Более того, $v_i(x) \neq v_j(x)$ для $i \neq j$. [В этом мы убеждаемся от противного. Предположим, что $v_i(x) = v_j(x)$ для $i \neq j$. Тогда из выражений $x^i = p(x)q_i(x) + v_i(x)$ и $x^j = p(x)q_j(x) + v_j(x)$ получаем $x^i - x^j = p(x)[q_i(x) - q_j(x)] + v_i(x) - v_j(x) = p(x)[q_i(x) - q_j(x)]$. Из последнего равенства следует, что $p(x)|(x^i - x^j) = x^i(1 - x^{j-i})$ при условии, конечно, что $j > i$. Так как $p(x)$ не делит x^i, то $p(x)|(1 - x^{j-i})$ — полином степени $<$

$2^r - 1$, а это и есть необходимое противоречие, поскольку $p(x)$ c-примитивен.] Поэтому для $i = 0, 1, 2, \ldots, 2^r - 2$ мы получаем $2^r - 1$ различных ненулевых полиномов $v_i(x)$ степени $\leq r-1$. Заменяя x на α в выражении $x^i = p(x)q_i(x) + v_i(x)$, получаем

$$\alpha^i = v_i(\alpha) = v_{ir-1}\alpha^{r-1} + \cdots + v_{i2}\alpha^2 + v_{i1}\alpha + v_{i0}.$$

Таким образом, из этого соотношения видно, что каждый элемент из F^* *единственным способом представляется в виде полинома* от α над $GF(2)$ степени $\leq r-1$. Нулевой элемент из F представляется нулевым полиномом. Следовательно, сложение в F можно рассматривать как сложение полиномов, и нам известно, как делать это над $GF(2)$. Теперь легко можно проверить, что F — коммутативная группа по сложению, а вместе с предыдущими результатами это означает, что F — поле.

До настоящего времени мы имели дело с двумя представлениями поля $GF(2^r)$: степенное представление (удобное при умножении) и полиномиальное представление (удобное при сложении); см. также табл. 3.3.1. Существует также третье представление, получаемое, если представлять полином вектором его коэффициентов. Определяя r-кортеж

$$\mathbf{v} = (v_{r-1}, v_{r-2}, \ldots, v_2, v_1, v_0)$$

как *вектор* над $GF(2)$, получаем очевидным образом векторное представление поля $GF(2^r)$. Отметим, что существует 2^r различных векторов длины r. Сложение векторов выполняется покомпонентно, и результат сложения — снова вектор; следовательно, оно весьма удобно для сложения элементов поля $GF(2^r)$. Умножение вектора $\mathbf{v}$ на скаляр s определяется правилом

$$s(v_{r-1}, v_{r-2}, \ldots, v_2, v_1, v_0) = (sv_{r-1}, sv_{r-2}, \ldots, sv_2, sv_1, sv_0).$$

Кроме того, имеется нулевой вектор. Множество всех двоичных векторов длины r с операциями и свойствами, определенными выше, образует *векторное пространство* над $GF(2)$, обозначаемое $\mathbf{V}_r$.

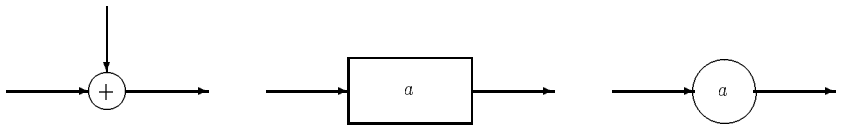
Скалярное произведение двух векторов $\mathbf{u} = (u_{r-1}, u_{r-2}, \ldots, u_2, u_1, u_0)$ и $\mathbf{v} = (v_{r-1}, v_{r-2}, \ldots, v_2, v_1, v_0)$ в $\mathbf{V}_r$ определим формулой $\mathbf{u} \cdot \mathbf{v} = (u_{r-1}v_{r-1}, u_{r-2}v_{r-2}, \ldots, u_2v_2, u_1v_1, u_0v_0)$. Если скалярное произведение двух векторов равняется нулю, то векторы называются *ортогональными*.

Пример. Пользуясь c-примитивным полиномом $p(x) = x^4 + x + 1$, построим поле $GF(2^4)$. Элементы получаются повторным применением равенства $\alpha^4 = \alpha + 1$, где $p(\alpha) = 0$; в табл. 3.3.1 для них даны описанные выше три представления. Например, $\alpha^5 = \alpha^4 \cdot \alpha = (\alpha+1)\alpha = \alpha^2 + \alpha$ и т.д. Для перемножения двух элементов мы просто складываем их показатели степени и пользуемся тем, что $\alpha^{15} = 1$; например, $\alpha^8 \cdot \alpha^9 = \alpha^{17} = \alpha^2$. Кроме того, $\alpha^5/\alpha^8 = \alpha^5 \cdot \alpha^{(15-8)} = \alpha^5 \cdot \alpha^7 = \alpha^{12}$. Для сложения мы используем либо полиномиальное, либо векторное представление элементов и выполняем сложение покомпонентно.

3.3.3. Схемы для полиномиальной арифметики в $GF(2^r)$

На материале этого раздела основана теория кодов, обнаруживающих и исправляющих ошибки, обсуждаемая в гл. 4 [мы придерживаемся книги (Afrati, 1985)]; однако тема довольно технична, и если читатель не знаком с регистрами сдвига, то для хорошего понимания от него могут потребоваться дополнительные усилия [см., например, (Taub, 1982)].

Схемы, которые мы будем обсуждать, в основном состоят из элементов, показанных на рис. 3.3.1. Сумматор выводит сумму двух значений, представленных на входе, а умножитель выводит произведение значения, представленного на входе, на константу a. Элемент памяти «сохраняет» значение, представленное на входе, а потом выводит его.



Сумматор Элемент памяти Умножитель

Рис. 3.3.1.
Строительные блоки схем.

Наши схемы очень напоминают регистры сдвига. Регистры сдвига работают с сигналом сдвига, который обычно обеспечивается генератором частоты; этот сигнал *не* будет включаться в приведенные ниже диаграммы. В регистрах сдвига элементы памяти — просто устройства задержки (триггеры D-типа), где значение, представленное на выходе — это значение, представленное на входе точно одним тактом раньше. Каждый элемент задержки рассматривается как этап регистра сдвига. Более того, поскольку мы имеем дело с элементами поля

$GF(2)$, сумматор — это просто схема исключающего ИЛИ, а умножитель — просто соединение, если константа равна 1, или разрыв соединения, если константа равна нулю.

Ввод и вывод в регистре сдвига выполняются последовательно. Когда вводится или выводится полином, то на вход или выход подаются только коэффициенты [которые принадлежат полю $GF(2)$] по одному элементу за такт. Отметим, что *сначала передаются коэффициенты при самых высоких степенях.* (Это происходит потому, что при делении мы сначала работаем с коэффициентами при слагаемых наивысшей степени в делимом.) Например, полином

$$p(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_0$$

вводится в регистр сдвига или выводится из него как последовательность элементов поля $GF(2)$, в которой первым идет c_n, одним тактом позже — c_{n-1} и т.д.

Ниже представлены схемы для умножения или деления произвольного полинома на другой данный полином. Схема, представленная на рис. 3.3.2, умножает любой полином

$$a(x) = a_k x^k + a_{k-1} x^{k-1} + \cdots + a_0,$$

появляющийся на входе, на данный полином

$$h(x) = h_r x^r + h_{r-1} x^{r-1} + \cdots + h_0.$$

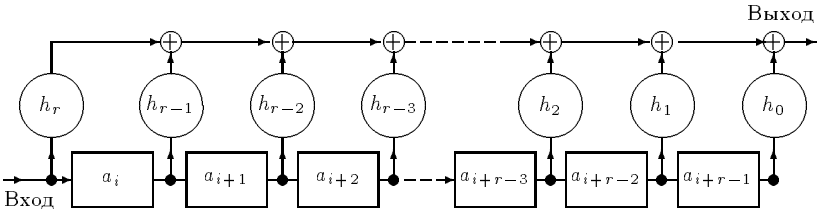


Рис. 3.3.2.
Схема для умножения полиномов.

Предполагается, что изначально все элементы задержки содержат «0»; кроме того, за коэффициентами полинома $a(x)$, которые вводятся последовательно, следует r раз «0».

Очевидно, что вычисляемое произведение равно

$$\begin{aligned} a(x)h(x) = {} & a_k h_r x^{k+r} + (a_{k-1}h_r + a_k h_{r-1}) x^{k+r-1} \\ & + (a_{k-2}h_r + a_{k-1}h_{r-1} + a_k h_{r-2}) x^{k+r-2} \\ & + \ldots (a_0 h_2 + a_1 h_1 + a_2 h_0) x^2 + (a_0 h_1 + a_1 h_0) x + a_0 h_0. \end{aligned}$$

Как видно из рис. 3.3.2, когда первый коэффициент a_k полинома $a(x)$ появляется на входе, первый коэффициент $a_k h_r$ полинома $a(x)h(x)$ появляется на выходе. В этот момент все элементы задержки содержат «0». Через такт на входе появляется a_{k-1}, a_k содержится в первом элементе памяти, а остальные элементы памяти содержат «0»; выход равен $a_{k-1}h_r + a_k h_{r-1}$, что совпадает со вторым коэффициентом произведения $a(x)h(x)$. Аналогично, после двух тактов a_{k-2} появляется на входе, элементы памяти содержат $a_{k-1}, a_k, 0, \ldots, 0, 0, 0$, и на выходе появляется третий коэффициент полинома $a(x)h(x)$. Этот процесс продолжается таким же образом. После $r + k - 1$ тактов регистр сдвига содержит $0, 0, 0, \ldots, 0, a_0, a_1$, и предпоследний коэффициент полинома $a(x)h(x)$ появляется на выходе, а именно $a_0 h_1 + a_1 h_0$. После $r + k$ тактов регистр сдвига содержит $0, 0, 0, \ldots, 0, 0, a_0$, и выход — $a_0 h_0$, последний коэффициент произведения $a(x)h(x)$.

Другая схема для умножения показана на рис. 3.3.3. Коэффициенты произведения формируются в элементах памяти регистра сдвига. Когда первый коэффициент появляется на входе, на выходе появляется $a_k h_r$, и все элементы памяти содержат «0». Через такт регистр сдвига содержит $a_k h_0, a_k h_1, \ldots, a_k h_{r-1}$, вход — a_{k-1}, а выход — это $a_{k-1}h_r + a_k h_{r-1}$, что является вторым коэффициентом произведения $a(x)h(x)$. В следующем такте регистр сдвига содержит $a_{k-1}h_0, a_k h_0 + a_{k-1}h_1, a_k h_1 + a_{k-1}h_2, \ldots, a_k h_{r-2} + a_{k-1}h_{r-1}$, вход есть a_{k-2}, а выход — это $a_{k-2}h_r + a_{k-1}h_{r-1} + a_k h_{r-2}$, третий коэффициент полинома $a(x)h(x)$. Процесс продолжается таким образом и дальше.

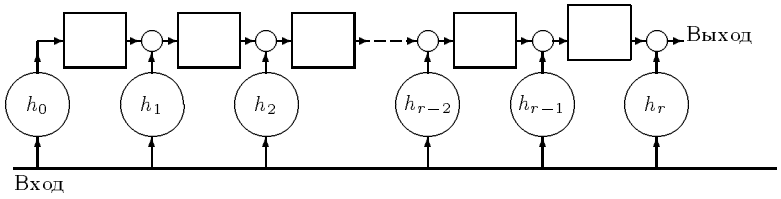


Рис. 3.3.3.
Другая схема для умножения полиномов.

На последнюю схему можно взглянуть с другой стороны. А именно, множество r элементов памяти образует регистр сдвига, который может запоминать полином. В начале этот полином равен 0. Когда вводится a_k, к содержимому регистра добавляется $a_k h(x)$. Задержка на один такт означает умножение на x, и на выходе мы получаем первый коэффициент. Появление a_{k-1} на входе добавляет $a_{k-1}h(x)$ к содержимому регистра, а единичная задержка умножает на x, и мы получаем второй коэффициент на выходе и т.д.

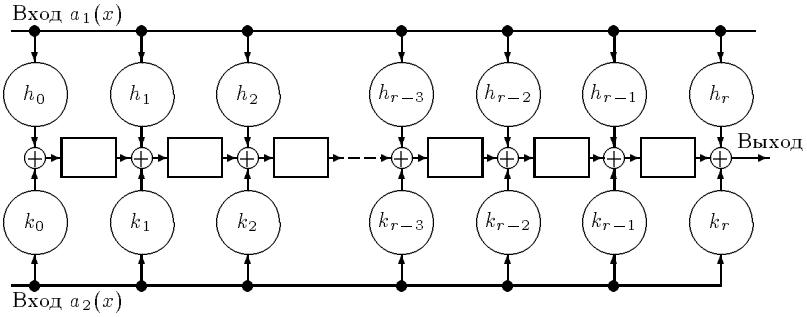


Рис. 3.3.4.
Полиномиальный умножитель с двумя входами.

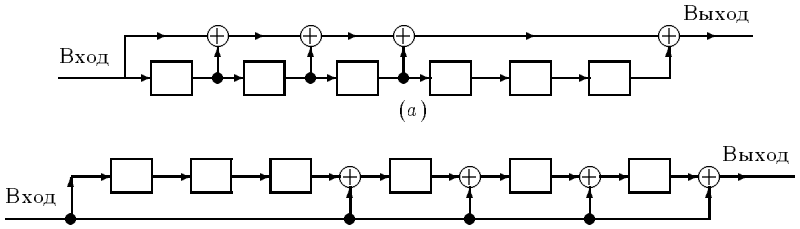


Рис. 3.3.5.
Схема для умножения на $x^6 + x^5 + x^4 + x^3 + 1$.

Схема представленного на рис. 3.3.3 типа может иметь несколько входов. Например, схема на рис. 3.3.4 имеет два входа, $a_1(x)$ и $a_2(x)$, и выход равен

$$b(x) = a_1(x)h(x) + a_2(x)k(x),$$

где

$$h(x) = h_r x^r + h_{r-1}x^{r-1} + \cdots + h_0, \quad k(x) = k_r x^r + k_{r-1}x^{r-1} + \cdots + k_0.$$

Пример. Схема, приведенная на рис. 3.3.5, умножает входной полином на $h(x) = x^6 + x^5 + x^4 + x^3 + 1$ над $GF(2)$. Читателю следует шаг за шагом рассмотреть операции этой схемы и сравнить результаты с результатами вычислений вручную.

Схема для деления произвольного полинома, скажем, $d(x) = d_n x^n + d_{n-1}x^{n-1} + \cdots + d_0$ на полином $g(x) = g_r x^r + g_{r-1}x^{r-1} + \cdots + g_0$, представлена на рис. 3.3.6. Регистр содержит вначале все «0». Выход равен «0» для первых r тактов. Затем появляется первый ненулевой выход, это $d_n g_r^{-1}$ — первый коэффициент частного. Для

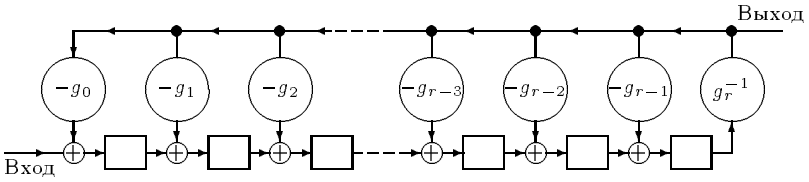


Рис. 3.3.6.
Схема для деления полиномов.

каждого коэффициента частного q_j полином $q_j g(x)$ должен вычитаться из делимого [Напомним, что в $GF(2)$ сложение совпадает с вычитанием.] Это достигается с помощью линий обратной связи. После n тактов частное полностью появляется на выходе, а остаток находится в элементах памяти регистра. Это легче понять, если рассмотреть схему на рис. 3.3.7, взять в качестве делимого полином $x^{13} + x^{11} + x^{10} + x^7 + x^4 + x^3 + x + 1$ и выполнить деление.

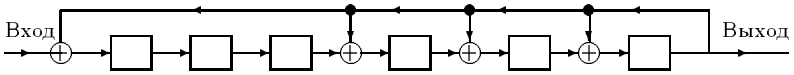


Рис. 3.3.7.
Схема для деления на $x^6 + x^5 + x^4 + x^3 + 1$.

Приведенные выше схемы легко приспособить для полиномиальной арифметики по модулю $g(x)$. Регистр, показанный на рис. 3.3.6, может запоминать элементы поля $GF(2)$, которые можно рассматривать как коэффициенты полинома

$$b(x) = b_{r-1}x^{r-1} + b_{r-2}x^{r-2} + \cdots + b_0$$

степени $\le r - 1$. После сдвига содержимого регистра на единицу вправо (или по истечении одного такта) полином в памяти становится равным

$$b'(x) = b_{r-2}x^{r-2} + b_{r-3}x^{r-3} + \cdots + b_0 x - b_{r-1}\{g_r^{-1}g(x) - x^r\},$$

где последнее слагаемое — результат линии обратной связи. Это можно переписать в виде

$$b'(x) = xb(x) - b_{r-1}g_r^{-1}g(x),$$

что совпадает с

$$b'(x) = xb(x) \pmod{g(x)}.$$

Чтобы сделать изложение яснее, рассмотрим пример. Возьмем $g(x) = x^4 + x + 1$ и поле $GF(2)$. Полином $g(x)$ c-примитивен и его корень α — примитивный корень поля $GF(2^4)$, т.е. все элементы поля $GF(2^4)$ получаются из первых 16 степеней α. Соответствующий регистр показан на рис. 3.3.8. Если поместить «1» в первый слева элемент памяти и «0» в остальные, то последовательные сдвиги дадут нам представления различных степеней α в том же самом виде, как они даны в табл. 3.3.1. Отметим, например, что выход «1» из последнего элемента памяти соответствует α^4, что заменяется на $\alpha + 1$ с помощью линии обратной связи.

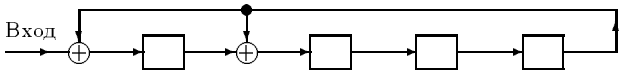


Рис. 3.3.8.
Схема, которая производит вычисления в $GF(2^4)$.

Перемножение двух элементов в $GF(2^4)$ может быть выполнено с использованием регистра на рис. 3.3.8 в качестве накапливающего регистра точно так же, как это реализуется в цифровых ЭВМ.

Пример. Используя регистр, изображенный на рис. 3.3.8, перемножим элементы α^{10} и α^7 поля $GF(2^4)$, как они представлены в табл. 3.3.1, т.е. $\alpha^{10} = \alpha^2 + \alpha + 1 = (0, 1, 1, 1)$ и $\alpha^7 = \alpha^3 + \alpha + 1 = (1, 0, 1, 1)$. Содержимое накапливающего регистра показывается ниже после каждой операции. Отметим, что используется схема, которая может прибавлять векторы к вектору, содержащемуся в накапливающем регистре; кроме того, векторное представление элементов не содержит запятых и коэффициенты при α^0 записываются слева.

1 1 0 1

	Накопитель
	0000
Добавить $1 \cdot (1110)$	1110
Сдвиг	0111
Добавить $0 \cdot (1110)$	0111
Сдвиг	1111
Добавить $1 \cdot (1110)$	0001
Сдвиг	1100
Добавить $1 \cdot (1110)$	0010

что и является ответом.

Упражнения

Раздел 3.1.1

1. Если мы работаем в полиномиальной арифметике по модулю 7, то чему равно $3x^2 + 5x + 1$ минус $6x + 4$? Чему равно произведение этих полиномов?
2. Является ли произведение нормированных полиномов нормированным? Чему равна степень произведения полиномов степеней m и n соответственно? Чему равна степень суммы полиномов степеней m и n соответственно?
3. Для каких простых чисел p существуют, и существуют ли вообще, такие числа, что полиномы $x^5 + 2x + 1$ и $x^8 + 8x^2 + 1$ совпадают как функции на $\mathbb{Z}_p$?
4. **a.** " Найдите корни полинома $x^3 + 3x + 3$ в $\mathbb{Z}_5$.
   b. Используя **PDF**, разделите $2x^4 + 3x^3 + x + 4$ на $x^2 + 2$ в $\mathbb{Z}_5$.
5. Покажите, что если $p_1(x)$ и $p_2(x)$ — два полинома в $\mathbb{Z}[x]$ и полином $p_2(x)$ нормирован, то при выполнении деления в кольце $\mathbb{Q}[x]$, $p_1(x) = p_2(x)q(x) + r(x)$, полиномы $q(x)$ и $r(x)$ принадлежат кольцу $\mathbb{Z}[x]$. Что произойдет, если полином $p_2(x)$ не нормирован?
6. Пусть $p_1(x)$ и $p_2(x)$ — два полинома над полем вещественных чисел степени не более n. Если $x_1, x_2, \ldots, x_k$, $k > n$, — различные числа, причем $p_1(x_i) = p_2(x_i)$, $i = 1, 2, \ldots, k$, то $p_1(x) = p_2(x)$ для всех значений x.
7. Пусть $p(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0$ — полином над полем вещественных чисел степени n, и пусть x_0 — вещественное число, такое, что $p(x_0) > 0$. Если $q(x)$ удовлетворяет соотношениям

$$\begin{aligned} p(x) &= (x - x_0)q(x) + p(x_0) \\ &= (x - x_0)(b_n x^{n-1} + b_{n-1}x^{n-2} + \cdots + b_2 x + b_1) + p(x_0) \end{aligned}$$

   и $b_i > 0$, $i = 1, 2, \ldots, n$, то ни один вещественный нуль полинома $p(x)$ не превосходит x_0.

В двух следующих упражнениях представлены два специальных типа полиномов. Они использовались для тестирования различных методов отделения вещественных корней, описанных в гл. 7.

8. Полиномы Чебышёва могут быть рекурсивно получены по формулам

$$T_0(x) := 1, \qquad T_1(x) := x \qquad \text{и}$$
$$T_{n+2}(x) := 2xT_{n+1}(x) - T_n(x), \quad n \geq 0.$$

Все корни этих полиномов простые, вещественные, расположены в открытом интервале $(-1, 1)$ и симметричны (т.е. если x — корень, то и $-x$ — также корень); см. также разд. 7.6. Определите $T_2(x)$, $T_3(x)$, $T_4(x)$ и $T_5(x)$.

9. Полиномы Лежандра могут быть рекурсивно получены по формулам

$$P_0(x) := 1, \qquad P_1(x) := x \qquad \text{и}$$
$$(n+2)P_{n+2}(x) := (2n+3)xP_{n+1}(x) - (n+1)P_n(x), \quad n \geq 0.$$

Все корни этих полиномов простые, вещественные, расположены в открытом интервале $(-1, 1)$ и симметричны (т.е. если x — корень, то и $-x$ — также корень). Определите $P_2(x)$, $P_3(x)$, $P_4(x)$ и $P_5(x)$ с *целыми* коэффициентами.

Раздел 3.1.2

1. Для полинома $p(x)$ найдите оценку для времени вычисления полинома $p(\alpha + x)$, $\alpha \geq 1$. [*Указание.* См. также статью (Akritas, Danielopoulos, 1980).]

2. Для $p(x) = x^3 - 7x + 7$ вычислите $p(3 + x)$, используя схему, о которой говорилось в этом разделе.

3. Чему равно время вычислений подхода «грубой силы» для вычисления значения полинома в данной точке? А именно, каждый член вычисляется по формуле $x^i := x \cdot x^{i-1}$, а затем умножается на c_i.

Следующие упражнения распространяют метод Руффини–Горнера на полиномы с комплексными коэффициентами.

4. Рассмотрим полином $p_n(z) = c_0 + c_1 z + \cdots + c_n z^n$, где теперь $z = x + iy$, $c_j = a_j + ib_j$, $j = 0, 1, \ldots, n$ (т.е. коэффициенты являются гауссовыми числами, определенными в упр. 7 разд. 2.2.2) и $p_n(z) = g_n(x, y) + ih_n(x, y)$ (гауссов полином). Следовательно, в рекурсивной схеме (RH) этого раздела p_k будет комплексным и вида $p_k := g_k + ih_k$. (Напомним, что $\alpha = z = x + iy$.)

Покажите, что следующая рекурсивная схема — это метод Руффини–Горнера для полиномов над полем комплексных чисел:

$$g_0 := a_n,$$
$$h_0 := b_n$$

и для $k > 0$ (пока $k \neq n$)

$$g_k := a_{n-k} + x \cdot g_{k-1} - y \cdot h_{k-1},$$
$$h_k := b_{n-k} + y \cdot g_{k-1} + x \cdot h_{k-1}.$$

5. Вычислите значение полинома $p_3(z) = (1-5i)z^3 - 2iz^2 - 7z + (4-3i)$ в точке $z = 2+3i$.

6. Вычислите значение полинома $p_5(z) = z^5 + (-9+2i)z^4 + (18+21i)z^3 + (39-72i)z^2 - (29-69i)z - 116i$ в точке $z = 3+i$.

7. Разработайте алгоритм, который для данного гауссова полинома $p(z)$ и гауссова целого числа $a+bi$ вычисляет полином $p(z+a+ib)$.

8. Воспользуйтесь алгоритмом упр. 7 и вычислите $p(z+2+3i)$, где $p(z)$ — полином из упр. 5.

9. Воспользуйтесь алгоритмом упр. 7 и вычислите $p(z+3+i)$, где $p(z)$ — полином из упр. 6.

Раздел 3.1.3

1. Каково время вычисления интерполяционного полинома Лагранжа?

2. Используйте интерполяцию Лагранжа для нахождения полинома $p(x) \in \mathbb{R}[x]$ степени не более 2, такого, что $p(1) = -8$, $p(3) = 2$ и $p(4) = 13$.

3. Найдите полином $p(x) \in \mathbb{Z}_{11}[x]$, такой, что $p(1) = 8 \pmod{11}$, $p(3) = 1 \pmod{11}$ и $p(7) = 4 \pmod{11}$.

4. Покажите, что

$$\sum_{1 \le i \le n+1} L_i(x) = 1$$

для всех x.

5. Пусть $a_1, a_2, \ldots, a_{n+1}$ — различные элементы поля J, и пусть $b_1, b_2, \ldots, b_{n+1}$ принадлежат J. Пусть $p(x)$ — полином из $J[x]$, $\deg[p(x)] \le n$, такой, что $p(a_i) = b_i$, $1 \le i \le n+1$.

Покажите, что $p(x) - b_1 = (x - a_1)q(x)$ для некоторого полинома $q(x)$ из $J[x]$, такого, что $\deg[q(x)] \leq n - 1$. Чему равны значения полинома $q(x)$ в точках $a_2, a_3, \ldots, a_{n+1}$? Опишите рекурсивную процедуру вычисления $p(x)$. Используйте эту процедуру, чтобы выполнить интерполяцию в упр. 1 и 2.

6. (Двумерная интерполяция.) Разработайте алгоритм для вычисления полинома $p(x, y) \in J[x, y]$, J — поле, где степень полинома $p(x, y)$ по y меньше n и

$$p(x, a_i) = b_i(x), \qquad i = 1, 2, \ldots, n,$$

для различных $a_i \in J$ и $b_i(x) \in J[x]$. Покажите, что полином $p(x, y)$ определен однозначно.

a. Если Предположить, что операции в поле J выполняются за время $O(1)$ и что степени полиномов $b_i(x)$ меньше m, $i = 1, 2, \ldots, n$, чему равно время вычислений вашего алгоритма (в терминах m и n)?

b. Вычислите полином $p(x, y) \in \mathbb{Z}_{11}[x, y]$, такой, что

$$p(x, 0) = x^3 + 7,$$
$$p(x, 1) = x^3 + 2x + 3,$$
$$p(x, 2) = x^3 + 5.$$

7. (Многомерная интерполяция.) Разработайте алгоритм для вычисления полинома $p(x_1, x_2, \ldots, x_r) \in J[x_1, x_2, \ldots, x_r]$, J — поле, где степень полинома $p(x_1, x_2, \ldots, x_r)$ по x_r меньше n и

$$p(x_1, x_2, \ldots, x_{r-1}, a_i) = b_i(x_1, x_2, \ldots, x_{r-1}), \qquad i = 1, 2, \ldots, n,$$

для различных $a_i \in J$ и $b_i(x_1, x_2, \ldots, x_{r-1}) \in J[x_1, x_2, \ldots, x_{r-1}]$. Покажите, что полином $p(x_1, x_2, \ldots, x_r)$ определен однозначно. Более того, предполагая, что операции в поле J выполняются за время $O(1)$ и что степени полиномов $b_i(x_1, x_2, \ldots, x_{r-1})$ меньше n, $i = 1, 2, \ldots, n$, оцените время вычислений вашего алгоритма.

Раздел 3.1.4

1. Покажите, что $p_1(x) \equiv p_2(x)$ тогда и только тогда, когда $m(x)|[p_1(x) - p_2(x)]$ определяет отношение эквивалентности на $J[x]$, J — поле.

2. Завершите доказательство теоремы 3.1.9.

3. Используя схему «(вычисление значений)–интерполяция», разработайте алгоритм для *пробного* деления двух полиномов над полем, т.е. если $p_1(x)|p_2(x)$, то возвращают частное, в противном случае утверждается, что $p_1(x)$ не делит $p_2(x)$. [*Указание.* Здесь существенно то, что если $\deg[p_1(x)] = n$ и $\deg[p_2(x)] = m$, $m \geq n \geq 0$, то для частного $q(x)$ мы имеем $\deg[q(x)] = m - n$. Таким образом, если степень полинома, полученного интерполяцией, не равна $m - n$, то мы можем утверждать, что $p_1(x)$ не делит $p_2(x)$.]

4. **a.** Чему равно время вычислений алгоритма, разработанного в упр. 3?

b. Покажите, как полученный алгоритм вычисляет пробные деления $p_2(x)/p_1(x)$ над $\mathbb{Z}_{11}[x]$ для

$$p_1(x) = 5x + 7 \qquad \text{и} \qquad p_(x) = x^2 + 6x + 8;$$
$$p_1(x) = 5x + 7 \qquad \text{и} \qquad p_(x) = x^2 + 6x + 6.$$

5. Разработайте алгоритм, аналогичный алгоритму упр. 3 для *пробного* деления над $\mathbb{Z}$, разделив $969/19$ и $970/19$, пользуясь делениями только над $\mathbb{Z}_p$ для $p < 20$.

6. Чему равно время вычислений алгоритма «(вычисление значений)–интерполяция» для полиномиального умножения над $\mathbb{Z}[x]$, разработанного в этом разделе? (Как всегда, предполагается единичная цена операций одинарной точности.)

Раздел 3.2.1

1. Делит ли полином $x^2 + x + 1$ полином $x^6 - 1$ над целыми числами?

2. Найдите все m такие, что $x^3 + 5$ делит $x^5 + x^3 + x^2 - 15$ в $\mathbb{Z}_m[x]$.

3. Вычислите наибольший общий делитель полиномов $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$ над вещественными числами.

4. В упр. 7 разд. 2.2.2 мы определили кольцо $\mathbb{Z}[i]$ гауссовых целых чисел; напомним, что если $z = x + iy$ — такое число, $x, y \in \mathbb{Z}$, то его норма равна $n(z) = x^2 + y^2 [= (x + iy)(x - iy)]$. Покажите, что $\mathbb{Z}[i]$ — евклидова область.

Раздел 3.2.2

1. (Цепные дроби для полиномиальной аппроксимации.) Пусть $c(x)$ и $d(x)$ — два полинома над полем, причем $\deg[c(x)] > \deg[d(x)]$, и пусть $a_1(x), a_2(x), \ldots$ — полиномы, получающиеся как частные при применении алгоритма Евклида к $c(x)$ и $d(x)$;

кроме того, пусть $p_0(x) = q_{-1}(x) = 0$ и $p_{-1}(x) = q_0(x) = 1$. Мы хотим показать, что подходящие дроби $p_n(x)/q_n(x)$ для цепной дроби $[a_1(x), a_2(x), \ldots]$ суть «лучшие» приближения низких степеней для рациональной функции $d(x)/c(x)$, где подходящие дроби определяются аналогично тому, как они определялись в разд. 2.2.4. (Читателю следует еще раз посмотреть его.)

Докажите, что если $p(x)$ и $q(x)$ — два полинома, таких, что $\deg[q(x)] < \deg[q_n(x)]$ и $\deg[p(x)c(x) - q(x)d(x)] \leq \deg[p_{n-1}(x)c(x) - q_{n-1}(x)d(x)]$ для некоторого $n \geq 1$, то $p(x) = cp_{n-1}(x)$ и $q(x) = cq_{n-1}(x)$ для некоторой константы c. Каждый $q_n(x)$ — «лучший» полином в том смысле, что ни для какого ненулевого полинома $q(x)$ меньшей степени нельзя подобрать полином $p(x)$ так, чтобы степень полинома $p(x)c(x) - q(x)d(x)$ не превосходила степени $p_n(x)c(x) - q_n(x)d(x)$. [*Указание.* Воспользуйтесь алгоритмом **XEA-P**; см. также (Knuth, 1981; p. 622).]

2. Докажите, что если $p_1(x), p_2(x), p_3(x) \in J[x]$, J — поле, $\gcd[p_1(x), p_2(x)] = 1$ и $p_3(x) | p_1(x)$, то $\gcd[p_3(x), p_2(x)] = 1$.

3. Докажите, что если $p_1(x), p_2(x), p_3(x) \in J[x]$, J — поле, $\gcd[p_1(x), p_2(x)] = 1$, то $\gcd[p_1(x)p_3(x), p_2(x)] = \gcd[p_3(x), p_2(x)]$.

4. **a.** Примените алгоритм **XEA-P** к полиномам $x^2 + 1$ и $x^5 + 1$ в $\mathbb{Z}_2[x]$.

b. Решите в $\mathbb{Q}[x]$ уравнение $p(x)(x^2 - 3x + 2) + q(x)(x^2 + x + 1) = 1$.

c. В $\mathbb{Q}[x]$ найдите gcd полиномов $x^{11} - 1$ и $x^9 - 1$. [*Указание.* Воспользуйтесь тождеством $x^m - 1 \pmod{x^n - 1} = x^{m \pmod n} - 1$.]

5. Используйте алгоритм **XEA-P**, чтобы найти $\gcd[p_1(x), p_2(x)]$ для $p_1(x), p_2(x) \in \mathbb{Z}_p[x]$ в каждом из следующих примеров. В каждом случае вычислите соответствующие полиномы $g(x)$ и $f(x)$, такие, что $\gcd[p_1(x), p_2(x)] = p_1(x)g(x) + p_2(x)f(x)$.

a. $p_1(x) = x^3 + x + 1$, $p_2(x) = x^2 + x + 1$ для $p = 3$ *и* $p = 2$.

b. $p_1(x) = x^4 + x^3 + x + 1$, $p_2(x) = x^3 + x^2 + x + 1$ для $p = 2$ *и* $p = 3$.

c. $p_1(x) = x^5 + x^4 + x^3 + x + 1$, $p_2(x) = x^4 + x^3 + x^2 + x + 1$ для $p = 5$.

d. $p_1(x) = x^5 + x^4 + x^3 - x^2 - x + 1$, $p_2(x) = x^3 + x^2 + x + 1$ для $p = 5$ *и* $p = 3$.

Раздел 3.2.3

1. Докажите, что в $\mathbb{R}[x]$ нет неприводимых полиномов нечетной степени и что в $\mathbb{C}[x]$ нет неприводимых полиномов степени 2.
2. Известно, что у полинома $p(x) = x^3 - 2x^2 + x - 2$ есть корни $-i$ и $+i$; разложите $p(x)$ в произведение неприводимых полиномов в $\mathbb{Z}[x]$.
3. Найдите, если они существуют, все сомножители степени 1 в $\mathbb{Z}[x]$ следующих полиномов:
   a. $x^3 - 2x^2 - 5x + 6$.
   b. $3x^4 + 4x^3 - 6x^2 - 9x - 10$.
4. Докажите, что если полином $p(x)$ в $\mathbb{Z}[x]$ нормирован, то он примитивен.
5. Покажите, что если число $r \in \mathbb{Q}$ таково, что полиномы $r \cdot p(x)$ и $p(x)$ оба примитивны, то $r = \pm 1$.
6. Является ли полином $p(x) = 2x^5 + 15x^4 + 9x^3 + 3$ неприводимым в $\mathbb{Z}[x]$.
7. Покажите, что если p — простое число, то полином $x^n - p$ неприводим в $\mathbb{Q}[x]$.
8. Определите, какие из следующих полиномов неприводимы в $\mathbb{Q}[x]$:
   a. $x^4 + 1$;
   b. $x^7 + 11x^3 - 33x + 22$;
   c. $x^4 + x^3 + x^2 + x + 1$;
   d. $x^3 - 7x^2 + 3x + 3$;
   e. $x^3 + x^2 - 2x - 1$.

Раздел 3.2.4

1. Примените алгоритм **PSQFF** для вычисления разложения полинома $p(x) = x^{12} - 36x^{10} + 510x^8 - 3580x^6 + 12825x^4 - 21384x^2 + 11664$ на свободные от квадратов множители. [*Указание.* Данный полином — произведение полиномов $x - 1$, $x + 1$, $(x - 2)^2$, $(x + 2)^2$, $(x - 3)^3$, $(x + 3)^3$.]
2. Вычислите разложение на свободные от квадратов множители следующих полиномов:
   a. $x^6 - x^5 - 4x^4 + 2x^3 + 5x^2 - x - 2$;
   b. $x^6 - 3x^5 + 6x^3 - 3x^2 - 3x + 2$;
   c. $x^5 - 2x^4 - 2x^3 + 4x^2 + x - 2$;
   d. $x^6 - 2x^5 - 4x^4 + 6x^3 + 7x^2 - 4x - 4$;
   e. $x^6 - 6x^5 + 12x^4 - 6x^3 - 9x^2 + 12x - 4$.

f. $x^7 - 15x^6 + 94x^5 - 318x^4 + 625x^3 - 711x^2 + 432x - 108$. (Явная демонстрация различия между разложением на неприводимые множители и разложением на свободные от квадратов множители).

3. Для каждого простого p найдите отличный от константы полином $p(x)$ над $\mathbb{Z}_p$ такой, что $p'(x) = 0$.

4. Докажите, что если $p(x)$ — полином из $F[x]$, где F — поле, содержащееся в поле комплексных чисел, то $p(x)$ не имеет кратных множителей тогда и только тогда, когда $\gcd[p(x), p'(x)] = 1$.

5. Исследуйте следующие полиномы на кратные сомножители в $\mathbb{Q}[x]$:

a. $x^3 - 3x^2 + 4$;

b. $x^3 - 2x^2 - x + 2$.

6. Докажите, что полином $p(x)$ имеет кратный сомножитель в $\mathbb{Q}[x]$ тогда и только тогда, когда $p(x)$ имеет кратный сомножитель в $\mathbb{C}[x]$.

Раздел 3.3.1

1. Каковы порядки ненулевых элементов поля $GF(7)$?

2. Докажите теорему 3.3.9.

3. Пусть p — простое число; покажите, что $x^{p^m} - x$ делит $x^{p^n} - x$ в $\mathbb{Z}_p[x]$ тогда и только тогда, когда m делит n.

4. Покажите, что если $a \in GF(q)$ и $a^r = 1$, то $a^d = 1$, где $d = \gcd[r, q-1]$.

5. Пусть $m(x) = x^2 + 1$; найдите примитивный корень β поля $\mathbb{Z}_3[x]_{m(x)}$. Найдите также минимальный полином элемента β в $\mathbb{Z}_3[x]$.

6. Докажите теорему 3.3.16.

7. Найдите все неприводимые сомножители полинома $x^8 - x$ в $\mathbb{Z}_2[x]$.

8. Теорема 3.3.8 утверждает, что у каждого конечного поля F_q есть образующая. Покажите, что если a — образующая поля F_q, то a^j также является образующей тогда и только тогда, когда $\gcd(j, q-1) = 1$. В частности, имеется всего $\phi(q-1)$ различных образующих поля F_q.

9. Как мы видели непосредственно перед теоремой 3.3.9, нахождение образующей в конечном поле F_q сильно зависит от разложения числа $p - 1$. Покажите, что существует последовательность простых чисел p, таких, что вероятность того, что случайный ненулевой элемент $a \in F_p$ является образующим,

стремится к нулю. (*Указание.* Воспользуйтесь следующими фактами: (a) теоремой Дирихле о простых числах в арифметической прогрессии, которая утверждает, что если n и k взаимно просты, то существует бесконечно много простых чисел, которые $\equiv k \pmod{n}$; (b) предыдущим упр. 8, откуда мы делаем вывод, что отношение числа всех ненулевых элементов, являющихся образующими, к числу ненулевых элементов равно $\phi(p-1)/(p-1) = \prod(1-1/d)$, где произведение берется по всем $d|(p-1)$; (c) $\prod 1/(1-1/d)$ стремится к бесконечности, когда произведение берется по всем простым числам.)

10. Используйте теорему 3.3.21, чтобы доказать, что если r — простое число, то существует $(p^r - p)/r$ различных нормированных неприводимых полиномов степени r. (Более общий результат получен в разд. 6.2.2.) Заметим, что по малой теореме Ферма $(p^r - p)/r$ — целое число.

11. Предположим, что r — степень простого числа f. Найдите простую формулу для числа нормированных неприводимых полиномов степени r над F_p.

12. Докажите, что сравнение $f(x) \equiv 0 \pmod{p}$ степени n ($n < p$, $f(x)$ нормирован) имеет n решений тогда и только тогда, когда $f(x)$ является делителем полинома $x^p - x \pmod{p}$; т.е. тогда и только тогда, когда $x^p - x = f(x)q(x) + p \cdot r(x)$, где $q(x)$ и $r(x)$ — полиномы с целыми коэффициентами и $r(x)$ — либо полином степени $< n$, либо нулевой полином.

13. Для $p = 7, 11$ и 13 найдите наименьшее положительное целое число, которое порождает ненулевые элементы поля F_p, и определите, сколько из чисел $1, 2, 3, \ldots, p-1$ является образующими.

14. Сколько элементов содержится в наименьшем расширении поля F_5, которое содержит все корни полиномов $x^2 + x + 1$ и $x^3 + x + 1$?

15. У каких полиномов в $F_p[x]$ производные тождественно равны нулю?

16. Пусть F_q — конечное поле из $q = p^r$ элементов и σ — отображение, переводящее каждый элемент в его p-ю степень: $\sigma(a) = a^p$. Докажите, что (a) σ является автоморфизмом поля F_q (т.е. σ — это взаимно однозначное отображение, сохраняющее сложение и умножение); (b) элементы поля F_q, остающиеся неподвижными под действием σ, — это в точности элементы простого поля F_p; (c) множество элементов, остающихся неподвижными при σ^j, — это поле F_{p^d}, где $d = \gcd(j, r)$.

17. Докажите, что если α — какой-либо элемент $\in F_q$, то элементы поля F_q, удовлетворяющие тому же нормированному неприводимому полиному с коэффициентами из F_q (называемые также сопряженными), — это элементы $\sigma^j(\alpha) = \alpha^{p'}$, где $p' = p^j$, а σ определено выше в упр. 16.

18. При каких условиях на p и r *каждый* элемент поля F_q, кроме 0 и 1, является образующим ненулевых элементов поля F_q, $q = p^r$? При каких условиях каждый элемент, отличный от 0, 1, является либо образующим, либо квадратом образующего?

19. Предположим, что $\alpha \in F_{p^2}$ удовлетворяет полиному x^2+ax+b, где $a, b \in F_p$.

a. Докажите, что α^p также удовлетворяет этому полиному.

b. Докажите, что если $\alpha \notin F_p$, то $a = -\alpha - \alpha^p$ и $b = \alpha^{p+1}$.

c. Докажите, что если $\alpha \notin F_p$ и $c, d \in F_p$, то $(c\alpha + d)^{p+1} = d^2 - acd + bc^2$ (и $\in F_p$).

20. Мы хотим ответить на вопрос: сколько корней имеет уравнение $x^n = 1$ над F_q? (Эти корни называются также корнями n-й степени из единицы.) Пусть a — образующий (ненулевых элементов) поля F_q. Докажите, что a^k есть корень n-й степени из единицы тогда и только тогда, когда $nk \equiv 0 \pmod{q-1}$, и что число корней n-й степени из единицы равно $\gcd(n, q-1)$. В частности, F_q имеет *примитивный* корень n-й степени из единицы (т.е. элемент ξ, такой, что степени ξ пробегают все n корней n-й степени из единицы) тогда и только тогда, когда $n|(q-1)$; если ξ — примитивный корень n-й степени из единицы в F_q, то ξ^k — также примитивный корень n-й степени тогда и только тогда, когда $\gcd(k, n) = 1$.

21. Сколько корней 15-й степени из единицы имеется в поле из 7^3 элементов?

22. Для всех a, таких, что $\gcd(a, m) = 1$, элемент a называется *квадратичным вычетом* по модулю m, если сравнение $x^2 \equiv a \pmod{m}$ имеет решение. Если оно не имеет решения, то a называется *квадратичным невычетом*. (См. также упр. 18(b) раздела 2.3.2.)

a. Докажите, что квадратичный вычет никогда не может быть образующим группы ненулевых элементов поля F_p.

b. Покажите, что если p — простое число Ферма, то любой квадратичный невычет является образующим группы ненулевых элементов поля F_p; более того, покажите, что 5 является образующим группы ненулевых элементов поля F_p, кроме случая $p = 3$, а 7 — образующим для F_p,

кроме случая $p = 3$.

23. (Вычисление квадратных корней по модулю p.) Пусть p — нечетное простое число. Рассмотрим сравнение $x^2 \equiv a \pmod p$, про которое мы знаем, что оно имеет решение (см. упр. 18(b) разд. 2.3.2); мы хотим найти такое целое число x, которое удовлетворяет этому сравнению. Нужно действовать следующим образом. Пусть k — квадратичный невычет, вычисленный нами каким-либо образом. Запишем $p-1$ в виде $2^\alpha \cdot s$, где s нечетно, и положим $u := k^s \pmod p$ и $r := a^{(s+1)/2} \pmod p$. Если $p \equiv 3 \pmod 4$, то $\alpha = 1$, $s = (p-1)/2$, $(s+1)/2 = (p+1)/4$, и мы видим, что $x = r = a^{(p+1)/4}$ — требуемый квадратный корень. В противном случае r достаточно близко подходит к квадратному корню из a и должно быть соответствующим образом подправлено (с помощью корня степени 2^α из единицы). Прежде чем осуществить эту подгонку, заметим следующее:

(i) Величина $(r^2/a)^{2^{\alpha-1}}$ является корнем степени $2^{\alpha-1}$ из единицы $\pmod p$, т.е.

$$(r^2/a)^{2^{\alpha-1}} \equiv a^{s\,2^{\alpha-1}} \equiv a^{(p-1)/2} \equiv 1 \pmod p.$$

(ii) Элемент u является примитивным корнем степени 2^α из единицы (см. выше упр. 20). Чтобы убедиться в этом, заметим прежде всего, что u — корень степени 2^α из единицы $\pmod p$, потому что

$$u^{2^\alpha} \equiv k^{s\,2^\alpha} \equiv k^{p-1} \equiv 1 \pmod p.$$

(Заметим, что $u^{2^{\alpha-1}} \equiv -1 \pmod p$, поскольку k — невычет.) Теперь, если u не примитивен, должна существовать меньшая степень u (делитель числа 2^α), дающая 1, откуда следует, что элемент u должен быть четной степенью примитивного корня степени 2^α из единицы и, таким образом, u должен быть квадратом в F_p. Однако это невозможно, так как

$$u^{(p-1)/2} \equiv k^{s(p-1)/2} \equiv -1 \pmod p,$$

поскольку s нечетно, а k — невычет.

Таким образом остается вычислить подходящую степень u^b, $0 \le b < 2^\alpha$, такую, что $u^b r$ — требуемый корень из a.

Для этого напишем $b = b_0 + 2b_1 + 4b_2 + \cdots + 2^{\alpha-2}b_{\alpha-2}$, где последовательность битов вычисляется следующим образом:

[1] Поскольку (r^2/a) — корень степени $2^{\alpha-1}$ из единицы $(\mathrm{mod}\ p)$, то, если мы возведем его в степень $2^{\alpha-2}$, результат будет равен ± 1. Если он равен -1, положим $b_0 := 1$, в противном случае положим $b_0 := 0$. В любом случае b_0 выбрано так, что $[(u^{b_0}r)^2/a]^{2^{\alpha-2}} \equiv 1\ (\mathrm{mod}\ p)$. (Напомним, что $u^{2^{\alpha-1}} \equiv -1$.)

[2] Предположим теперь, что мы вычислили $b_0, b_1, \ldots, b_{k-1}$ таким образом, что $[(u^{b_0+2b_1+\cdots+2^{k-1}b_{k-1}}r)^2/a]^{2^{\alpha-k-1}} \equiv 1\ (\mathrm{mod}\ p)$, и хотим найти b_k. Снова, как в [1], вычисляем $[(u^{b_0+2b_1+\cdots+2^{k-1}b_{k-1}}r)^2/a]^{2^{\alpha-k-2}}\ (\mathrm{mod}\ p)$, (где мы возводим теперь в степень $2^{\alpha-k-2}$), и если результат равен -1, то полагаем $b_k := 1$, в противном случае полагаем $b_k := 0$. Таким образом мы гарантируем, что $[(u^{b_0+2b_1+\cdots+2^kb_k}r)^2/a]^{2^{\alpha-k-2}} \equiv 1\ (\mathrm{mod}\ p)$. Наконец, когда мы дойдем до $k = \alpha - 2$ и найдем $b_{\alpha-2}$, получим

$$(u^{b_0+2b_1+\cdots+2^{\alpha-2}b_{\alpha-2}}r)^2/a \equiv 1 \pmod p,$$

и мы вычислили квадратный корень из a.

Примените описанную процедуру к сравнению $x^2 \equiv 17\ (\mathrm{mod}\ 89)$. См. также упр. 9 разд. 6.3.1. (См. статью Adleman L., Manders K., Miller G. On taking roots in finite fields. *Proceedings of the 18th Annual Symposium on the Foundations of Computer Science, pp. 175–178, 1977.*)

24. (Вычисление дискретных логарифмов в конечных полях.) Напомним, что если G — конечная группа, а $b \in G$ и $y \in G$ таковы, что y является степенью элемента b, то *дискретный логарифм* y по основанию b — это любое целое число x, обладающее свойством $y = b^x$.

Если F_q — поле, в котором мы работаем, то следующий алгоритм основан на предположении, что b — образующий ненулевых элементов поля F_q и что все простые сомножители числа $q-1$ маленькие (т.е. число $q-1$ «гладкое»). (См. Odlyzko A.M. Discrete logarithms in finite fields and their cryptographic significance. Advances in Cryptology, Proceedings of Eurocrypt 84, Springer-Verlag, pp. 224–314, 1985.)

Наша цель — найти значение $x\ [\mathrm{mod}\ (q-1)]$, такое, что $b^x = y$ (т.е. $0 \le x < q-1$). Если $q - 1 = \prod_p p^\alpha$ — разложение числа $q-1$ на простые сомножители, то достаточно найти $x\ (\mathrm{mod}\ p^\alpha)$ *для каждого простого числа* p, *делящего* $q-1$; мы

можем затем применить греко-китайскую теорему об остатках для вычисления $x \pmod{(q-1)}$.

1 [Предварительные вычисления.] Для *каждого* простого p, делящего $q-1$, вычисляем корни p-й степени из единицы $r_{p,k} = b^{k(q-1)/p}$ для $k = 0, 1, 2, \ldots, p-1$. (Заметим, что если b фиксировано, то эти вычисления нужно выполнить только один раз; с помощью нашей таблицы значений $\{r_{p,k}\}$ мы готовы вычислять дискретный логарифм любого $y \in F_q - \{0\}$.)

2 [Основной шаг вычисления $x \pmod{p^\alpha}$!!] Для *каждого* простого p, делящего $q-1$, выполняем следующее. Пусть $x \equiv x_0 + x_1 p + \cdots + x_{\alpha-1} p^{\alpha-1} \pmod{p^\alpha}$, $0 \le x_i < p$. Чтобы получить x_0, мы вычисляем $y^{(q-1)/p}$, являющийся p-м корнем из единицы, поскольку $y^{(q-1)} = 1$; из $y = b^x$ следует, что $y^{(q-1)/p} = b^{x(q-1)/p} = b^{x_0(q-1)/p} = r_{p,x_0}$. (Напомним, что $b^{q-1} = 1$!) Таким образом, мы сравниваем $y^{(q-1)/p}$ с $\{r_{p,k}\}_{0 \le k \le p}$ и полагаем x_0 равным тому значению k для которого $y^{(q-1)/p} = r_{p,k}$.

Чтобы получить x_1, полагаем $y := y/b^{x_0}$ $(= b^{x_1 p} \cdot b^{x_2 p^2} \ldots b^{x_{\alpha-1} p^{\alpha-1}})$. Поскольку этот новый элемент y является p-й степенью, мы имеем $y^{(q-1)/p} = 1$ (не забывайте, что $b^{q-1} = 1$) и $y^{(q-1)/p^2} = b^{(x_1 p + x_2 p \ldots)(q-1)/p} = b^{x_1 (q-1)/p} = r_{p,x_1}$. Снова мы полагаем x_1 равным тому значению k, для которого $y^{(q-1)/p^2} = r_{p,k}$. Продолжаем аналогичным образом, пока не вычислим все значения x_i для соответствующего p.

3 [Греко-китайский алгоритм.] В этом месте нам известны $x \pmod{p^\alpha}$ для каждого простого числа p, делящего $q-1$. Применим греко-китайский алгоритм, чтобы получить $x \pmod{(q-1)}$.

Раздел 3.3.2

1. Пусть α_i, $i = 0, 1, \ldots, 7$, — восемь элементов поля $GF(8)$, определенного неприводимым полиномом $x^3 + x + 1$ над $\mathbb{Z}_2$. Выпишите таблицы сложения и умножения элементов поля $GF(8)$.

2. Постройте F_9.

3. Для каждой степени $d \le 5$ найдите число неприводимых над F_2 полиномов степени d и составьте их список.

4. Для каждой степени $d \le 6$ найдите число нормированных неприводимых над F_3 полиномов степени d и для $d \le 3$ составьте их список.

5. Для каждого из следующих полей F_q, где $q = p^r$, найдите неприводимый полином с коэффициентами из простого поля, корень α которого примитивен (т.е. порождает ненулевые элементы поля F_q), и выпишите все степени α как полиномы от α степени $< r$: (a) F_4; (b) F_8; (c) F_{27}; (d) F_{25}.

Упражнения по программированию

Раздел 3.3.1

1. Реализуйте алгоритм вычисления квадратных корней по модулю p, описанный в упр. 23 этого раздела. (Воспользуйтесь системой компьютерной алгебры с теоретико-числовым пакетом.)

2. Реализуйте алгоритм вычисления дискретных логарифмов в конечных полях, описанный в упр. 24 этого раздела.

Литература

Afrati F. *Introduction to the theory of information* (in Greek). National Technical University of Athens, Greece, 1985.

Akritas A.G., Danielopoulos S.D. On the complexity of algorithms for the translation of polynomials. *Computing* **24**, 51–60, 1980.

Albert A.A. *Fundamental concepts of higher algebra.* University of Chicago Press, Chicago, 1958.

Berlekamp E.R. *Algebraic coding theory.* McGraw-Hill, New York, 1968. [Имеется перевод: Берлекэмп Э. Алгебраическая теория кодирования. М.: Мир, 1974.]

Cajori F. Horner's method of approximation anticipated by Ruffini. *American Society Bulletin* **17**, 409–414, 1911.

Childs L. *A concrete introduction to higher algebra.* Springer-Verlag, New York, 1979.

Knuth D.E. *The art of computer programming*, Vol. 2, 2nd ed. *Seminumerical algorithms.* Addison-Wesley, Reading, MA, 1981. [Имеется перевод

первого издания: Кнут Д. *Искусство программирования для ЭВМ.* Т. 2. — М.: Мир, 1977.]

Netto E. *Vorlesungen über Algebra.* Erster Band. Teubner, Leipzig, 1896.

Sims C.C. *Abstract algebra, a computational approach.* Wiley, New York, 1984.

Taub H. *Digital circuits and microprocessors.* McGraw-Hill, New York, 1982.

Wang S.P., Trager B.T. New algorithms for polynomial square-free decompositions over the integers. *SIAM Journal on Computing* **8**, 300–305, 1979.

Yun D.Y.Y. On square-free decomposition algorithms. *Proceedings of the 1976 SYMSAC* (Yorktown Heights, NY), 26–35, 1976.

Часть III
ПРИЛОЖЕНИЯ И СОВРЕМЕННЫЕ МЕТОДЫ

Эта часть книги посвящена приложениям материала, изложенного в ч. I и II, и современным методам. В ней рассматриваются следующие темы:

Глава 4. Коды, исправляющие ошибки, и криптография.

Глава 5. Наибольшие общие делители полиномов над целыми числами и последовательности полиномиальных остатков.

Глава 6. Разложение на множители полиномов над целыми числами.

Глава 7. Отделение и аппроксимация вещественных корней полиномиальных уравнений.

Читателю следует помнить, что разд. 7.2 зависит от разд. 5.2.

Глава 4

Коды, исправляющие ошибки, и криптография

Кодирование и декодирование информации для поддержания секретности называется *криптологией*. Однако имеется много других типов кодов, где секретность не представляет интереса. Примерами могут служить zip-коды, используемые почтовыми учреждениями, коды ASCII или EBCDIC, используемые для преобразования символов алфавита в двоичную форму для представления в ЭВМ, и Универсальный промышленный код, ряд черных вертикальных линий, содержащих информацию об изделии, который можно обнаружить на многих товарах.

В этой главе мы будем рассматривать коды, предназначенные защищать передаваемую информацию как от искажения этой информации, так и от несанкционированных получателей.

4.1

Коды, исправляющие ошибки, — основные понятия

Ошибки при передаче данных по каналу могут возникать по самым разным причинам, например из-за отказа аппаратуры, «шумов», или «сбоев», которым подвержено чувствительное и сложное электронное оборудование. Для защиты от ошибок мы можем кодировать посылаемую информацию и декодировать ее на другом конце таким образом, чтобы максимизировать вероятность исправления или по крайней мере обнаружения таких ошибок. Удачные схемы кодирования часто опираются на абстрактную алгебру.

Алгебраическая теория кодирования насчитывает к настоящему времени более 30 лет. Изначально она возникла в ответ на знаменитую теорему Шеннона о кодировании (Shannon, 1948, 1949), которая ограничивала эффективность ошибки, которая может получиться на дискретном канале, но почти не давала указаний, как эта эффективность может получиться. То есть теорема Шеннона гарантирует существование кодов, которые могут передавать информацию со скоростью, близкой к пропускной способности, с произвольно малой вероятностью ошибки.

В этом разделе мы описываем два семейства кодов: первый — способный обнаруживать и исправлять одну ошибку, был разработан в 1950 г. Хэммингом, в то время как второй, который может обнаруживать и исправлять две ошибки, был разработан на 10 лет позже Боузом, Чоудхури и Хоквингемом (БЧХ для краткости) (Bose, Chaudhuri, 1960; Hocquenghem, 1959). Главным инструментом в коде Хэмминга служат линейная алгебра (читатель может просмотреть приложение в конце книги) и язык смежных классов, в то время как в БЧХ-кодах важную роль играют конечные поля. (В изложении мы следуем работе (Afrati, 1985); см. также (Berlekamp, 1968; Blake, 1979; Levinston, 1970; MacWilliams, 1977; Peterson и др., 1972 и Wakery, 1978).)

На рис. 4.1.1 мы описываем нашу модель системы связи. Сообщения проходят по системе от отправителя (называемого также *источником*, или *передатчиком*). Каналом могут служить воздух или провода.

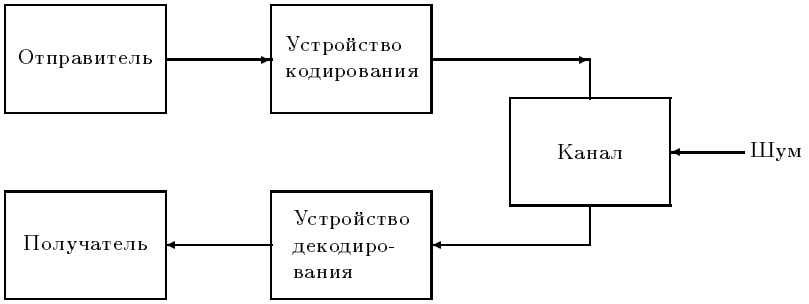


Рис. 4.1.1.
Модель системы связи.

В противоположность непрерывным источникам, таким, как радио, мы рассматриваем только источники с конечным числом дискретных сигналов. Более точно, мы имеем дело с *двоичными кодами*, предполагая, что любая передаваемая нами информация может быть представлена в виде последовательности двоичных цифр; это — *сообщение*. Более того, мы предполагаем, что наш канал — это *двоичный симметричный канал* приведенного на рис. 4.1.2 вида.

Это значит, что если p — вероятность правильного получения двоичного сигнала, то $1 - p$ — вероятность приема с ошибкой. Мы предполагаем, что возникающие при передаче ошибки независимы и что изменилась только одна двоичная цифра, в то время как соседние цифры остались нетронутыми. (В действительности это не

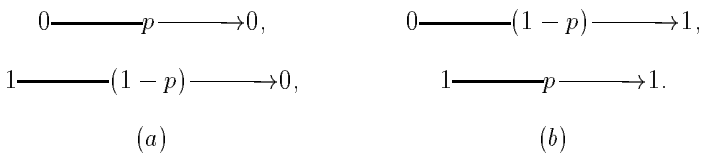


Рис. 4.1.2.
Два вида двоичных симметричных каналов.

так, и внимание уделяется кодам, имеющим дело с каналами, в которых ошибки встречаются *пакетами*, или *кластерами*.) Наконец, мы предполагаем, что вероятность того, что «0» преобразуется в «1», равна вероятности того, что «1» преобразуется в «0», и что вероятность любого такого события относительно мала, скажем 0.01. (В действительности единицы преобразуются чаще, чем нули.)

Хотя мы не в состоянии помешать каналу допускать такие ошибки, мы можем принять некоторые меры, чтобы защититься от них. Идея состоит в том, чтобы к k *информационным* цифрам, которые мы хотим передать, добавить r *проверочных* цифр, а затем передавать весь блок из $n = k + r$ цифр. Предполагается, естественно, что канал изменит относительно немного из этих n передаваемых цифр, так что пользователь на основе r проверочных цифр будет иметь достаточно информации, чтобы обнаружить и исправить ошибки канала. (Это — пример *избыточности*, т.е. передачи более чем k информационных цифр, чтобы улучшить надежность процесса передачи.) Конечно, должно существовать правило выбора r проверочных цифр, и этот процесс носит название *кодирования*. Мы назовем $R = k/n$ *скоростью передачи информации* кода.

Определение 4.1.1. k-мерное подпространство C векторного пространства $\mathbf{V}_n$ всех двоичных n-кортежей над $GF(2)$ называется (n, k)-*кодом*.

Любая последовательность из n цифр, которая может быть передана, называется *кодовым словом*. Очевидно, имеется 2^n последовательностей, или векторов, длины n, но только 2^k из них являются кодовыми словами, потому что r проверочных цифр в каждом кодовом слове полностью определяются k информационными цифрами; мы будем называть *кодом* множество этих 2^k векторов длины n. Для любого передаваемого кодового слова, учитывая тот факт, что мы имеем дело с зашумленным каналом, может быть получен любой из

2^n векторов длины n; тогда пользователь должен решить, какое из 2^k возможных кодовых слов передавалось.

Определение 4.1.2. Если $\mathbf{u}$ — переданное кодовое слово, а $\mathbf{v}$ — полученный вектор, то $\mathbf{e} = \mathbf{u} - \mathbf{v} = (e_1, e_2, \ldots, e_n)$ называется *вектором ошибок*.

Получив $\mathbf{v}$, пользователь должен решить, какое кодовое слово было передано, определив наиболее вероятную ошибку. Если все кодовые слова длины n равновероятны, то эта процедура минимизирует вероятность некорректного декодирования. *Декодирование по максимуму правдоподобия* просто предполагает наиболее вероятную ситуацию, а именно что встречается наименьшее число ошибок.

Определение 4.1.3. *Расстояние Хэмминга* $d(\mathbf{u}, \mathbf{v})$ между двумя векторами длины n — это число координат, в которых $\mathbf{u}$ и $\mathbf{v}$ различаются. *Вес Хэмминга* $w(\mathbf{u})$ вектора длины n — это число отличных от нуля координат u_i, и очевидно, что $w(\mathbf{u}) = d(\mathbf{u}, \mathbf{0})$.

Например, если $\mathbf{u} = (1, 0, 1, 1, 0)$ и $\mathbf{v} = (1, 1, 0, 0, 1)$, то $d(\mathbf{u}, \mathbf{v}) = 4$. Можно показать, что расстояние Хэмминга — это метрика на векторном пространстве $\mathbf{V}_n$ над $GF(2)$ (см. упр. 1 к этому разделу), т.е. для любых $\mathbf{u}_1$, $\mathbf{u}_2$ из $\mathbf{V}_n$ выполнены следующие условия:

a. $d(\mathbf{u}_1, \mathbf{u}_2) \geq 0$ и $d(\mathbf{u}_1, \mathbf{u}_2) = 0$ тогда и только тогда, когда $\mathbf{u}_1 = \mathbf{u}_2$.

b. $d(\mathbf{u}_1, \mathbf{u}_2) = d(\mathbf{u}_2, \mathbf{u}_1)$.

c. $d(\mathbf{u}_1, \mathbf{u}_3) \leq d(\mathbf{u}_1, \mathbf{u}_2) + d(\mathbf{u}_2, \mathbf{u}_3)$ (неравенство треугольника).

Аналогично можно показать, что вес Хэмминга — это *норма* на векторном пространстве $\mathbf{V}_n$ над $GF(2)$, т.е. для любых $\mathbf{u}$, $\mathbf{v}$ из $\mathbf{V}_n$ и λ из в $\mathbf{R}$ имеют место следующие утверждения:

a. $w(\mathbf{u}) \geq 0$ и $w(\mathbf{u}) = 0$ тогда и только тогда, когда $\mathbf{u} = \mathbf{0}$.

b. $w(\mathbf{u} + \mathbf{v}) \leq w(\mathbf{u}) + w(\mathbf{v})$.

c. $w(\lambda \mathbf{u}) = |\lambda| w(\mathbf{u})$.

Для понимания следующей теоремы требуется некоторое знакомство с теорией вероятностей.

Теорема 4.1.4. Пусть C — код для двоичного симметричного канала, и предположим, что C содержит s равновероятных кодовых слов длины n. Тогда *декодирование в ближайшее кодовое слово* является декодированием по максимуму правдоподобия, если вероятность p правильной передачи $> 1/2$.

Доказательство. Пусть $d(\mathbf{u},\mathbf{v})$ — расстояние между кодовым словом $\mathbf{u}$ и полученным вектором $\mathbf{v}$. Для того чтобы пользователь мог получить вектор $\mathbf{v}$, если передавался $\mathbf{u}$, должны встретиться $d(\mathbf{u},\mathbf{v})$ ошибок. Вероятность этого события равна

$$\mathrm{prob}(\mathbf{v}|\mathbf{u}) = (1-p)^{d(\mathbf{u},\mathbf{v})}p^{n-d(\mathbf{u},\mathbf{v})},$$

где p — вероятность правильной передачи. Положим $d_i = d(\mathbf{u}_i,\mathbf{v})$, $i = 1,2$, и сравним вероятности $\mathrm{prob}(\mathbf{v}|\mathbf{u}_1)$ и $\mathrm{prob}(\mathbf{v}|\mathbf{u}_2)$ для двух кодовых слов $\mathbf{u}_1$ и $\mathbf{u}_2$. Очевидно, что

$$\frac{\mathrm{prob}(\mathbf{v}|\mathbf{u}_1)}{\mathrm{prob}(\mathbf{v}|\mathbf{u}_2)} = \frac{(1-p)^{d_1}p^{n-d_1}}{(1-p)^{d_2}p^{n-d_2}} = \left(\frac{p}{1-p}\right)^{d_2-d_1},$$

а поскольку мы всегда предполагаем, что $p > 1/2$, то

$$\mathrm{prob}(\mathbf{v}|\mathbf{u}_1) > \mathrm{prob}(\mathbf{v}|\mathbf{u}_2) \quad \text{тогда и только тогда, когда} \quad d_1 < d_2.$$

Поэтому вероятность $\mathrm{prob}(\mathbf{v}|\mathbf{u})$ максимальна, когда $\mathbf{u}$ — ближайшее к $\mathbf{v}$ кодовое слово. Если имеется несколько кодовых слов с таким свойством, то мы можем выбирать одно из них произвольным образом. □

Пример. Предположим, что код состоит из четырех следующих кодовых слов:

$$\mathbf{u}_1 = (0,0,0,0,0),$$
$$\mathbf{u}_2 = (1,0,0,1,1),$$
$$\mathbf{u}_3 = (1,1,1,0,0),$$
$$\mathbf{u}_4 = (0,1,1,1,1).$$

Если получен вектор $\mathbf{v} = (0,1,0,1,1)$, то он будет декодирован как $\mathbf{u}_4$, потому что $d(\mathbf{u}_4,\mathbf{v}) = 1$ и $d(\mathbf{u}_i,\mathbf{v}) > 1$ для $i = 1,2,3$. Аналогично принятый вектор $\mathbf{w} = (0,0,1,1,0)$ может быть декодирован как $\mathbf{u}_1$ или $\mathbf{u}_4$, потому что $d(\mathbf{u}_1,\mathbf{w}) = d(\mathbf{u}_4,\mathbf{w}) = 2$, а $d(\mathbf{u}_2,\mathbf{w}) = d(\mathbf{u}_3,\mathbf{w}) = 3$.

Поскольку любой вектор $\mathbf{v}$, поступивший к получателю, декодируется в ближайшее кодовое слово (ближайшее относительно расстояния Хэмминга), то естественно предположить, что хороший код — это код, в котором кодовые слова находятся «далеко» друг от друга. Это действительно так, потому что если расстояние между кодовыми словами велико, то чтобы кодовое $\mathbf{u}$ преобразовалось в (полученный) вектор $\mathbf{v}$, который ближе к другому кодовому слову $\mathbf{u}'$, чем к $\mathbf{u}$, должно встретиться много ошибок. Имеет место следующая

Лемма 4.1.5. Пусть C — двоичный код с s кодовыми словами $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_s$ длины n. Этот код может исправлять все комбинации не более чем t ошибок тогда и только тогда, когда

$$d(\mathbf{u}_i, \mathbf{u}_j) \geq 2t + 1$$

для всех $i \neq j$.

Доказательство. Мы докажем лемму только в одном направлении, оставив остальное читателю. Предположим, что расстояние между любыми двумя кодовыми словами не меньше, чем $2t+1$. Если $\mathbf{u}$ — переданное кодовое слово, а $\mathbf{v}$ — полученный вектор с $t' \leq t$ ошибками, т.е. $d(\mathbf{u}, \mathbf{v}) = t'$, то по неравенству треугольника расстояние $d(\mathbf{u}', \mathbf{v})$ от $\mathbf{v}$ до любого другого кодового слова $\mathbf{u}'$ удовлетворяет соотношению

$$d(\mathbf{u}', \mathbf{v}) \geq d(\mathbf{u}', \mathbf{u}) - d(\mathbf{u}, \mathbf{v}) \geq 2t + 1 - t = t + 1 > t > t' = d(\mathbf{u}, \mathbf{v}),$$

т.е. всегда больше, чем $d(\mathbf{u}, \mathbf{v})$, и если $t' \leq t$, то вектор $\mathbf{v}$ будет декодирован в правильное кодовое слово $\mathbf{u}$. □

Определение 4.1.6. Кодовое расстояние $d_{\min}$ двоичного кода C дается формулой $d_{\min} = \min d(\mathbf{u}, \mathbf{v})$, где минимум берется по всем векторам $\mathbf{u}, \mathbf{v}$ в C, $\mathbf{u} \neq \mathbf{v}$.

Если положить $d = d_{\min}$, то получаем другую характеристику кода, аналогичную приведенной в определении 4.1.1, а именно (n, k, d). Более того, имеет место следующая

Теорема 4.1.7. Кодовое расстояние двоичного кода C равно наименьшему весу ненулевых кодовых слов.

Доказательство. Доказательство непосредственно следует из того, что $d(\mathbf{u}, \mathbf{v}) = d(\mathbf{u} - \mathbf{v}, \mathbf{0}) = w(\mathbf{u} - \mathbf{v})$. □

Определение 4.1.8. Пусть $\mathbf{V}_n$ — векторное пространство всех двоичных n-кортежей над $GF(2)$. Тогда множество $S_r(\mathbf{y}) = \{x \in \mathbf{V}_n : d(\mathbf{x}, \mathbf{y}) \leq r\}$ называется *шаром радиуса* r с центром в $\mathbf{y} \in \mathbf{V}_n$.

Лемма 4.1.9. Двоичный код C с кодовым расстоянием $d_{\min}$ может исправлять все комбинации от 1 до t ошибок и может обнаруживать все комбинации от $t + 1$ до $t + s$ ошибок тогда и только тогда, когда $d_{\min} \geq 2t + s$.

Доказательство. Первая часть доказана в лемме 4.1.5. Оставляем читателю в качестве упражнения доказательство того, что может быть обнаружена любая комбинация $t + s$ ошибок. □

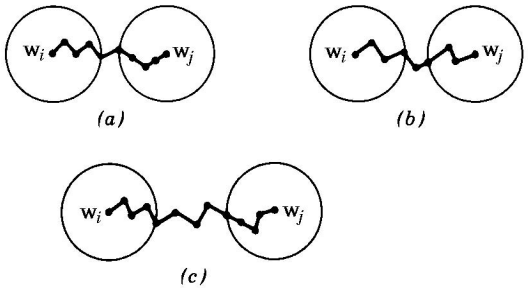


Рис. 4.1.3.
Связь между расстоянием и корректирующей способностью кода.

Геометрическая интерпретация лемм 4.1.5 и 4.1.9 дана на рис. 4.1.3. Например, если $d(\mathbf{w}_i, \mathbf{w}_j) = 9$, как на рис. 4.1.3(a), то изменение четырех или менее цифр в кодовом слове $\mathbf{w}_i$ приведет к тому, что полученный вектор будет ближе к $\mathbf{w}_i$, чем к $\mathbf{w}_j$. Если $d(\mathbf{w}_i, \mathbf{w}_j) = 8$, как на рис. 4.1.3(b), то ошибки в трех или менее цифрах будут исправлены, но четыре ошибки могут привести к тому, что полученный вектор будет равноудален как от $\mathbf{w}_i$, так и от $\mathbf{w}_j$ и т.д.

Другое геометрическое представление двоичного кода — рассматривать векторы длины n как вершины n-мерного гиперкуба (см. рис. 4.1.4). Изменение цифры в двоичном векторе соответствует переходу к соседней вершине, при этом движение выполняется вдоль ребра, параллельного направлению, соответствующему позиции изменившейся цифры. Чтобы получить код с кодовым расстоянием d, выбираем те вершины гиперкуба, которые не могут быть соединены путем, состоящим менее чем из d ребер. На рис. 4.1.4 код, состоящий из кодовых слов $(0,0,1)$, $(0,1,0)$ и $(1,1,1)$, имеет $d_{\min} = 2$.

Интересно узнать, сколько кодовых слов может иметь код, если он способен исправлять все комбинации t или менее ошибок. Верхнюю границу числа кодовых слов дает следующая

Теорема 4.1.10 (верхняя граница Хэмминга числа кодовых слов). Если двоичный код C, имеющий s кодовых слов длины n, может исправлять все комбинации t или менее ошибок, то

$$s \leq \frac{2^n}{\sum_{0 \leq i \leq t} \binom{n}{i}}.$$

Доказательство. Пусть $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_s$ — кодовые слова длины n в C. Для каждого кодового слова рассмотрим шар $S_t(\mathbf{u}_i)$ радиуса t с

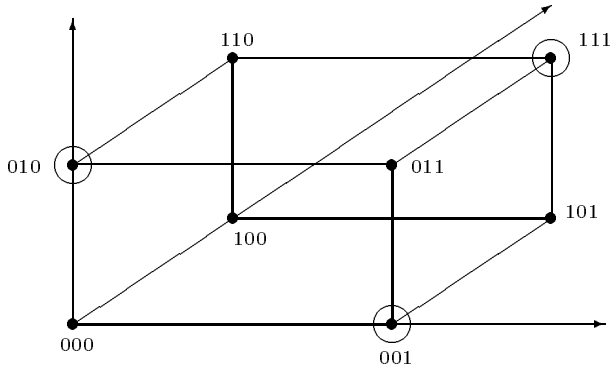


Рис. 4.1.4.
Двоичные кодовые слова как вершины гиперкуба.

центром $\mathbf{u}_i$ (определение 4.1.8). Поскольку код исправляет до t ошибок, шары не пересекаются. Поскольку шар с центром $\mathbf{u}_i$ содержит все векторы, которые отличаются от $\mathbf{u}_i$ в $0, 1, 2, \ldots, t$ позициях, мы видим, что число этих векторов равно

$$1 + n + \binom{n}{2} + \cdots + \binom{n}{t} = \sum_{0 \le i \le t} \binom{n}{i}.$$

Поэтому во всех s шарах содержится

$$s \sum_{0 \le i \le t} \binom{n}{i}$$

векторов, и доказательство завершается, если мы заметим, что общее число векторов внутри всех шаров не может превосходить 2^n — общее число двоичных векторов длины n. □

Если в теореме 4.1.10 имеет место равенство, то код называется *совершенным*. Отметим также, что не для всех троек t, n и s, определенных этой теоремой, непременно существует код с s кодовыми словами длины n, который может исправлять t ошибок. Например, положим $t = 1$ и $n = 4$; тогда по теореме 4.1.10 $s = 3$, но кода с такими параметрами не существует. (Проверьте это.) Когда такие коды существуют, другой их характеристикой является (n, s, t).

4.1.1. Коды Хэмминга

Подведем итог рассмотренному. Мы рассмотрели двоичный симметричный канал и код, состоящий из 2^k кодовых слов длины n,

скорость передачи информации которого равна $R = k/n$. Чтобы получить высокую скорость передачи информации, нам хотелось бы иметь как можно больше кодовых слов. В то же время мы хотели бы исправлять как можно больше ошибок. Однако эти два желания противоречат друг другу, потому что чем больше у нас кодовых слов, тем ближе они будут друг к другу и, согласно лемме 4.1.5, тем меньше будет способность кода исправлять ошибки. Например, если $n = 5$ и мы хотим исправлять не более одной ошибки, то мы можем найти код с четырьмя кодовыми словами, а именно $(0,0,0,0,0)$, $(1,0,0,1,1)$, $(1,1,1,0,0)$, $(0,1,1,1,1)$. Однако если $n = 5$ и мы хотим исправлять две ошибки, то мы не можем найти больше двух слов, $(0,0,0,0,0)$, $(1,1,1,1,1)$. Теорема 4.1.10 показывает, что требование увеличения способности исправлять ошибки для кода фиксированной длины n приводит к уменьшению максимального возможного числа кодовых слов.

До сего времени мы полностью игнорировали одну проблему — насколько легко может быть осуществлено декодирование. Декодер в ближайшее кодовое слово, рассматриваемый нами ранее, декодирует полученный вектор $\mathbf{v}$ в кодовое слово $\mathbf{u}$, расстояние $d(\mathbf{v},\mathbf{u})$ до которого минимально. Однако, чтобы сделать это, декодер должен содержать таблицу всех кодовых слов, и когда вектор $\mathbf{v}$ поступает на вход, он определяет, к какому кодовому слову $\mathbf{u}$ вектор $\mathbf{v}$ ближе всего. Однако такой декодер требует огромной памяти, и большое время декодирования сделает декодирование в реальном времени в большинстве случаев неприемлемым.

Далее мы развиваем теорию линейных (термин будет определен ниже) двоичных кодов, которые имеют много желательных характеристик в отношении сложности кодирования и декодирования.

Один из простейших примеров линейного двоичного кода — $(n,1)$-код *с повторением*, т.е. $k = 1$ и $r = n - k$. Другими словами, этот код является 1-мерным подпространством всех двоичных n-кортежей, которое содержит два кодовых слова: $(0,0,0,\ldots,0)$, вектор из n нулей, и $(1,1,1,\ldots,1)$, вектор из n единиц. Получив вектор длины n, пользователь может решить, какое кодовое слово было передано, использовав различные правила; например, если число полученных нулей/единиц больше, чем число полученных единиц/нулей, то пользователь может считать, что передавалось кодовое слово, состоящее из всех нулей/единиц. Эта схема обладает большой способностью обнаружения и исправления ошибок, но скорость передачи информации у нее очень низкая.

Другая крайность — $(n,n-1)$-коды *с одной проверкой на четность*,

которые содержат только один контрольный бит, $r = 1$. То есть эти коды имеют очень высокую скорость передачи информации, но нулевую корректирующую способность; они могут только обнаруживать *нечетное* число ошибок. (Фактически любое нечетное число ошибок неотличимо от единичной ошибки.) Контрольный бит, который называется также *битом четности*, определяется как сумма $n-1$ информационных цифр по модулю 2. В *проверке на четность/нечетность* последняя цифра выбирается так, чтобы общее число единиц было четно/нечетно. Например, в коде с проверкой на четность кодовое слово для 1101 равно 11011. Отметим, что любое четное число ошибок канала не может быть обнаружено. (Такие коды используются для хранения данных на магнитных лентах.)

Нас интересуют коды со средней скоростью передачи информации и средними способностями исправления ошибок.

Рассмотрим описанный выше код с одной проверкой на четность. Заметим, что $n-1$ информационных битов можно выбрать произвольно, в то время как n-й бит, проверочный разряд, — это линейная комбинация первых $n-1$ битов и добавляется, чтобы помочь нам обнаруживать возможные ошибки передачи. Если обозначить информационные биты через m_i, а проверочные — через c_i, то в коде с единственной проверкой на четность

$$c_1 = m_1 + m_2 + \cdots + m_{n-1}.$$

Обобщим эту идею следующим образом. Пусть $m_1, m_2, \ldots, m_k$ суть k информационных битов, и предположим, что мы добавили r проверочных битов, чтобы сформировать кодовое слово длины $n = k + r$. (Это процедура кодирования.) В то время как каждый из k информационных битов формируется независимо от предыдущих, r проверочных битов являются линейными комбинациями информационных, т.е.

$$\begin{aligned}
c_1 &= h_{11}m_1 + h_{12}m_2 + \cdots + h_{1k}m_k, \\
c_2 &= h_{21}m_1 + h_{22}m_2 + \cdots + h_{2k}m_k, \\
&\cdots\cdots \\
c_r &= h_{r1}m_1 + h_{r2}m_2 + \cdots + h_{rk}m_k,
\end{aligned}$$

или, поскольку сложение в $GF(2)$ совпадает с вычитанием,

$$\begin{aligned}
h_{11}m_1 + h_{12}m_2 + \cdots + h_{1k}m_k + c_1 &= 0, \\
h_{21}m_1 + h_{22}m_2 + \cdots + h_{2k}m_k + c_2 &= 0, \\
&\cdots\cdots \\
h_{r1}m_1 + h_{r2}m_2 + \cdots + h_{rk}m_k + c_r &= 0.
\end{aligned}$$

Выписанная система уравнений может быть также записана в виде

$$\overbrace{\begin{bmatrix} h_{11} & h_{12} & \dots & h_{1k} & 1 & 0 & \dots & 0 \\ h_{21} & h_{22} & \dots & h_{2k} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{r1} & h_{r2} & \dots & h_{rk} & 0 & 0 & \dots & 1 \end{bmatrix}}^{\text{(Матрица } \mathbf{H}\text{)}\ n} \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_k \\ c_1 \\ c_2 \\ \vdots \\ c_r \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

или

$$\mathbf{H}\mathbf{v}^T = \mathbf{0}^T, \quad \text{или} \quad \mathbf{v}\mathbf{H}^T = \mathbf{0}, \tag{$*$}$$

где $\mathbf{v} = (m_1, m_2, \dots, m_k, c_1, c_2, \dots, c_r)$ — кодовое слово, определенное предыдущей системой уравнений, а $\mathbf{v}^T$ — транспонированный вектор. Более точно, вектор $\mathbf{v}$ является кодовым словом кода, определенного матрицей $\mathbf{H}$, тогда и только тогда, когда $\mathbf{v}$ удовлетворяет системе $(*)$. Выписанные уравнения называются *проверочными уравнениями*, а $r \times n$-матрица $\mathbf{H}$ называется *проверочной матрицей*.

Если $\mathbf{v}_1$ и $\mathbf{v}_2$ — два кодовых слова, то

$$\mathbf{H}(\mathbf{v}_1^T + \mathbf{v}_2^T) = \mathbf{H}\mathbf{v}_1^T + \mathbf{H}\mathbf{v}_2^T = \mathbf{0}^T + \mathbf{0}^T = \mathbf{0}^T,$$

из чего мы делаем вывод, что $\mathbf{v}_1 + \mathbf{v}_2$ — также кодовое слово. Поэтому кодовые слова образуют подпространство векторного пространства $\mathbf{V}_n$. Мы говорим, что код *линейный* тогда и только тогда, когда кодовые слова образуют подпространство пространства $\mathbf{V}_n$.

Поскольку кодовые слова образуют подпространство, они могут быть выражены в виде линейных комбинаций k линейно независимых векторов, составляющих базис этого подпространства. Эти k линейно независимых векторов можно рассматривать как строки $\mathbf{g}_i$ матрицы $\mathbf{G}$ размера $k \times n$, называемой *порождающей матрицей* рассматриваемого кода. Каждая строка $\mathbf{g}_i$ сама является кодовым словом, а значит,

$$\mathbf{g}_i\mathbf{H}^T = \mathbf{0},$$

откуда мы делаем вывод, что

$$\mathbf{G}\mathbf{H}^T = \mathbf{0}.$$

Пример. Предположим, что для кода с $n = 7$ и $k = 4$ мы имеем следующие проверочные уравнения:

$$\begin{aligned} c_1 &= m_1 + m_2 \quad + m_3, \\ c_2 &= m_1 + m_2 + \qquad m_4, \\ c_3 &= m_1 \qquad + m_3 \; + m_4. \end{aligned}$$

Тогда проверочная матрица $\mathbf{H}$ имеет вид

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Порождающая матрица $\mathbf{H}$ этого кода (см. теорему 4.1.11 ниже) — это

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Кодовые слова — это 2^4 линейных комбинаций строк матрицы $\mathbf{G}$.

Очевидно, что векторное пространство, получающееся из строк матрицы, не изменится, если мы выполним любую из следующих элементарных операций со строками:

1. Поменять любые две строки местами.

2. Умножить любую строку на ненулевой элемент.

3. Заменить строку ее суммой с любой другой строкой.

(В двоичном случае вторая операция бессмысленна, потому что единственный ненулевой элемент — это 1.)

Поскольку мы исследуем те свойства линейных кодов, которые связаны со способностью исправлять ошибки, заметим, что два кода, отличающиеся только порядком расположения цифр, имеют одну и ту же вероятность ошибки, и такие коды называются *эквивалентными*. Более точно, если $\mathbf{V}$ — векторное пространство строк матрицы $\mathbf{G}$, то код $\mathbf{V}'$ эквивалентен $\mathbf{V}$ тогда и только тогда, когда $\mathbf{V}'$ — векторное пространство строк матрицы $\mathbf{G}'$, полученной переупорядочением столбцов матрицы $\mathbf{G}$. Таким образом переупорядочение столбцов порождающей матрицы приводит к порождающей матрице эквивалентного кода. Более того, любая из перечисленных выше трех элементарных операций, выполненная над строками матрицы $\mathbf{G}$,

приводит к матрице $\mathbf{G}''$ с тем же самым векторным пространством, и поэтому $\mathbf{G}$ и $\mathbf{G}''$ — порождающие одного и того же кода. Объединяя эти два случая, видим, что если матрица $\mathbf{G}'$ получена из матрицы $\mathbf{G}$ последовательностью элементарных операций над строками и переупорядочением столбцов, то $\mathbf{G}$ и $\mathbf{G}'$ — порождающие матрицы эквивалентных кодов.

Обычно мы хотим, чтобы порождающая матрица была представлена в следующем виде:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & p_{11} & \dots & p_{1,n-k} \\ 0 & 1 & 0 & \dots & 0 & p_{21} & \dots & p_{2,n-k} \\ & & & \ddots & & & \ddots & \\ 0 & 0 & 0 & \dots & 0 & p_{k1} & \dots & p_{k,n-k} \end{bmatrix},$$

т.е. $\mathbf{G} = [\mathbf{I}_k \mathbf{P}]$, где $\mathbf{I}_k$ — единичная $k \times k$-матрица, а $\mathbf{P}$ — произвольная $k \times (n-k)$-матрица. Пользуясь элементарными операциями над строками, мы всегда можем привести порождающую матрицу к такому виду, и это лучше всего иллюстрируется приводимым ниже примером.

Пример. Предположим, что мы имеем порождающую матрицу

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

На каждом шаге мы выполняем операции, необходимые для того, чтобы привести один из столбцов к требуемому виду. Мы начинаем слева направо.

Для первого столбца мы прибавим первую строку ко второй и заменим вторую строку суммой:

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

Для второго столбца переставим вторую строку с третьей:

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

и заменим первую строку суммой первой и второй строк:

$$\begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

Для третьего столбца заменим первую и вторую строки их суммами с третьей строкой:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

Таким образом мы убеждаемся, что последовательным применением элементарных операций мы всегда можем привести порождающую матрицу $\mathbf{G}$ к виду $\mathbf{G} = [\mathbf{I}_k \mathbf{P}]$.

Пусть теперь $\mathbf{a} = (a_1, a_2, \ldots, a_k)$ — вектор размерности k, и рассмотрим произведение $\mathbf{a}$ на порождающую матрицу $\mathbf{G} = [\mathbf{I}_k \mathbf{P}]$

$$\mathbf{b} = \mathbf{aG} = (a_1, a_2, \ldots, a_k, c_1, c_2, \ldots, c_{n-k}),$$

где

$$c_j = \sum_{1 \le i \le k} a_i p_{ij}. \qquad (**)$$

Заметим, что если мы рассматриваем вектор $\mathbf{a}$ как вектор с информационными битами, то в кодовом слове $\mathbf{b}$ кода, полученного из $\mathbf{G}$, первые k битов информационные, а остальные $n-k$ компонент — линейные комбинации первых k. Таким образом кодирование чрезвычайно просто: достаточно умножить информационный вектор на порождающую матрицу. *Упорядоченный* код — это код, состоящий из кодовых слов, в каждом из которых первые k битов — информационные, а остальные — проверочные. Из сказанного выше следует, что любой линейный код эквивалентен упорядоченному. Мы говорим, что линейный код с кодовыми словами длины n имеет размерность k тогда и только тогда, когда он имеет k информационных битов, и, как мы уже видели, мы называем такие коды (n, k)-кодами.

Подводя итоги сказанному, мы видим, что двоичный линейный код длины n и размерности k — это k-мерное подпространство n-мерного векторного пространства над $GF(2)$. Таким образом, код полностью определяется порождающей матрицей $\mathbf{G}$, строки которой суть базисные векторы этого подпространства. В качестве альтернативы, линейный код C может быть полностью определен проверочной матрицей $\mathbf{H}$, которая удовлетворяет уравнению $\mathbf{Hv}^T = \mathbf{0}^T$, или

$\mathbf{v}\mathbf{H}^T = \mathbf{0}$, для любого кодового слова $\mathbf{v}$. Мы говорим, что векторное пространство, определяющее код, *ортогонально* векторному пространству, определенному строками матрицы $\mathbf{H}$. Векторное пространство, определенное строками матрицы $\mathbf{H}$, можно также рассматривать как определяющее линейный код C': $\mathbf{H}$ теперь — порождающая матрица, а $\mathbf{G}$ — проверочная. Коды C и C' называются *дуальными* кодами, и если C есть (n, k)-код, то C' является $(n, n-k)$-кодом.

Теорема 4.1.11. Если V — код с порождающей матрицей $\mathbf{G} = [\mathbf{I}_k \mathbf{P}]$, где $\mathbf{I}_k$ — единичная $k \times k$-матрица, а $\mathbf{P}$ — произвольная $k \times (n-k)$-матрица, то проверочная матрица этого кода равна $\mathbf{H} = [\mathbf{P}^T \mathbf{I}_{n-k}]$.

Доказательство. Если $\mathbf{v} = (a_1, a_2, \ldots, a_k, c_1, c_2, \ldots, c_{n-k})$ — кодовое слово, то

$$\mathbf{v}\mathbf{H}^T = \mathbf{0} = (\sum_i a_i p_{i1} + c_1, \sum_i a_i p_{i2} + c_2, \ldots, \sum_i a_i p_{i,n-k} + c_{n-k}),$$

что в точности совпадает с $(**)$. (Связь между $\mathbf{G}$ и $\mathbf{H}$ станет понятней, если заметить, что в обоих случаях p_{ij} — это коэффициент при i-м информационном бите в сумме, определяющей j-й проверочный бит c_j.) □

Пример. Для порождающей матрицы

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} = [\mathbf{I}_3 \mathbf{P}]$$

находим проверочную матрицу

$$\mathbf{H} = [\mathbf{P}^T \mathbf{I}_2] = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

Кодовые слова — это

$$00000\ 10011\ 01010\ 11001\ 00101\ 10110\ 01111\ 11100.$$

Теорема 4.1.12. Код с проверочной матрицей $\mathbf{H}$ имеет минимальный вес (а следовательно, кодовое расстояние) w тогда и только тогда, когда любое множество из $w-1$ или меньшего числа столбцов матрицы $\mathbf{H}$ линейно независимо.

Доказательство. Мы докажем теорему в одном направлении, а остальную часть оставим читателю. Вектор $\mathbf{v} = (a_1, a_2, \ldots, a_n)$ является кодовым словом тогда и только тогда, когда $\mathbf{v}\mathbf{H}^T = \mathbf{0}$, или, что эквивалентно, тогда и только тогда, когда

$$\sum_{1 \le i \le n} a_i \mathbf{h}_i = \mathbf{0},$$

где $\mathbf{h}_i$ есть i-й столбец матрицы $\mathbf{H}$. Таким образом, если w — минимальный вес кода, то не существует равной 0 линейной комбинации $w-1$ или меньшего числа столбцов матрицы $\mathbf{H}$. □

Из теоремы 4.1.12 мы видим, что код может иметь кодовое расстояние 3 тогда и только тогда, когда любые $3-1=2$ столбца проверочной матрицы $\mathbf{H}$ линейно независимы. Однако два ненулевых вектора в $\mathbf{V}_{n-k}$ над $GF(2)$ линейно независимы, если они просто различны. Поэтому мы можем построить код с кодовым расстоянием 3 (т.е. код может исправлять *одну* ошибку), если будем рассматривать в качестве проверочной матрицы $\mathbf{H}$ матрицу, столбцы которой составляют все отличные от нуля элементы пространства $\mathbf{V}_{n-k}$. Значит, мы можем построить так называемые $(2^m-1, 2^m-m-1, 3)$-коды Хэмминга, т.е. $n = 2^m - 1$, $k = 2^m - m - 1$ и $d_{\min} = 3$.

Определение 4.1.13. Пусть $\mathbf{H}$ — проверочная матрица линейного (n,k)-кода. Если $\mathbf{v}$ — полученный вектор, то вектор $\mathbf{s} = \mathbf{v}\mathbf{H}^T$ длины $n-k$ называется *синдромом* вектора $\mathbf{v}$.

Из этого определения видно, что вектор является кодовым словом тогда и только тогда, когда его синдром равен $\mathbf{0}$. Если учитывать это, то процесс декодирования для кодов Хэмминга очень прост. Если $\mathbf{v}$ — полученный, а $\mathbf{u}$ — переданный вектор, где $\mathbf{e} = \mathbf{v} - \mathbf{u}$, мы действуем следующим образом: сначала вычисляем синдром вектора $\mathbf{v}$, который равен

$$\mathbf{s} = \mathbf{v}\mathbf{H}^T = (\mathbf{u} + \mathbf{e})\mathbf{H}^T = \mathbf{e}\mathbf{H}^T.$$

Если произошла *одна* ошибка, то $\mathbf{e}$ — вектор с весом 1, т.е. $n-1$ компонент равны 0 и мы имеем «1» в i-й позиции, где встретилась ошибка. Таким образом, произведение $\mathbf{e}\mathbf{H}^T$ будет i-м столбцом матрицы $\mathbf{H}$, и мы получим переданный вектор, если прибавим к $\mathbf{v}$ вектор $\mathbf{e}$, который состоит из нулей, за исключением i-й позиции, которая равна 1.

Пример. Рассмотрим (7,4)-код Хэмминга с проверочной матрицей

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

и предположим, что получен вектор $\mathbf{v} = (0,1,1,0,1,0,0)$. Мы видим, что синдром вектора $\mathbf{v}$ равен

$$\mathbf{s} = \mathbf{v}\mathbf{H}^T = (1,1,0),$$

что совпадает с четвертым столбцом матрицы $\mathbf{H}$; следовательно, вектор ошибок равен $\mathbf{e} = (0,0,0,1,0,0,0)$, и, добавляя его к $\mathbf{v}$, мы получаем переданный вектор $\mathbf{u} = (0,1,1,1,1,0,0)$.

Рассмотрим теперь другую процедуру декодирования линейных кодов, т.е. мы получим другой взгляд на декодирование по максимуму правдоподобия, рассматривая *смежные классы* кода C в $\mathbf{V}_n$.

Пусть C — линейный (n,k)-код над $GF(2)$, где этот код рассматривается как подпространство пространства $\mathbf{V}_n$, векторного пространства всех двоичных n-кортежей. Факторпространство $\mathbf{V}_n/C$ состоит из всех смежных классов $\mathbf{v} + C = \{\mathbf{v} + \mathbf{x} : \mathbf{x} \in C\}$ для произвольного $\mathbf{v} \in \mathbf{V}_n$, где каждый смежный класс содержит 2^k векторов. Имеется разбиение множества $\mathbf{V}_n$ вида

$$\mathbf{V}_n = C \cup \{\mathbf{v}_1 + C\} \cup \cdots \cup \{\mathbf{v}_s + C\}, \quad s = 2^{n-k} - 1.$$

Если получен вектор $\mathbf{y}$, то $\mathbf{y}$ должен быть элементом одного из этих смежных классов, скажем $\{\mathbf{v}_i + C\}$. Если передано кодовое слово $\mathbf{x}_i$, то вектор ошибок $\mathbf{e}$ равен $\mathbf{e} = \mathbf{y} - \mathbf{x}_i \in \mathbf{v}_i + C - \mathbf{x}_i = \mathbf{v}_i + C$. Следовательно, мы имеем следующее правило декодирования: если получен вектор $\mathbf{y}$, то возможные векторы ошибки $\mathbf{e}$ — это векторы из смежного класса, содержащего $\mathbf{y}$. Наиболее вероятная ошибка — вектор $\mathbf{e}'$ с минимальным весом в классе смежности вектора $\mathbf{y}$. Поэтому $\mathbf{y}$ декодируется как $\mathbf{x}' = \mathbf{y} + \mathbf{e}'$. (Вектор наименьшего веса в смежном классе называется *лидером смежного класса*. Если имеется несколько таких векторов, то произвольный из них выбирается в качестве лидера смежного класса.)

Чтобы использовать указанную схему декодирования, нам нужны таблицы смежных классов, и ниже мы покажем, как создавать таблицу декодирования Слепиана.

Пусть C — упомянутый выше (n,k)-линейный код, и пусть $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_r$, $r = 2^k$, — кодовые слова, где $\mathbf{h}_1$ — нулевой вектор кода. Кодовые слова располагают в первой строке с нулевым вектором слева. Затем выбирают один из оставшихся векторов пространства $\mathbf{V}_n$, например $\mathbf{g}_1$, и помещают его под нулевым вектором. (Обычно $\mathbf{g}_1$ выбирается так, что он имеет наибольшую вероятность поступить на приемник, если передан нулевой вектор.) После этого заполняют вторую строку, помещая под каждым кодовым словом $\mathbf{h}_i$ вектор $\mathbf{g}_1 + \mathbf{h}_i$. Аналогично, второй вектор $\mathbf{g}_2$ из $\mathbf{V}_n$, который еще не появился в таблице, помещают в первом столбце третьей строки, и строка заполняется тем же способом. Этот процесс продолжается до тех пор, пока все векторы из $\mathbf{V}_n$ не появятся в таблице. Множество элементов строки образует смежный класс, и, если на каждом шаге вектор $\mathbf{g}_i$ выбирался как вектор наименьшего веса, не представленный в первых i столбцах, то элементы *первого столбца* гарантированно будут лидерами смежных классов (проверьте это на следующем ниже примере).

Два элемента $\mathbf{g}_i$ и $\mathbf{g}_j$, которые принадлежат одному и тому же смежному классу k, обладают следующим свойством: $\mathbf{g}_i + \mathbf{g}_j = \mathbf{g}_k + \mathbf{h}_i + \mathbf{g}_k + \mathbf{h}_j = \mathbf{h}_i + \mathbf{h}_j = \mathbf{h}_m$, т.е. их сумма равна кодовому слову. Кроме того, каждый вектор из пространства $\mathbf{V}_n$ появляется в таблице ровно один раз. (В этом легко убедиться, если предположить, что два вектора $\mathbf{g} = \mathbf{g}_k + \mathbf{h}_i$ и $\mathbf{g}' = \mathbf{g}_m + \mathbf{h}_j$ совпадают, где $\mathbf{g}_k$ и $\mathbf{g}_m$ —лидеры смежных классов. Равенство $\mathbf{g}_k + \mathbf{h}_i = \mathbf{g}_m + \mathbf{h}_j$ имеет место тогда и только тогда, когда $\mathbf{g}_k = \mathbf{g}_m + \mathbf{h}_i + \mathbf{h}_j = \mathbf{g}_m + \mathbf{h}_m$, где $\mathbf{h}_m$ — кодовое слово; но тогда $\mathbf{g}_k$ принадлежит смежному классу, который имеет $\mathbf{g}_m$ в качестве лидера, а это противоречит методу построения таблицы, согласно которому первый элемент в каждой строке — это не появлявшийся ранее вектор.)

Пример. Рассмотрим $(5,3)$-линейный код, проверочные биты которого определяются уравнениями

$$\begin{aligned} c_1 &= m_1 + m_2, \\ c_2 &= m_1 \qquad\quad + m_3. \end{aligned}$$

Тогда таблица Слепиана этого кода имеет вид

$$\begin{array}{ccccccccc} 00000 & 10011 & 01010 & 11001 & 00101 & 10110 & 01111 & 11100 \\ 00001 & 10010 & 01011 & 11000 & 00100 & 10111 & 01110 & 11101 \\ 00010 & 10001 & 01000 & 11011 & 00111 & 10100 & 01101 & 11110 \\ 10000 & 00011 & 11010 & 01001 & 10101 & 00110 & 11111 & 01100. \end{array}$$

Отметим, что имеется $2^{5-3} = 4$ смежных класса, каждый из которых содержит $2^3 = 8$ элементов. Следующие две теоремы показывают, как описанная выше таблица Слепиана может использоваться для декодирования линейных кодов.

Теорема 4.1.14. Если мы пользуемся таблицей Слепиана для декодирования полученного вектора в кодовое слово, расположенное над ним, то полученный вектор $\mathbf{v}$ правильно декодируется в переданный вектор $\mathbf{u}$ тогда и только тогда, когда вектор ошибки $\mathbf{e}_i = \mathbf{v} - \mathbf{u}$ — лидер смежного класса.

Доказательство. Если $\mathbf{e}_i = \mathbf{v} - \mathbf{u}$ — лидер i-го смежного класса, то $\mathbf{v} = \mathbf{u} + \mathbf{e}_i$ появляется в i-м смежном классе таблицы Слепиана под кодовым словом $\mathbf{u}$ и будет правильно декодирован. Однако если $\mathbf{v} - \mathbf{u}$ не является лидером смежного класса, то полученный вектор $\mathbf{v}$ появится в некотором смежном классе, скажем j-м, с лидером $\mathbf{g}_j$. Тогда $\mathbf{v}$ принадлежит j-й строке таблицы, но не находится под $\mathbf{u}$, потому что $\mathbf{v} \neq \mathbf{g}_j + \mathbf{u}$. □

Теорема 4.1.15. Два вектора $\mathbf{v}_1$ и $\mathbf{v}_2$ принадлежат одному и тому же смежному классу тогда и только тогда, когда их синдромы равны.

Доказательство. Если и $\mathbf{v}_1$, и $\mathbf{v}_2$ принадлежат i-му смежному классу, то $\mathbf{v}_1 = \mathbf{g}_i + \mathbf{h}_1$ и $\mathbf{v}_2 = \mathbf{g}_i + \mathbf{h}_2$. Значит, $\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{h}_1 + \mathbf{h}_2$ и $(\mathbf{v}_1 + \mathbf{v}_2)\mathbf{H}^T = (\mathbf{h}_1 + \mathbf{h}_2)\mathbf{H}^T = \mathbf{0}$, откуда следует, что $\mathbf{v}_1\mathbf{H}^T = \mathbf{v}_2\mathbf{H}^T$.

Обратно, если $\mathbf{v}_1\mathbf{H}^T = \mathbf{v}_2\mathbf{H}^T$, то $(\mathbf{v}_1 + \mathbf{v}_2)\mathbf{H}^T = \mathbf{0}$, откуда следует, что $\mathbf{v}_1 + \mathbf{v}_2 = \mathbf{h}_m$, и если $\mathbf{v}_1 = \mathbf{h}_1 + \mathbf{g}_i$, то $\mathbf{v}_2 = \mathbf{h}_m + \mathbf{v}_1 = \mathbf{h}_m + \mathbf{h}_1 + \mathbf{g}_i = \mathbf{g}_i + \mathbf{h}_2$, откуда следует, что $\mathbf{v}_1$ и $\mathbf{v}_2$ принадлежат одному и тому же смежному классу. □

Теперь мы в состоянии представить упрощенную процедуру декодирования, которая работает следующим образом. Сформируем таблицу декодирования, которая состоит из двух столбцов, первый составлен из 2^{n-k} синдромов, а второй — из соответствующих лидеров смежных классов. Когда получен вектор $\mathbf{v}$, то мы сначала вычисляем его синдром, а затем находим из таблицы соответствующий лидер смежного класса. Мы полагаем, что лидер смежного класса — это вектор ошибки, и, прибавляя его к вектору $\mathbf{v}$, получаем переданный вектор $\mathbf{u}$.

Пример. Для линейного (5,3)-кода, описанного в предыдущем при-

мере, мы имеем следующую таблицу декодирования:

Лидер смежного класса	*Синдром*
00000	00
00001	01
00010	10
10000	11

Мы видим, что декодирующей таблице для (n,k)-кода требуется список синдромов и лидеров смежных классов, соответствующих 2^{n-k} смежным классам. Это может оказаться как дороже, так и дешевле, чем сравнивать полученный вектор со всеми возможными 2^k кодовыми словами и выбирать ближайший. В (100,80)-коде, например, имеется только 2^{20} смежных классов в противовес 2^{80} кодовым словам, хотя 2^{20} — очень большое число.

Теорема 4.1.16. Пусть C есть (n,k)-линейный код, кодовые слова которого имеют одинаковую вероятность передачи. Тогда использование построенной выше таблицы Слепиана (т.е. лидеров смежных классов, имеющих наименьший вес в смежном классе) максимизирует вероятность правильного декодирования.

Доказательство. Пусть $\mathbf{v}_{ij}$ — вектор в i-й строке и j-м столбце таблицы Слепиана. Кодовые слова $\mathbf{v}_{0j}$ находятся в 0-й строке и наверху каждого столбца. Пусть d_{ij} — расстояние Хэмминга между полученным вектором $\mathbf{v}_{ij}$ и кодовым словом $\mathbf{v}_{0j}$, в которое декодируется $\mathbf{v}_{ij}$. Мы уже видели, что вероятность правильного декодирования максимизируется, когда полученный вектор декодируется в ближайшее кодовое слово.

Предположим теперь, что вектор $\mathbf{v}$ появляется в таблице Слепиана под кодовым словом $\mathbf{u}$, где $d(\mathbf{v},\mathbf{u}) = w$, и пусть $\mathbf{g}$ — лидер смежного класса, содержащего $\mathbf{v}$; более того, предположим, что $\mathbf{u}_1$ — ближайшее к $\mathbf{v}$ кодовое слово, причем $d(\mathbf{v},\mathbf{u}_1) = w_1$. Тогда вес вектора $\mathbf{g} = \mathbf{v} - \mathbf{u}$ равен w, в то время как элемент $\mathbf{v} - \mathbf{u}_1 = \mathbf{g} + (\mathbf{u} - \mathbf{u}_1)$ имеет вес w_1 и находится в том же самом смежном классе. Поскольку по определению $\mathbf{g}$ имеет наименьший вес в смежном классе, $w_1 \geq w$, и, таким образом, $\mathbf{v}$ по крайней мере так же близок к $\mathbf{u}$, как и к $\mathbf{u}_1$. $\square$

4.1.2. БЧХ-коды

В этом разделе мы рассматриваем БЧХ-коды, позволяющие обнаруживать и исправлять две ошибки. Эти коды — главное улучшение

кодов Хэмминга, которые могут обнаруживать две ошибки, но исправлять только одну. БЧХ-коды были разработаны Боузом и Чоудхури в 1960 г. и Хоквингемом в 1959 г., и они образуют подмножество *циклических* кодов; последние рассматриваются ниже подробно. Начнем с двух определений.

Определение 4.1.17. Подпространство C n-мерного двоичного векторного пространства $\mathbf{V}_n$ называется *циклическим подпространством* или *циклическим кодом*, если для каждого вектора $\mathbf{v} = (v_{n-1}, v_{n-2}, \ldots, v_0)$ кода C вектор $\mathbf{v}' = (v_0, v_{n-1}, v_{n-2}, \ldots, v_1)$, полученный из $\mathbf{v}$ «циклическим сдвигом» на одну позицию вправо, также принадлежит C.

Определение 4.1.18. Множество полиномов называется *идеалом* тогда и только тогда, когда оно состоит из всех кратных некоторого полинома.

Строго говоря, мы определили только что *главный идеал*. Однако легко показать, что если J — поле, то любой идеал в кольце полиномов $J[x]$ — главный идеал, порожденный полиномом минимальной степени в $J[x]$. (Остаток от деления любого другого полинома в идеале на *образующий* должен быть нулем.) Поэтому в наших рассуждениях мы используем просто термин «идеал». Кроме того, для описания циклических кодов мы будет использовать алгебру полиномов по модулю $x^n - 1$, а также тот факт, что полином степени $n-1$ может быть представлен вектором в $\mathbf{V}_n$; последнее всегда предполагается двоичным векторным пространством, даже когда это явно не упоминается.

Чтобы понять, зачем нам понадобится алгебра полиномов по модулю $x^n - 1$, рассмотрим следующее альтернативное описание уже встречавшегося нам (7,4)-кода Хэмминга.

Столбцы проверочной матрицы $\mathbf{H}$ (7,4)-кода Хэмминга суть двоичные векторы длины 3. Мы знаем, что их можно рассматривать как полиномы степени ≤ 2 над $GF(2)$. Это наводит на мысль, что столбцы матрицы $\mathbf{H}$ могут индексироваться элементами из $GF(2^3)$. Более точно, если α — корень полинома $x^3 + x + 1$, то семь ненулевых элементов поля $GF(2^3)$ — это степени α^i, и они соответствуют столбцам матрицы $\mathbf{H}$. (Имеется соответствие, аналогичное табл. 3.3.1, и читатель может построить подобную таблицу, чтобы в этом убедиться; см. также теорему 3.3.11.) Аналогичным образом кодовые слова в (7,4)-коде Хэмминга могут быть записаны в виде двоичных полиномов степени ≤ 6; в этом случае кодовые слова — это все полиномы $p(x)$ степени ≤ 6, для которых α является корнем.

Итак, $m(x) = x^3 + x + 1$ — минимальный полином элемента α над $GF(2)$, и все полиномы, для которых α является корнем, — это в точности элементы идеала, порожденного полиномом $m(x)$ в кольце $\mathbb{Z}_2[x]$. Поскольку α – элемент порядка 7 (т.е. α — корень полинома $x^7 - 1$), мы имеем $m(x)|(x^7 - 1)$. Рассмотрим факторкольцо $\mathbb{Z}_2[x]_{(x^7-1)}$ и образ порожденного $m(x)$ идеала в этом факторкольце. Ясно, что если дано какое-то кратное $k(x)m(x)$ полинома $m(x)$, то после деления его на $x^7 - 1$ мы получим $k(x)m(x) = q(x)(x^7 - 1) + r(x)$, $\deg[r(x)] \leq 6$. Далее, $r(x) = k(x)m(x) - q(x)(x^7 - 1)$, и поскольку $m(x)|(x^7 - 1)$, то $m(x)|r(x)$ и $r(x)$ — кратное полинома $m(x)$. Поэтому образ идеала, порожденного полиномом $m(x)$, при естественном отображении из $\mathbb{Z}_2[x]$ в $\mathbb{Z}_2[x]_{(x^7-1)}$ можно рассматривать как совокупность в точности тех полиномов $p(x)$ степени ≤ 6, которые делятся на $m(x)$. Таким образом, мы получаем описание (7,4)-кода Хэмминга в виде идеала, порожденного образом полинома $m(x)$ в $\mathbb{Z}_2[x]_{(x^7-1)}$. Далее мы обобщим приведенные рассуждения, но сначала представим альтернативную схему кодирования/декодирования для (7,4)-кода Хэмминга.

Кодирование. Предположим, что мы хотим передать информационные биты m_1, m_2, m_3 и m_4. Мы образуем полином $m_1x^6 + m_2x^5 + m_3x^4 + m_4x^3$ и делим его на $m(x) = x^3 + x + 1$, чтобы получить частное $q(x)$ и остаток $r(x) = r_1x^2 + r_2x + r_3$, где $\deg[r(x)] < 3$. Тогда передается кодовое слово $m_1, m_2, m_3, m_4, r_1, r_2, r_3$, соответствующее полиному $m(x)q(x) = m_1x^6 + m_2x^5 + m_3x^4 + m_4x^3 + r_1x^2 + r_2x + r_3$, который в точке α обращается в 0.

Декодирование. Предположим, что произошла по крайней мере одна ошибка. Тогда, получив 7-битовый вектор, пользователь вычисляет значение соответствующего полинома в точке α, используя, конечно, свойство $\alpha^2 = \alpha + 1$. Если ответ нулевой, то ошибок нет, а если ответ равен α^e, то ошибка была в коэффициенте при e-й степени переменной x.

Пример. Чтобы передать информацию $(1, 0, 0, 1)$, мы разделим полином $x^6 + x^3$ на $x^3 + x + 1$, чтобы получить остаток $x^2 + x$; передается кодовое слово $(1, 0, 0, 1, 1, 1, 0)$. Заметим, что значение полинома $x^6 + x^3 + x^2 + x$ в точке α равно нулю. (Отметим, что $\alpha^3 = \alpha + 1$ и $\alpha^6 = \alpha^2 + 1$.) Если теперь получен вектор $(0, 0, 0, 1, 1, 1, 0)$, то мы вычисляем значение соответствующего полинома в точке α и получаем $\alpha^2 + 1 = \alpha^6$; из этого мы заключаем, что ошибка произошла в коэффициенте при x^6.

Оправдывая необходимость использования алгебры полиномов по

модулю x^n-1 для описания циклических кодов, мы представляем две теоремы, которые определяют некоторые основные свойства идеалов.

Теорема 4.1.19. Пусть J — идеал алгебры полиномов по модулю $f(x)$, и пусть $g(x)$ — один из ненулевых полиномов наименьшей степени в J. Тогда полином $s(x)$ принадлежит J в том и только том случае, когда $g(x)|s(x)$. Более того, $g(x)|f(x)$.

Доказательство. Применяя алгоритм деления полиномов, получаем $s(x) = g(x)q(x) + r(x)$, где $\deg[r(x)] < \deg[g(x)]$. Отсюда следует, что $r(x) = s(x) + g(x)q(x)$, и поскольку и $s(x)$, и $g(x)$ принадлежат J, $r(x)$ также принадлежит J. Однако $g(x)$ — полином наименьшей степени в J, так что $r(x)$ должен быть нулевым полиномом, а $s(x)$ — кратное полинома $g(x)$. Рассматривая $f(x)$, имеем $f(x) = g(x)q(x) + r(x)$, и поскольку $f(x) = 0 \pmod{f(x)}$, то $g(x)q(x) + r(x) = 0 \pmod{f(x)}$, откуда $g(x)q(x) = r(x) \pmod{f(x)}$. Подобными рассуждениями получаем, что $r(x) = 0$ и $g(x)|f(x)$. □

Таким образом, каждый идеал алгебры полиномов по модулю $f(x)$ порожден полиномом $g(x)$, степень которого меньше степени любого другого полинома в J, и элементы идеала J — это кратные полинома $g(x)$. Если $n = \deg[f(x)]$ и $r = \deg[g(x)]$, то идеал J имеет размерность $n-r$. Действительно, J — векторное подпространство и векторы $g(x), xg(x), \ldots, x^{n-r-1}g(x)$ линейно независимы, потому что любая их линейная комбинация $(a_0 + \cdots + a_{n-r-1}x^{n-r-1})g(x)$ не может быть равной $0 \pmod{f(x)}$, поскольку его степень $i < n$. Очевидно, каждый полином $s(x)$ в J может быть выражен через эти базисные векторы. Полином $g(x)$ называется *образующим* идеала J или его *порождающим* полиномом.

Мы говорим, что полином $r(x)$ *ортогонален* идеалу J, если $r(x)s(x) = 0 \pmod{f(x)}$ для любого полинома $s(x)$ в J.

Теорема 4.1.20. Рассмотрим полиномы $f(x), g(x)$ и $h(x)$, где $f(x) = g(x)h(x)$. Тогда в алгебре полиномов по модулю $f(x)$ полином $a(x)$ ортогонален идеалу, порожденному полиномом $h(x)$, тогда и только тогда, когда $a(x)$ принадлежит идеалу, порожденному полиномом $g(x)$.

Доказательство. Предположим, что $a(x)$ принадлежит идеалу, порожденному полиномом $g(x)$, и что $b(x)$ принадлежит идеалу, порожденному полиномом $h(x)$; тогда по теореме 4.1.19 $a(x)b(x) = q(x)g(x)h(x) = q(x)f(x) = 0 \pmod{f(x)}$.

Обратно, если $a(x)$ ортогонален идеалу, порожденному полиномом $h(x)$, то $a(x)h(x) = 0 \pmod{f(x)}$, т.е. $a(x)$ — кратное полинома $f(x) = g(x)h(x)$. Отсюда мы заключаем, что $a(x)$ должен быть кратным полинома $g(x)$ и поэтому принадлежит идеалу, порожденному полиномом $g(x)$. □

Пример. Рассмотрим алгебру полиномов по модулю x^7-1; поскольку коэффициенты принадлежат $GF(2)$, мы имеем $-1 = +1$; следовательно, $x^7 - 1 = x^7 + 1$. Будем теперь работать с $x^7 + 1$. Полином $g(x) = x^3 + x^2 + 1$ делит $x^7 + 1$, и, значит, в силу теоремы 4.1.19 мы можем рассматривать его как образующий идеала с базисными векторами $g(x), xg(x), x^2g(x)$ и $x^3g(x)$. Полином $a(x) = (x^2+1)g(x) = x^5+x^4+x^3+1$ принадлежит этому идеалу. С учетом соответствия между векторами и полиномами этот идеал есть не что иное, как подпространство векторного пространства $\mathbf{V}_7$ с базисными векторами $(0,0,0,1,1,0,1)$, $(0,0,1,1,0,1,0)$, $(0,1,1,0,1,0,0)$, $(1,1,0,1,0,0,0)$, а полином $a(x)$ — это вектор $(0,1,1,1,0,0,1) = (0,1,1,0,1,0,0) + (0,0,0,1,1,0,1)$. Идеал, ортогональный к упомянутому выше, — это идеал, порожденный полиномом $h(x) = (x^7+1)/(x^3+x^2+1) = x^4+x^3+x^2+1$, а базисными векторами являются $h(x)$, $xh(x)$ и $x^2h(x)$.

Заметим, что в алгебре полиномов по модулю $(x^n - 1)$ размерность идеала равна $n-r$, где r — степень его образующего полинома. Размерность ортогонального ему идеала равна r, и степень его порождающего полинома равна $n - r$. В приведенном выше примере $\deg[g(x)] = 3$, $\deg[h(x)] = 7 - 3 = 4$ и размерности соответствующих идеалов равны 4 и 3.

Теорема 4.1.21. В алгебре полиномов по модулю $(x^n - 1)$ подпространство пространства $\mathbf{V}_n$ является циклическим тогда и только тогда, когда оно — идеал.

Доказательство. Основное замечание в этом случае состоит в том, что умножение по модулю $(x^n - 1)$ полинома на x эквивалентно циклическому сдвигу, потому что

$$\begin{aligned} &x(a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x + a_0) \\ &\quad = a_{n-1}(x^n - 1) + a_{n-2}x^{n-1} + \cdots + a_1x^2 + a_0x + a_{n-1} \\ &\quad = a_{n-2}x^{n-1} + \cdots + a_1x^2 + a_0x + a_{n-1} \pmod{(x^n - 1)}. \end{aligned}$$

Если C — циклическое подпространство пространства $\mathbf{V}_n$, то из $\mathbf{v} \in C$ следует, что $x \cdot \mathbf{v}$, $x^j \cdot \mathbf{v}$ и $(c_{n-1}x^{n-1}+c_{n-2}x^{n-2}+\cdots+c_1x+c_0)\cdot\mathbf{V}$ все принадлежат C, и по определению C — идеал.

Обратно, если подпространство C пространства $\mathbf{V}_n$ является идеалом, то из $\mathbf{v} \in C$ следует, что $\mathbf{v}' = x \cdot \mathbf{v} \in C$, а поскольку коэффициенты полинома $\mathbf{v}'$ получены циклическим сдвигом коэффициентов полинома $\mathbf{v}$, мы заключаем, что C — циклическое подпространство. $\square$

Таким образом, циклический код полностью определяется порождающим полиномом, который делит $x^n - 1$. Сообщение $a_{k-1}, a_{k-2}, \ldots, a_0$ (максимальной длины k) может быть закодировано с использованием кода C путем вычисления произведения $a(x)g(x)$, где $a(x)$ — полином, соответствующий $a_{k-1}, a_{k-2}, \ldots, a_0$, и $g(x)$, $\deg[g(x)] = r$, является образующим идеала C, размерность которого равна $n - r$.

Пример. Предположим, что нужно передать сообщение, состоящее из трех битов, и что порождающий полином циклического кода равен $g(x) = x^3 + x + 1 = (1, 0, 1, 1)$. Здесь $x^3 + x + 1$ делит $x^7 - 1$ или $x^7 + 1$. Тогда возможные кодовые слова вычисляются как произведение $a(x)$ на $g(x)$ для каждого полинома-сообщения $a(x)$. То есть мы имеем

$$\begin{aligned} 000 &\to 0000000, & 100 &\to 0101001, \\ 001 &\to 0001011, & 101 &\to 0100010, \\ 010 &\to 0010110, & 110 &\to 0111010, \\ 011 &\to 0011101, & 111 &\to 0110001. \end{aligned}$$

Очевидно, что для этого кода сообщение может иметь не более четырех битов информации.

Эквивалентно, код может быть определен как подпространство, ортогональное идеалу, порожденному полиномом $h(x) = (x^n - 1)/g(x)$. Заметим, что $g(x)h(x) \equiv 0 \pmod{(x^n - 1)}$. Если $\deg[g(x)] = r$, то размерность кода равна $n - r$. Элемент $a(x)$ принадлежит коду тогда и только тогда, когда $g(x) | a(x)$, или, эквивалентно, тогда и только тогда, когда $a(x)h(x) \equiv 0 \pmod{(x^n - 1)}$. Чтобы доказать это в одном направлении, рассмотрим полином $a(x)$ в коде; тогда имеется полином $u(x)$, такой, что $a(x) = u(x)g(x)$. Однако поскольку $g(x)h(x) \equiv 0 \pmod{(x^n - 1)}$, то $a(x)h(x) \equiv u(x)g(x)h(x) \equiv 0 \mod (x^n - 1)$. В качестве упражнения читателю остается доказательство в обратном направлении. Из $a(x)h(x) \equiv 0 \pmod{(x^n - 1)}$ мы заключаем, что коэффициент при x^i в произведении $a(x)h(x) \equiv 0 \pmod{(x^n - 1)}$ задается формулой $\sum_{0 \le i \le n-1} a_i h_{j-i} = 0$, $j = 0, 1, \ldots, n-1$, где индексы вычисляются по модулю n. Это объясняет тот факт, что матрица $\mathbf{H}$, соответствующая полиному $h(x)$, имеет обратный порядок элементов. Детали даны в приводимом ниже примере. Например, в

приведенном выше примере кодовое слово 0110001 соответствует полиному x^5+x^4+1 и $h(x)=(x^7+1)/(x^3+x+1)=x^4+x^2+x+1$. Ясно, что $a(x)h(x)\equiv 0\ [\mathrm{mod}\ (x^7-1)]$. Полином $h(x)$ называется *проверочным полиномом* кода, порожденного полиномом $g(x)$. Поскольку $h(x)|(x^n-1)$, он может быть также использован как порождающий полином другого кода C', *дуального* коду C. Аналогия с введенным выше дуальным кодом, а также соответствие между полиномами $g(x)$ и $h(x)$ и матрицами $\mathbf{G}$ и $\mathbf{H}$ становится понятнее, если разобрать следующий пример.

Пример. Рассмотрим полином $x^7-1=x^7+1=(x+1)(x^3+x+1)(x^3+x^2+1)$. Полином $g(x)=x^3+x^2+1$ порождает циклический код C, у которого $n=7$, $k=4$. Элементы $x^3g(x), x^2g(x), xg(x)$ и $g(x)$ образуют базис кода, и, следовательно, матрицу $\mathbf{G}$ можно рассматривать как порождающую матрицу кода.

$$\begin{aligned} x^3g(x)&=(1,1,0,1,0,0,0),\\ x^2g(x)&=(0,1,1,0,1,0,0),\\ xg(x)&=(0,0,1,1,0,1,0),\\ g(x)&=(0,0,0,1,1,0,1), \end{aligned} \qquad \mathbf{G}=\begin{bmatrix} 1&1&0&1&0&0&0\\ 0&1&1&0&1&0&0\\ 0&0&1&1&0&1&0\\ 0&0&0&1&1&0&1 \end{bmatrix}.$$

Кроме того, этот код является подпространством, ортогональным идеалу, порожденному полиномом $h(x)=(x^7-1)/g(x)=(x-1)(x^3+x+1)=x^4+x^3+x^2+1$; базисные векторы этого идеала суть

$$\begin{aligned} x^2h(x)&=(1,1,1,0,1,0,0),\\ xh(x)&=(0,1,1,1,0,1,0),\\ h(x)&=(0,0,1,1,1,0,1). \end{aligned}$$

Поскольку умножение полиномов отличается от скалярного произведения векторов, код C определяет подпространство, ортогональное матрице $\mathbf{H}$, образованной из векторов $x^2h(x)$, $xh(x)$, $h(x)$ с *обратным* порядком координат. Поэтому

$$\mathbf{H}=\begin{bmatrix} 0&0&1&0&1&1&1\\ 0&1&0&1&1&1&0\\ 1&0&1&1&1&0&0 \end{bmatrix}.$$

Легко убедиться, что $\mathbf{GH}^T$ — нулевая матрица. Этот код эквивалентен (7,4)-коду Хэмминга.

Другой способ определить циклический код — задать корни порождающего полинома $g(x)$ в $GF(2^m)$, т.е. полином $g(x)$ определяется неявно. При таком подходе вектор $a(x)$ принадлежит коду, если все корни полинома $g(x)$ являются корнями этого вектора. Более того, поскольку $g(x)$ должен делить $x^n - 1$, все корни $\alpha_1, \alpha_2, \ldots, \alpha_r$ полинома $g(x)$ должны быть корнями и полинома $x^n - 1$, и в соответствии со сказанным в разд. 3.3 порядок каждого корня должен делить n. Поэтому если нам известны корни порождающего полинома $g(x)$ в $GF(2^m)$, то мы можем определить длину кода n как наименьшее общее кратное порядков корней. Следующие примеры разъясняют это понятие.

Пример. Код из предыдущего примера может быть определен следующим образом. Каждый полином, принадлежащий коду, должен иметь корень α, где α — один из примитивных элементов поля $GF(2^3)$. Примитивные элементы поля $GF(2^3)$ — это α и α^3 (проверьте!), откуда следует, что порождающий полином кода имеет вид

$$g(x) = (x+\alpha)(x+\alpha^2)(x+\alpha^4) \quad \text{или} \quad g(x) = (x+\alpha^3)(x+\alpha^6)(x+\alpha^5),$$

т.е. порождающий полином — это один из неприводимых полиномов $x^3 + x^2 + 1$ или $x^3 + x + 1$. Порядок корня α равен $2^3 - 1 = 7$, и, следовательно, длина кода равна $n = 7$. Поскольку $\deg[g(x)] = 3$, то имеется $7 - 3 = 4$ информационных бита.

Пример. Пусть α — примитивный элемент поля $GF(2^4)$. Рассмотрим код, в котором каждый полином имеет корни $\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5$ и α^6; для α мы имеем соотношение $\alpha^{15} - 1 = 0$.

Однако полином, имеющий корень α, будет также иметь корни α^2, α^4 и α^8. Аналогично, полином с корнем α^3 будет также иметь корни $\alpha^6, \alpha^{12}, \alpha^{24} = \alpha^{15}\alpha^9 = \alpha^9$, и, наконец, полином с корнем α^5 будет также иметь корень α^{10}. Поэтому

$$\begin{aligned} g(x) =& [(x+\alpha)(x+\alpha^2)(x+\alpha^4)(x+\alpha^8)] \\ & \times [(x+\alpha^3)(x+\alpha^6)(x+\alpha^{12})(x+\alpha^9)][(x+\alpha^5)(x+\alpha^{10})] \\ =& (x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1). \end{aligned}$$

Полином $g(x)$ имеет степень 10, длина кода равна $2^4 - 1 = 15$, и он содержит $k = 15 - 10 = 5$ информационных битов.

Теперь мы подытожим простое правило кодирования и декодирования, соответствующее этому определению циклического кода.

Кодирование. Пусть $g(x)$ — порождающий полином кода, $\deg[g(x)] = r$, и пусть $a_{k-1}, a_{k-2}, \ldots, a_0$ — сообщение с $k = n - r$ информационными битами. Тогда это сообщение рассматривается как полином $a(x)$ над $GF(2)$ и кодируется как $a(x)g(x)$.

Декодирование. Полученный вектор $v(x)$ делим на $g(x)$ [остаток от этого деления — это синдром вектора $v(x)$]. Если в результате деления получен ненулевой остаток, то должна иметь место ошибка передачи. Чтобы распознать ошибку $e(x)$, мы вычисляем произведение $v(x)h(x) \pmod{(x^n - 1)}$, где $h(x)$ — проверочный полином кода. В соответствии со сделанными ранее замечаниями $v(x)h(x) = [u(x) + e(x)]h(x) = 0 + e(x)h(x) \pmod{(x^n - 1)}$. Тогда, разделив произведение на $h(x)$, получим полином ошибки $e(x)$, и $u(x) = v(x) - e(x)$. Разделив $u(x)$ на $g(x)$, мы получаем переданное сообщение.

Пример. Пусть $g(x) = x^3 + x^2 + 1$ — порождающий полином кода (7,4), который мы рассматривали в предыдущем примере. Сообщение (1,1,1,0) кодируется как (1,0,0,0,1,1,0).

Предположим теперь, что получен вектор $v(x) = (1,0,0,0,0,1,1)$ (с двумя ошибками); тогда, разделив $x^6 + x + 1$ на $g(x)$, мы получим в остатке $x^2 + 1$; значит, при передаче произошла ошибка. Проверочным полиномом является $h(x) = x^4 + x^3 + x^2 + 1$, и $v(x)h(x) \pmod{(x^7 - 1)}$ равен $x^6 + x^5 + x^3 + 1 = (1,1,0,1,0,0,1)$. Итак, $e(x)h(x) = x^6 + x^5 + x^3 + 1$, и, разделив на $h(x)$, мы получим $e(x) = x^2 + 1 = (0,0,0,0,1,0,1)$. Таким образом, $u(x) = v(x) - e(x) = (1,0,0,0,1,1,0) = x^6 + x^2 + x$, и, разделив полученное на $g(x)$, мы получим переданный вектор $(1,1,1,0)$.

Если имеется только одна ошибка передачи, то $e(x) = x^i$, и мы исправляем i-й бит.

В общем случае если $g(x) = g_r x^r + g_{r-1} x^{r-1} + \cdots + g_0$ — порождающий полином кода, то полиномы $x^{n-r-1}g(x)$, $x^{n-r-2}g(x)$, $\ldots$, $g(x)$ являются кодовыми словами. Таким образом, строки следующей матрицы — это все кодовые слова:

$$\mathbf{G} = \begin{bmatrix} g_r & g_{r-1} & \ldots & g_0 & 0 & 0 & \ldots & 0 \\ 0 & g_r & \ldots & g_1 & g_0 & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & g_r & g_{r-1} & & \ldots & 0 \\ 0 & 0 & \ldots & 0 & g_r & & \ldots & g_0 \end{bmatrix}.$$

Очевидно, что строки матрицы $\mathbf{G}$ линейно независимы и ее ранг равен $n-r$, что совпадает с размерностью кода.

Мы придем к другому представлению циклического кода, если предположим, что полином $f(x)$ принадлежит коду тогда и только тогда, когда элементы $\alpha_1, \alpha_2, \ldots, \alpha_r$ из поля $GF(2^m)$ будут его корнями. Тогда если $f(x) = f_{n-1}x^{n-1} + f_{n-2}x^{n-2} + \cdots + f_0$, то

$$f(\alpha_i) = f_{n-1}\alpha_i^{n-1} + f_{n-2}\alpha_i^{n-2} + \cdots + f_0 = 0$$

для $i = 1, 2, \ldots, r$. Это можно также записать в виде произведения матриц

$$[f_{n-1}, f_{n-2}, \ldots, f_0][\alpha_i^{n-1}, \alpha_i^{n-2}, \ldots, \alpha_i, 1]^T = \mathbf{0}.$$

Это условие эквивалентно тому, что α — корень полинома $f(x)$. Однако из нашего обсуждения конечных полей следует, что это эквивалентно делимости полинома $f(x)$ на минимальный полином $m_i(x)$ элемента α. Это условие для всех α_i может быть записано в виде

$$\begin{bmatrix} \alpha_1^{n-1} & \alpha_1^{n-2} & \ldots & \alpha_1 & 1 \\ \alpha_2^{n-1} & \alpha_2^{n-2} & \ldots & \alpha_2 & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \alpha_{r-1}^{n-1} & \alpha_{r-1}^{n-2} & \ldots & \alpha_{r-1} & 1 \\ \alpha_r^{n-1} & \alpha_r^{n-2} & \ldots & \alpha_r & 1 \end{bmatrix} \begin{bmatrix} f_{n-1} \\ f_{n-2} \\ \vdots \\ f_1 \\ f_0 \end{bmatrix} = \mathbf{0}^T,$$

откуда мы делаем вывод, что проверочная матрица равна

$$\mathbf{H} = \begin{bmatrix} \alpha_1^{n-1} & \alpha_1^{n-2} & \ldots & \alpha_1^1 & \alpha_1^0 \\ \alpha_2^{n-1} & \alpha_2^{n-2} & \ldots & \alpha_2^1 & \alpha_2^0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \alpha_{r-1}^{n-1} & \alpha_{r-1}^{n-2} & \ldots & \alpha_{r-1}^1 & \alpha_{r-1}^0 \\ \alpha_r^{n-1} & \alpha_r^{n-2} & \ldots & \alpha_r^1 & \alpha_r^0 \end{bmatrix}.$$

i-я строка матрицы $\mathbf{H}$ означает, что α_i является корнем каждого полинома кода. Рассмотрим внимательнее i-ю строку матрицы. Все полиномы, имеющие корень α_i, составляют пространство, ортогональное пространству строк матрицы

$$[\,\alpha_i^{n-1} \quad \alpha_i^{n-2} \quad \ldots \quad \alpha_i^1 \quad \alpha_i^0\,], \qquad (***)$$

а поскольку это пространство состоит в точности из тех полиномов, которые делятся на m_i, минимальный полином элемента α_i, где $\deg[m_i(x)] = m_i$, оно является идеалом. [Матричная форма $(***)$ легко получается, если заменить каждую степень элемента α_i столбцом,

соответствующим его векторному представлению, как в табл. 3.3.1.] Поскольку размерность ортогонального пространства матрицы $(***)$ равна $n - m_i$, размерность пространства ее строк равна m_i. Отметим, что коэффициенты полиномов, представляющих кодовые слова, лежат в $GF(2)$, в то время как элементы α_i находятся в $GF(2^{m_i})$. Однако мы упомянули, что элементы поля $GF(2^{m_i})$ можно рассматривать как векторы, имеющие m_i компонент из $GF(2)$, следовательно, размерность пространства строк матрицы $(***)$ над $GF(2)$ равна m_i. См. также приведенный ниже пример.

Если α_i и α_j имеют одинаковые минимальные полиномы, то их ортогональные пространства совпадают, и, следовательно, пространства строк соответствующих матриц вида $(***)$, рассматриваемых как пространства над $GF(2)$, также совпадают. Итак, для того чтобы построить матрицу $\mathbf{H}$, *нам нужно знать только один корень для каждого неприводимого сомножителя полинома* $g(x)$.

Пример. Мы хотим построить двоичный циклический код, для которого полином $f(x)$ будет кодовым словом тогда и только тогда, когда его корнями являются элементы $\alpha, \alpha^2, \ldots, \alpha^6$, где α — примитивный элемент поля $GF(2^4)$. Минимальный полином $m_1(x)$ элемента α имеет также корни α^2, α^4, α^8. Аналогично, $m_2(x)$, минимальный полином элемента α^3, имеет корни α^6, α^{12}, α^9, и $m_3(x)$, минимальный полином элемента α^5, имеет также корень α^{10}. Таким образом,

$$g(x) = m_1(x)m_2(x)m_3(x),$$

и достаточно проверить, что каждый полином $f(x)$ имеет корни α, α^3 и α^5. Значит, искомый код — это ортогональное пространство матрицы

$$\mathbf{H}=\begin{bmatrix} \alpha^{14} & \alpha^{13} & \alpha^{12} & \ldots & \alpha^{1} & \alpha^{0}=1 \\ (\alpha^3)^{14}=\alpha^{12} & (\alpha^3)^{13}=\alpha^{9} & (\alpha^3)^{12}=\alpha^{6} & \ldots & (\alpha^3)^{1}=\alpha^{3} & (\alpha^3)^{0}=1 \\ (\alpha^5)^{14}=\alpha^{10} & (\alpha^5)^{13}=\alpha^{5} & (\alpha^5)^{12}=1 & \ldots & (\alpha^5)^{1}=\alpha^{5} & (\alpha^5)^{0}=1 \end{bmatrix}$$

или, принимая во внимание векторное представление каждой степени

элемента α (см. табл. 3.3.1), матрице $\mathbf{H}$ можно придать вид

$$\mathbf{H} = \left[\begin{array}{cccccccccccccc}
1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
 & & & & & & & & & & & & & & \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
 & & & & & & & & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1
\end{array}\right].$$

Разложение полинома $x^{15}-1$ на неприводимые множители имеет вид

$$x^{15}-1 = (x-1)(x^2+x+1)(x^4+x^3+x^2+x+1)(x^4+x^3+1)(x^4+x+1),$$

и легко убедиться, что $m_1(x) = x^4 + x + 1$, где $m_1(\alpha) = 0$, $m_2(x) = x^4 + x^3 + x^2 + x + 1$, где $m_2(\alpha^3) = 0$, и $m_3(x) = x^2 + x + 1$, где $m_3(\alpha^5) = 0$. (В этом месте читатель должен обратить внимание на потребность эффективной процедуры разложения на множители полиномов с коэффициентами из конечного поля; эта тема обсуждается в гл. 6.)

Покажем теперь, что $(2^m - 1, 2^m - m - 1, 3)$-коды Хэмминга, с которыми мы уже сталкивались, эквивалентны циклическим кодам.

Пусть α — примитивный элемент поля $GF(2^m)$, и рассмотрим код с проверочной матрицей

$$\mathbf{H} = (\alpha^{q-2}, \alpha^{q-3}, \ldots, \alpha, 1), \qquad q = 2^m.$$

Если степени элемента α представлены векторами (столбцами) длины m над $GF(2)$, то каждый ненулевой вектор длины m появляется ровно один раз как столбец матрицы $\mathbf{H}$. Таким образом, $\mathbf{H}$ действительно описывает код Хэмминга. Его можно описать как циклический код, если мы скажем, что полином $f(x)$ — кодовое слово тогда и только тогда, когда α является корнем полинома $f(x)$. Минимальный полином элемента α является c-примитивным полиномом, следовательно, порождающий полином $g(x)$ c-примитивен. Аналогично, каждый код, порожденный c-примитивным полиномом, является кодом Хэмминга.

Пример. Пусть $m = 4$, и пусть α — корень c-примитивного полинома $x^4 + x + 1$. Тогда проверочная матрица (15,11)-кода Хэмминга, порожденного этим полиномом, имеет вид

$$\mathbf{H} = (\alpha^{14}, \alpha^{13}, \ldots, \alpha^2, \alpha, 1)$$
$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Опишем теперь две процедуры кодирования для циклических кодов, где хорошо видна простота этих кодов. Общая характеристика обоих методов состоит в том, что регистр принимает k информационных битов из источника и без промедления отправляет их в передающий канал. За каждым таким блоком информационных битов следуют $n - k$ проверочных битов. Очевидно, что пока готовятся проверочные биты, регистр не может принимать следующий блок информационных битов, и, таким образом, либо источник должен иметь возможность приостанавливать и повторно начинать передачу, либо должен использоваться временно́й буфер.

Первая из описываемых процедур кодирования использует k-разрядный регистр, в то время как вторая использует $n-k$-разрядный регистр. Если проверочных битов больше, чем информационных, то предпочтительнее первый метод, в противном случае более экономичен второй. Обе процедуры выдают *одно и то же* кодовое слово.

Кодирование циклического (n, k)-кода, порожденного полиномом $g(x)$, $\deg[g(x)] = n - k$, достигается с помощью схемы, изображенной на рис. 4.1.5, где используется k-разрядный регистр; соединения регистра соответствуют полиному $h(x) = (x^n - 1)/g(x)$. Информационные биты первоначально размещаются в k элементах памяти, а затем происходят циклические сдвиги. Первые k битов, выходящие из регистра, являются информационными, а остальные $n - k$ — проверочными для кодового слова из n битов. (См. упр. 5 к этому разделу.)

При использовании $n-k$-разрядного регистра процедура кодирования выглядит следующим образом: кодовое слово может быть образовано умножением произвольного полинома степени не более $k - 1$, коэффициенты которого — информационные биты, на полином $g(x)$, который порождает код. Это может быть достигнуто использованием либо схемы, показанной на рис. 3.3.2, либо схемы на рис. 3.3.3. Информационные биты теперь меняются и могут быть получены из

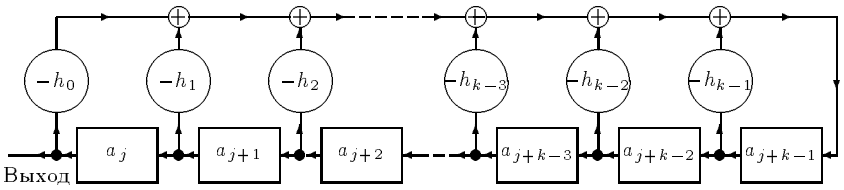


Рис. 4.1.5.
Кодирование циклического (n, k)-кода.

кодового слова делением соответствующего полинома на $g(x)$; для этой цели может использоваться схема рис. 3.3.6. (См. упр. 6 к этому разделу.)

Процедура обнаружения ошибок для циклических кодов также совсем проста. Если $v(x)$ — принятый полином, то достаточно проверить, равен или нет нулю полином-синдром $r(x) = g(x)v(x)$ (см. упр. 7 к этому разделу). Таким образом, для циклических кодов кодирование и обнаружение ошибок выполняются за линейное время относительно длины кода.

В заключение мы обсудим возможности циклических кодов исправлять ошибки, подойдя таким образом к БЧХ-кодам. Мы говорим, что вектор ошибок длины n — это *пакет ошибок* длины b, тогда и только тогда, когда все ненулевые компоненты не разбросаны по всей длине n, а сконцентрированы в интервале длины b. Мы рассмотрим некоторые теоремы, касающиеся корректирующей способности циклических кодов относительно пакетов ошибок.

Теорема 4.1.22. В циклическом (n, k)-коде никакое кодовое слово не является пакетом ошибок длины не более $n - k$. Таким образом, каждый циклический код может обнаруживать любой пакет ошибок длины, не превосходящей $n - k$.

Доказательство. Пусть $r(x)$ — пакет ошибок длины $\leq n - k$, и предположим, что обсуждаемый циклический (n, k)-код порожден полиномом $g(x)$ степени $n - k$. Кроме того, предположим, что первый отличный от нуля коэффициент полинома $r(x)$ — это коэффициент при x^j. Тогда $r(x) = x^j r_0(x)$, где $\deg[r_0(x)] < n - k$, поскольку $r(x)$ — пакет ошибок длины $\leq n - k$. Далее, поскольку $g(x)|(x^n - 1)$, $g(x)$ не делится на x и, следовательно, полиномы x^j и $g(x)$ взаимно просты. Кроме того, из того, что $g(x)$ должен делить $r(x)$, следует, что если $r(x)$ — кодовое слово, оно обязательно делит $r_0(x)$, но это противоречит предположению, что $\deg[r_0(x)] < n - k$. Таким образом, $r(x)$ не может быть кодовым словом. □

Следующая теорема дает нижнюю границу кодового расстояния любого циклического кода. В случае БЧХ-кодов, определяемых ниже, порождающие полиномы выбираются так, чтобы гарантировать относительно большое кодовое расстояние.

Теорема 4.1.23. Пусть $g(x)$ — порождающий полином двоичного циклического кода длины n, и пусть $\alpha^{e_1}, \alpha^{e_2}, \ldots, \alpha^{e_{n-k}}$ — корни полинома $g(x)$, где α — элемент порядка n в некотором поле Галуа. Тогда кодовое расстояние этого кода больше, чем максимальное число последовательных целых чисел в множестве $e = (e_1, e_2, \ldots, e_{n-k})$.

Доказательство. Предположим, что $m_0, m_0+1, \ldots, m_0+d_0-2$ — наибольшее подмножество последовательных целых чисел в множестве e. Мы уже упоминали, что циклический код с корнями $\alpha^{e_1}, \alpha^{e_2}, \ldots, \alpha^{e_{n-k}}$ — это пространство, ортогональное матрице

$$\mathbf{H} = \begin{bmatrix} (\alpha^{e_1})^{n-1} & (\alpha^{e_1})^{n-2} & \ldots & \alpha^{e_1} & 1 \\ (\alpha^{e_2})^{n-1} & (\alpha^{e_2})^{n-2} & \ldots & \alpha^{e_2} & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ (\alpha^{e_{n-k}})^{n-1} & (\alpha^{e_{n-k}})^{n-2} & \ldots & \alpha^{e_{n-k}} & 1 \end{bmatrix}.$$

Теперь если мы докажем, что никакая линейная комбинация $d_0 - 1$ столбцов подматрицы

$$\mathbf{S} = \begin{bmatrix} (\alpha^{m_0})^{n-1} & (\alpha^{m_0})^{n-2} & \ldots & \alpha^{m_0} & 1 \\ (\alpha^{m_0+1})^{n-1} & (\alpha^{m_0+1})^{n-2} & \ldots & \alpha^{m_0+1} & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ (\alpha^{m_0+d_0-2})^{n-1} & (\alpha^{m_0+d_0-2})^{n-2} & \ldots & \alpha^{m_0+d_0-2} & 1 \end{bmatrix}$$

не обращается в нуль, то, очевидно, то же самое будет верно для столбцов матрицы $\mathbf{H}$, и по теореме 4.1.12 кодовое расстояние будет не меньше, чем d_0.

Тот факт, что выписанная матрица $\mathbf{S}$ обладает нужным свойством, может быть доказан демонстрацией того, что определитель матрицы, полученной взятием любого множества $d_0 - 1$ столбцов матрицы $\mathbf{S}$, отличен от нуля. Действительно, такой определитель может быть записан в виде

$$\det \begin{bmatrix} (\alpha^{m_0})^{j_{d_0-1}} & (\alpha^{m_0})^{j_{d_0-2}} & \ldots & (\alpha^{m_0})^{j_1} \\ (\alpha^{m_0+1})^{j_{d_0-1}} & (\alpha^{m_0+1})^{j_{d_0-2}} & \ldots & (\alpha^{m_0+1})^{j_1} \\ \ldots & \ldots & \ldots & \ldots \\ (\alpha^{m_0+d_0-2})^{j_{d_0-1}} & (\alpha^{m_0+d_0-2})^{j_{d_0-2}} & \ldots & (\alpha^{m_0+d_0-2})^{j_1} \end{bmatrix},$$

который, в свою очередь, может быть записан как

$$\alpha^{m_0(j_1+j_2+\cdots+j_{d_0-1})} \det \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_s \\ x_1^2 & x_2^2 & \dots & x_s^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{s-1} & x_2^{s-1} & \dots & x_s^{s-1} \end{bmatrix},$$

где $s = d_0 - 1$ и $x_i = \alpha^{j_{d_0}-i}$. Последний определитель, однако, является определителем матрицы Вандермонда и равен $\prod_{i>j}(x_i - x_j)$. Он обращается в нуль, только когда $x_i = x_j$ для некоторых i, j, но в нашем случае это невозможно. □

Определение 4.1.24. Пусть α — элемент поля $GF(2^m)$. Тогда для данных m_0 и d_0 код, порожденный полиномом $g(x)$, является БЧХ-кодом в том и только том случае, когда $g(x)$ — минимальный полином над $GF(2)$, корнями которого являются элементы α^i, $i = m_0, m_0 + 1, \dots, m_0 + d_0 - 2$.

Длина n такого кода — это наименьшее общее кратное (lcm) порядков его корней. Это эквивалентно утверждению, что n равно порядку e элемента α (за исключением тривиального случая, когда дан только один корень α^{m_0}). Последнее утверждение справедливо, так как мы имеем (поскольку n равно lcm порядков корней)

$$(\alpha^{m_0})^n = 1, \quad (\alpha^{m_0+1})^n = 1,$$

откуда заключаем, что $\alpha^n = 1$ и что порядок e элемента α делит n, и, следовательно, $e \le n$. С другой стороны, если $\alpha^e = 1$, то $(\alpha^j)^e = 1$ для всех j. Таким образом, порядок любого элемента α^j делит e, откуда мы заключаем, что $e = n$, поскольку n, lcm, не может быть больше e.

Число информационных битов, так же как число проверочных битов, может быть найдено с помощью общего метода, представленного нами для циклических кодов. Из теоремы 4.1.23 нам известно, что кодовое расстояние этих кодов больше d_0.

Наиболее важные БЧХ-коды получаются, если взять в качестве α примитивный элемент поля $GF(2^m)$ и положить $m_0 = 1$ и $d_0 = 2t_0 + 1$. Тогда корнями полинома являются $\alpha, \alpha^2, \alpha^3, \dots, \alpha^{2t_0}$. Однако, поскольку каждая четная степень α имеет тот же минимальный полином, что и некоторая предыдущая нечетная степень α (например, α^{10} имеет тот же минимальный полином, что и α^5), мы можем сказать, что $g(x)$ — полином с корнями $\alpha, \alpha^3, \dots, \alpha^{2t_0-1}$. Пусть

$m_1(x), m_3(x), \ldots, m_{2t_0-1}(x)$ — соответствующие минимальные полиномы. Тогда

$$g(x) = \mathrm{lcm}[m_1(x), m_3(x), \ldots, m_{2t_0-1}(x)].$$

Из наших предыдущих рассуждений видно, что $\deg[m_i(x)] < m$, где m определяет поле $GF(2^m)$, откуда следует, что $\deg[g(x)] < mt_0$, и, таким образом, код имеет меньше, чем mt_0, проверочных битов. Сделанные выше замечания подытожены в следующей теореме.

Теорема 4.1.25. Для любой пары положительных целых чисел m и $t_0 < n/2$ существует двоичный БЧХ-код длины $n = 2^m - 1$, который исправляет все комбинации не более t_0 ошибок и имеет не больше, чем mt_0, проверочных битов.

Пример. Рассмотрим поле $GF(2^4)$, и пусть α — примитивный элемент этого поля, т.е. $n = 2^4 - 1 = 15$. Рассмотрим, кроме того, БЧХ-код, состоящий из всех полиномов, имеющих корни α и α^3. Минимальные полиномы для α и α^3 суть $m_1(x) = x^4 + x + 1$ и $m_2(x) = x^4 + x^3 + x^2 + x + 1$ соответственно. Оба этих полинома делят $x^{15} - 1$. Циклический код C над $GF(2)$ определяется порождающим полиномом $g(x) = m_1(x)m_2(x)$ степени 8 и проверочной матрицей

$$\mathbf{H} = \begin{bmatrix} \alpha^{14} & \alpha^{13} & \alpha^{12} & \ldots & \alpha^{1} & \alpha^{0}{=}1 \\ (\alpha^3)^{14}{=}\alpha^{12} & (\alpha^3)^{13}{=}\alpha^{9} & (\alpha^3)^{12}{=}\alpha^{6} & \ldots & (\alpha^3)^{1}{=}\alpha^{3} & (\alpha^3)^{0}{=}1 \end{bmatrix}.$$

Из сказанного выше мы знаем, что кодовое расстояние этого кода ≥ 5 (в этом случае $t_0 = 2$); это подразумевает, что код может исправлять до двух ошибок; C — это $(15, 7)$-код. Далее $\mathbf{v} \in C$ тогда и только тогда, когда $\mathbf{S}(\mathbf{v})$, синдром вектора $\mathbf{v}$, равен $\mathbf{0}$, т.е. $\mathbf{v} \in C$ тогда и только тогда, когда $\mathbf{H}\mathbf{v}^T = \mathbf{0}^T$, или тогда и только тогда, когда $\mathbf{S}_1 = \mathbf{S}_3 = \mathbf{0}$, где $\mathbf{S}_1 = \sum_{1 \leq i \leq 14} v_i \alpha^i$ и $\mathbf{S}_3 = \sum_{1 \leq i \leq 14} v_i \alpha^{3i}$ — компоненты синдрома $\mathbf{S}(\mathbf{v}) = (\mathbf{S}_1, \mathbf{S}_3)^T$ вектора $\mathbf{v}$ относительно $\mathbf{H}$. Если мы используем векторное представление для элементов поля $GF(2^4)$, то получим $(8{\times}15)$-матрицу $\mathbf{H}$, подобную которой мы видели в предыдущем примере.

Предположим, что принятый вектор $\mathbf{v} = (v_0, v_2, \ldots, v_{14})$ — вектор с самое большее двумя ошибками. Пусть $e = x^{e_1} + x^{e_2}$, $0 \leq e_1, e_2 \leq 14$, $e_1 \neq e_2$ — вектор ошибок. Мы имеем

$$\mathbf{S}_1 = \alpha^{e_1} + \alpha^{e_2}, \quad \mathbf{S}_1 = \alpha^{3e_1} + \alpha^{3e_2}.$$

Кроме того, если $x_1 = \alpha^{e_1}$ и $x_2 = \alpha^{e_2}$, то $\mathbf{S}_1 = x_1 + x_2$ и $\mathbf{S}_3 = x_1^3 + x_2^3$, и после некоторых манипуляций мы видим, что если встретились две ошибки, то $1/x_1$ и $1/x_2$ — два корня в $GF(2^4)$ *полинома локаторов ошибок*

$$\sigma = 1 + \mathbf{S}_1 x + \left(\mathbf{S}_1^2 + \frac{\mathbf{S}_3}{\mathbf{S}_1} \right) x^2.$$

Если случилась только одна ошибка, то $\mathbf{S}_1 = x_1$, $\mathbf{S}_3 = x_1^3$ и $\mathbf{S}_1^3 + \mathbf{S}_3 = 0$. Следовательно, $\sigma = 1 + \mathbf{S}_1 x$. И наконец, если ошибок не было, то $\mathbf{S}_1 = \mathbf{S}_3 = 0$.

Из приведенного примера мы видим, что, для того чтобы иметь возможность исправлять две ошибки, мы должны специальным образом удвоить число строк матрицы $\mathbf{H}$, используемой в кодах Хэмминга. В литературе имеются более общие процедуры декодирования для исправления любого числа ошибок; см. например, (Berlekamp, 1968; Childs, 1979; Mackiw, 1985; MacWilliams, 1977; Peterson и др., 1972).

4.2

Криптография — общие понятия

Криптология —- это искусство проектирования и взлома секретных систем; часть, связанная с проектированием, называется *криптографией*, а «взламывающая» часть —- *криптографическим анализом*, или *криптоанализом*. Весь процесс можно рассматривать параллельно процессу кодирования и декодирования. Вновь будем рассматривать отправителя, который хочет сообщить «доверительную» информацию получателю по ненадежному каналу связи. Ненадежность канала может быть вызвана несанкционированным перехватчиком, имеющим доступ к каналу, цели которого состоят в том (но не всегда только в том), чтобы: (1) нарушить секретность сообщения, (2) сбить с толку получателя искаженным сообщением и (3) обмануть отправителя или получателя, или обоих относительно личности противоположной стороны.

Первая из перечисленных опасностей —- наиболее широко известная проблема криптографии. Защита от перехватчика, преследующего последние две цели, выдвинулась сравнительно недавно и включает в себя *установление полномочий* и *обеспечение сохранности*. Например, проблема установления полномочий возникает в процедуре входа в многопользовательских компьютерных системах, в то время

как проблема обеспечения сохранности возникает при электронной передаче фондов. Подлинность нарушается, когда сообщение C' на рис. 4.2.1 составлено перехватчиком, в то время как получатель убежден, что оно пришло от отправителя. Сохранность нарушается, когда $M \neq M'$ и ни отправитель, ни получатель не могут обнаружить изменений сообщения C.

Проблема тайнописи очень стара, и много попыток защитить доверительные сообщения, особенно военные и дипломатические, составили ее богатую историю [большой интерес представляет книга Кана (Kahn, 1967)]. В то время как в задаче кодирования основная цель — быстрая и правильная передача сообщения по зашумленному каналу, цель криптологии — безопасная передача первоначального сообщения, измененного таким образом, чтобы несанкционированный перехватчик не смог понять сообщение. Это достигается с помощью криптосистемы типа той, которая показана на рис. 4.2.1.

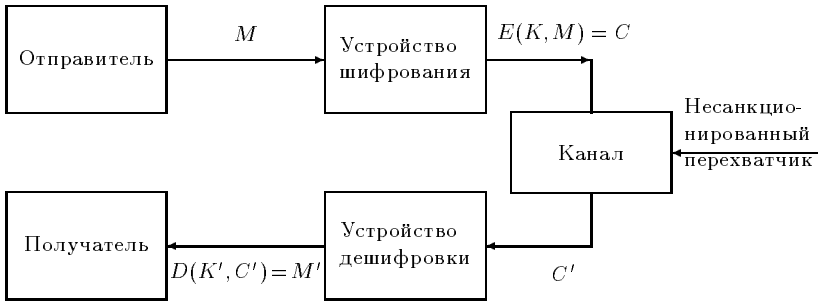


Рис. 4.2.1.
Криптосистема для передачи сообщения M.

Криптографическая система (*криптосистема*, или *шифр*) состоит в преобразовании сообщения M, которое называется *открытым текстом*, с помощью *шифровальной* схемы таким образом, что только законный получатель может обратить это преобразование и восстановить сообщение. Шифровальная схема обращается к функции шифрования E, которой кроме открытого текста M требуется также шифровальный *ключ* K, являющийся параметром, специфическим для каждого преобразования. (Более точное определение ключа дано ниже.) Функция шифрования определяется алгоритмом, и результат процесса шифрования $E(K, M) = C$ называется *шифрованным текстом* или *криптограммой*. Текст C передается по незащищенному каналу, где перехватчик может рассматривать его, запоминать, работать с ним и, наконец, заменить его на C'. *Дешифратор*, или схема

дешифровки, использует алгоритм расшифровки D, который берет в качестве аргументов полученный шифрованный текст C' (его подлинность сомнительна) и ключ для дешифровки K'. Ключ и шифрованный текст должны определять открытый текст однозначно. Отметим, что K, M не обязательно совпадают с K', M', где $M' = D(K', C')$, хотя в классической криптографии $K = K'$. Мы называем системы, где $K = K'$, системами *единого* ключа или *симметричными криптосистемами*, чтобы отличать их от более новых систем *открытого ключа*, или *асимметричных криптосистем*, где используются два различных ключа.

Коды и шифры — единственные две возможности для процесса шифровки/дешифровки. Использование кодов для шифрования сообщений означает замену некоторых или всех слов и фраз кодовыми словами и кодовыми фразами, полученными из специальной книги, напоминающей словарь; на самом деле слово *код* относится только к такой криптосистеме, хотя термины *секретный код* и *взлом кода* используются во всех разновидностях тайнописи. Иначе говоря, код должен иметь возможность установить семантическое содержание любого сообщения, которое можно передать по каналу, и как отправитель, так и получатель должны иметь кодовую книгу. При условии, что кодовая книга надежно защищена, такое сообщение чрезвычайно трудно (если вообще возможно) взломать. Однако передача сообщения невозможна, если фраза не включена в кодовую книгу. Напротив, при использовании шифра возможна передача произвольных сообщений, потому что шифр — это алгоритм, присваивающий новые символы шифрованного текста символам или группам символов открытого текста. Шифры представляют большой интерес, потому что связь между шифрованным текстом и открытым текстом не произвольная и может быть обнаружена с помощью криптоанализа.

4.2.1. Симметричные криптосистемы (единого ключа)

В классической криптографии имеется два основных преобразования открытого текста сообщений вместе с их комбинациями:

1. *Транспозиции*, или *перестановки*, переупорядочивают группу символов в соответствии с некоторым правилом, не меняя их, т.е. если сообщение M составлено из m блоков, $M = B_1 B_2 \ldots B_m$, где каждый блок B_i содержит n символов, $B_i = b_{i,1} b_{i,2} \ldots b_{i,n}$, $i = 1, 2, \ldots, m$, то шифрованный текст — это $C = C_1 C_2 \ldots C_m$, где $C_i = b_{i,\varpi(1)} b_{i,\varpi(2)} \ldots b_{i,\varpi(n)}$ для каждого $i = 1, 2, \ldots, m$, а ϖ — фиксированная перестановка целых чисел $1, 2, \ldots, n$.

2. *Подстановки* замещают символы открытого текста соответствующими символами из алфавита шифрованного текста (ключ задает отображение), т.е. если сообщение — это $M = a_1 a_2 \dots a_n$, то шифрованный текст $C = f_1(a_1) f_2(a_2) \dots f_n(a_n)$ определяется с помощью n отображений f_i, $i = 1, 2, \dots, n$, из алфавита открытого текста в алфавит шифрованного текста.
3. Разумеется, комбинируя (1) и (2), мы получаем шифр подстановки/перестановки (S/P)[1)].

Пример (транспозиция). Предположим, что мы хотим зашифровать сообщение «computer algebra». Первый пример транспозиции — *записать текст задом наперед*, традиционным способом группируя его по пять символов. Так, мы имеем

arbeg laret upmoc

Другой пример транспозиционного шифра — *шифр изгороди*, где мы расписываем текст побуквенно в две строки, а затем читаем его построчно, т.е. мы получаем

c	*m*	*u*	*e*	*a*	*g*	*b*	*a*
o	*p*	*t*	*r*	*l*	*e*	*r*	

и шифрованный текст имеет вид «cmuea gbaop trler». Способы взлома транспозиционных шифров можно найти в книгах (Kahn, 1967, pp. 225–226) и (Sinkov, 1968, ch. 5).

Если используется только один алфавит для шифрованных сообщений, то криптосистема называется *одноалфавитной*. Криптосистемы, в которых буква шифрованного текста может представлять более одной буквы открытого текста, называется *многоалфавитными*. Мы различаем также *потоковые* и *блочные* шифры. Потоковые шифры рассматривают открытый текст как последовательность символов, подлежащих шифровке, в то время как блочные шифры делят сообщения на блоки равной длины и производят шифрование, действуя на блоках символов открытого текста. В потоковом шифре основная допустимая операция на сообщении — подстановка одного символа вместо другого, в то время как в блочном шифре помимо подстановки мы имеем перестановку. Хотя потоковые шифры сохраняют свое значение для многих приложений, блочные шифры последнее время оказывают наибольшее воздействие на криптографию. Рассмотрим сначала потоковые шифры.

[1)] S/P происходит от английских «substitution» и «permutation». — *Прим. перев.*

Определение 4.2.1. Пусть A и B — алфавиты открытого и шифрованного текста соответственно. *Ключ* — инъективное (взаимно однозначное) отображение из A в B. Ключ называется *фиксированным*, если это отображение одно и то же для каждого элемента из A, в противном случае ключ *переменный*.

Фиксированный ключ $f : A \to B$ можно распространить на строки элементов из A, т.е. на слова с буквами из A, полагая $a_1a_2\ldots a_m \to f(a_1)f(a_2)\ldots f(a_m)$; аналогично, переменный ключ $f_1, f_2, \ldots$ можно распространить, полагая $a_1a_2\ldots a_m \to f_1(a_1)f_2(a_2)\ldots f_m(a_m)$. Упомянутые в определении алфавиты могут состоять из букв английского алфавита или n-ок букв, или элементов кольца $\mathbb{Z}_m$ ($m = 2$ или $m = 26$, или $m = 96$, как в некоторых коммерческих системах). В качестве упражнения читателю оставляется проверка того, что отображение $a \to an + k \pmod m$ из $\mathbb{Z}_m$ в себя инъективно (взаимно однозначно) тогда и только тогда, когда $\gcd(m, n) = 1$.

Определение 4.2.2. Отображение $a \to an + k \pmod m$ кольца $\mathbb{Z}_m$ в себя при фиксированных n, k из $\mathbb{Z}_m$ и $\gcd(m, n) = 1$ называется *модулярным шифром*.

При $n = 1$, $k = 3$ и $m = 26$ мы получаем шифр Цезаря — код, представляющий исторический интерес, поскольку, согласно легенде, он использовался римским императором Гаем Юлием Цезарем. Заметим, что шифр цезаря — циклический сдвиг алфавита на три буквы.

Пример. Отождествим a с 0, b с 1, c с 2, и т.д.:

a	b	c	d	e	f	g	h	i	j	k	l	m	n
0	1	2	3	4	5	6	7	8	9	10	11	12	13
o	p	q	r	s	t	u	v	w	x	y	z		
14	15	16	17	18	19	20	21	22	23	24	25		

Используя шифр Цезаря, $a \to a + 3$, и переписывая текст традиционным способом группами по пять символов, мы сообщение «computer algebra», эквивалентное $2, 14, 12, 15, 20 \quad 19, 4, 17, 0, 11 \quad 6, 4, 1, 17, 0$, зашифровываем как $5, 17, 15, 18, 23 \quad 22, 7, 20, 3, 14 \quad 9, 7, 4, 20, 3$ или, эквивалентно, как «ftpsx whudo jheud».

Одноалфавитные подстановки используют только один ключ, и их можно легко взломать, наблюдая частоту распределения символов в шифрованном тексте. Это — легкая задача, потому что имеются

таблицы различных частот букв, например частоты первых букв в слове, частоты последних букв в слове, частоты диграфов (т.е. частоты сочетания: буква a, за которой следует b) и т.д. Таблица, приведенная на рис. 4.2.2, была получена Синковым из выборки из 1000 букв [см. приложения в книге (Sinkov, 1968)].

	%		%		%
Буква	Частота	Буква	Частота	Буква	Частота
a	7.3	j	0.2	s	6.3
b	0.9	k	0.3	t	9.3
c	3.0	l	3.5	u	2.7
d	4.4	m	2.5	v	1.3
e	13.0	n	7.8	w	1.6
f	2.8	o	7.4	x	0.5
g	1.6	p	2.7	y	1.9
h	3.5	q	0.3	z	0.1
i	7.4	r	7.7		

Рис. 4.2.2.
Относительные частоты букв в английском языке.

Одноалфавитный шифр может быть усилен, если мы используем многоалфавитный шифр подстановки, скрывающий частоты букв за счет кратных подстановок. Здесь при шифровании сообщения используется более одного алфавита, и ключ включает указание, какая подстановка должна использоваться для каждого символа. Эти шифры известны также как *шифры Виженера*, по фамилии французского криптографа шестнадцатого столетия Блеза де Виженера.

Более точно, многоалфавитный шифр подстановки с периодом p состоит из p шифровальных алфавитов B_i и отображений $f_i : A \to B_i$, $i = 1, 2, \ldots, p$, определяемых ключом. Ключ — это чаще всего слово $K = k_1 k_2 \ldots k_p$ и $f_i(a) \equiv a + k_i \pmod{26}$. Открытое сообщение $M = a_1 a_2 \ldots a_p a_{p+1} \ldots a_{2p} \ldots$ зашифровывается как $f_1(a_1) f_2(a_2) \ldots f_p(a_p) f_1(a_{p+1}) \ldots f_p(a_{2p}) \ldots$ повторением последовательности отображений $f_1, f_2, \ldots, f_p$ для каждых p символов. Для шифрования может использоваться квадрат Виженера с буквами a, b, ..., z открытого алфавита в первой строке и первом столбце, т.е.

ab ... yz
bc ... za
. . . . . .
yz ... wx
za ... xy

Процедура шифрования может быть упрощена, если мы примем во внимание, что можно использовать не все строки (алфавиты) квадрата Виженера. Например, предположим, что ключ — «alkis», в этом случае период $p = 5$; тогда отображения f_i даются следующей простой таблицей, где ключ появляется в левом столбце.

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*q*	*r*	*s*	*t*	*u*	*v*	*w*	*x*	*y*	*z*
a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*q*	*r*	*s*	*t*	*u*	*v*	*w*	*x*	*y*	*z*
l	*m*	*n*	*o*	*p*	*q*	*r*	*s*	*t*	*u*	*v*	*w*	*x*	*y*	*z*	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*
k	*l*	*m*	*n*	*o*	*p*	*q*	*r*	*s*	*t*	*u*	*v*	*w*	*x*	*y*	*z*	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*
i	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*q*	*r*	*s*	*t*	*u*	*v*	*w*	*x*	*y*	*z*	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*
s	*t*	*u*	*v*	*w*	*x*	*y*	*z*	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*q*	*r*

В этом случае сообщение «computer algebra» шифруется как «czwxm tpbid gplzs», где i-я буква открытого текста отображается в символ, находящийся в соответствующем столбце в i-м $\mod 5$ алфавите шифра; например, буква m открытого текста отображается в букву w, которая находится в столбце, начинающемся буквой m, в алфавите, начинающемся буквой k.

Из последнего примера видно, что ключ «alkis» соответствует последовательности чисел 0, 11, 10, 8, 18, и, следовательно, шифрование может быть осуществлено последовательной записью под открытым текстом чисел 0, 11, 10, 8, 18, 0, 11, 10, 8, 18, 0, 11, 10, 8, 18, ... и прибавлением их по модулю 26 к числам, соответствующим буквам открытого текста.

Процедура дешифровки выглядит следующим образом: число, соответствующее i-й букве открытого текста, получается прибавлением по модулю 26 числа, соответствующего i-й букве шифрованного текста, к m-дополнению числа, соответствующего i-й $(\mod p)$ букве ключа. (Напомним, что получатель также знает ключ.) Например, если $m = 26$, полученное сообщение — «czwxm tpbid gplzs», ключ — «alkis», или 0, 11, 10, 8, 18, и мы хотим вычислить вторую букву открытого текста, то прибавляем по модулю 26 число 25, соответствующее букве z шифрованного текста, к числу 15, являющемуся дополнением до 26 второй буквы ключа $(26 - 11 = 15)$ и получаем 14 или «o».

Шифры Виженера считались в те дни невзламываемыми, но в 1863 г. прусский офицер по фамилии Ф.В. Касисский (F.W. Kasiski) нашел простой теоретико-числовой метод поиска длины ключа и опубликовал этот результат. В многоалфавитных шифрах так же, как в шифре Цезаря, применяется сдвиг, но длина сдвига меняется, обычно

периодически. Изменение сдвигов выравнивает частоты букв, обрекая на неуспех методы анализа, используемые для взлома шифров Цезаря, но характеристические частоты сохраняются в подпоследовательностях зашифрованного сообщения, соответствующих повторениям в ключевой последовательности. Если нам удастся определить длину периода ключа, мы сможем определить буквы с помощью частотного анализа.

Период ключа может быть обнаружен поиском повторяющихся блоков в шифрованном тексте. Часть из них носит случайный характер, но основная масса получается из соответствия между повторяемыми словами или подсловами в открытом тексте и повторами в последовательности ключа. Когда это имеет место, расстояние между повторами будет кратным периоду ключа. Например, открытый текст «send more men and more arms» при использовании шифровального ключа «bhenf» даст шифрованный текст «tlrqr pyizj ohrqr pyinw nz» (если мы оставим последний блок неполным). В этом случае расстояние между двумя вхождениями равно 10, что означает, что длина ключа равна 10 или 5.

Другим вариантом многоалфавитных подстановочных шифров являются *шифры с автоматическим выбором ключа*, когда само сообщение (открытый текст или шифрованный текст) используется в качестве ключа. Для запуска шифра используется короткий «затравочный» ключ, обычно одна буква. Эти варианты были предложены в 1550 г. Кардано и развиты Виженером. Если мы работаем в $\mathbb{Z}_m$, то шифры с автоматическим выбором ключа определяются отображением $a_i \to b_i \equiv na_i + ca_{i-1} \pmod m$, где даны c, a_0, или $a_i \to b_i \equiv na_i + cb_{i-1} \pmod m$, где даны c, b_0; разумеется, мы выбираем n так, что $\gcd(m, n) = 1$.

Пример. Если мы перенумеруем буквы от a до z, как в предыдущем примере (т.е. $m = 26$), то сообщение «computer algebra», эквивалентное последовательности 2,14,12,15,20, 19,4,17,0,11, 6,4,1,17,0= $a_1a_2 \ldots a_{15}$, может быть зашифровано следующими двумя способами:

1. Используя преобразование $a_i \to b_i \equiv na_i + ca_{i-1} \pmod m$ с $n = 1$, $c = 1$ и $a_0 = 2$, мы получаем $b_1 = 4$, $b_2 = 16$, $(14 + 2)$, и т.д., и, наконец, мы получаем шифрованный текст 4,16,0,1,9, 13,23,21,17,11, 17,10,5,18,17.
2. Используя преобразование $a_i \to b_i \equiv na_i + cb_{i-1} \pmod m$ с $n = 1$, $c = 1$ и $b_0 = 2$, мы получаем $b_1 = 4$, $b_2 = 18$, $(14 + 4)$, и т.д. и, наконец, получаем шифрованный текст 4,18,4,19,13, 6,10,1,1,12, 18,22,23,14,14.

Другой вариант — *шифр бегущего ключа Виженера*, использующий в качестве ключа неповторяющийся текст. Первоначально эти шифры также считались невзламываемыми, но в 1883 г. Керчофс нашел общее решение для многоалфавитных подстановок без ограничений на тип или длину ключа [детали можно найти в книге (Kahn, 1967, pp. 236–237)]; более общее решение проблемы было дано Фридманом в 1918 г. Интересное изложение его идей может быть найдено в статье Гасса (Gass, 1986).

Наиболее значительный вариант шифра Виженера был предложен в 1918 г. американским инженером Вернамом (G.S. Vernam), работавшим в системе телетайпной сети AT&T (American Telephone & Telegraph). Сообщения передавались тогда с использованием двоичного кода, и Вернам предложил прибавлять по модулю 2 случайные последовательности точек и пробелов так, чтобы вся частотная информация, корреляция между символами, периодичность и тому подобное терялись. Главный недостаток этой процедуры состоит в том, что она требует обмена огромным количеством ключей заблаговременно (т.е. один символ ключа на каждый символ сообщения). Вернам первоначально предполагал использовать или короткий ключ, или линейную комбинацию коротких ключей, но оба подхода оказались уязвимыми. С одной стороны, использование короткого ключа уязвимо по результатам типа Касисского, с другой стороны, как было доказано майором американских войск связи Мауборном, использование линейных комбинаций коротких ключей может быть успешно проанализировано по существу той же техникой, которая используется против систем бегущего ключа.

И Фридман, и Мауборн пришли к выводу, что для безоговорочно надежной криптосистемы ключ должен быть таким же длинным, как сообщение, и непоследовательным (т.е. нерегулярность каждого символа ключа должна быть по крайней мере так же велика, как среднее информационное содержание на символ открытого текста). Если мы предположим, что M — верхняя граница длин всех возможных сообщений, то мы выбираем ключевую последовательность по крайней мере такой же длины, как M; все ключевые последовательности тогда равновероятны. Если работа ведется по модулю 26, открытый текст имеет вид $a_1 a_2 \ldots a_M$ и каждый символ a_i представлен одним из чисел от 0 до 25, то ключевая последовательность — это $k_1 k_2 \ldots k_M$, где k_i — случайно выбранное число между 0 и 25. Тогда каждый символ шифрованного текста равен $c_i \equiv a_i + k_i \pmod{26}$. Эта схема называется шифром *одноразового блокнота*, и такое название связано с тем, что процесс шифровки/дешифровки использует выписанные

перечни случайных чисел («листы блокнота») для получения ключевых последовательностей, и каждая ключевая последовательность используется только один раз. Как уже упоминалось, главный недостаток этой схемы состоит в том, что она требует обмена огромным количеством ключей до начала связи. Однако шифр одноразового блокнота, очевидно, невзламываем. Случайность ключа означает, что любые две последовательности сообщений одинаковой длины с равной вероятностью могут превратиться в данный шифрованный текст. В результате одноразовые блокноты используются для передачи в высшей степени доверительной информации, например, в прямой связи между Вашингтоном и Москвой.

В итоге мы видим, что сохранность возрастает при движении от шифра Цезаря к шифру одноразового блокнота и одновременно возрастает длина ключа. Ключ состоит из единственного числа в шифре Цезаря, в то время как в шифре одноразового блокнота ключ потенциально бесконечен. В невзламываемой системе все сообщения равновероятны, следовательно, шифр неразрешим.

До сих пор мы имели дело главным образом с потоковыми шифрами, где каждый символ сообщения обрабатывался индивидуально. Рассмотрим теперь *полиграфовые системы*. Это — блочные шифры, рассматривающие одновременно группы символов открытого текста. Блочные шифры разрабатывались с целью помешать простому криптоанализу с помощью статистики частот вхождений; обычно они действуют на парах, *диграфах*, тройках, *триграфах*, и в общем случае на *полиграфах*. Системы, обрабатываемые вручную, ограничивались диграфами.

Лучшая из известных систем ручного диграфового шифрования принадлежит английскому ученому Плейферу. По этой схеме перемешанная алфавитная последовательность записывается в квадрате 5×5. (Буква «j» опускается, поскольку она встречается довольно редко и может быть потом восстановлена в контексте.) Так, например, мы может взять

a	l	k	i	s
b	y	t	q	x
r	n	c	u	g
d	w	h	m	o
z	p	f	v	e

и сообщение «computer algebra» шифруется как «ghwvc qzglk osrda», при этом пара «co» отображается в пару «gh», где g находится в той же строке, что и c (обратите внимание на параллелограмм, задаваемый буквами cgoh) и т.д. Из этого примера читатель легко может

вывести правила шифрования текста, когда две буквы лежат в одной строке или столбце [явно эти правила сформулированы в работе Синкова (Sinkov, 1968, p. 114)]. Используя подсчет частот диграфов, Мауборн провел криптоанализ этой схемы в 1914 г.

Краеугольный камень современной математической криптографии был заложен Хиллом (Hill, 1929, 1931). Хилл выяснил, что почти все существующие криптосистемы могут быть сформулированы в единой модели линейных преобразований пространства сообщений. Он отождествил n-ку сообщения с n-кой целых чисел и приравнял операции шифрования и дешифрования к паре взаимно обратных линейных преобразований. Обобщая понятие модулярного шифра, мы получаем

Определение 4.2.4. Пусть K есть $n \times n$-матрица, а $\mathbf{a}$ и $\mathbf{d}$ — некоторые n-мерные векторы с компонентами из $\mathbb{Z}_m$. *Шифр Хилла* — это отображение вида $\mathbf{a} \to \mathbf{K}\mathbf{a}^T + \mathbf{d}$, инъективное тогда и только тогда, когда $\gcd[\det(\mathbf{K}), m] = 1$; все операции выполняются по модулю m.

Для шифровки мы подразделяем открытый текст на блоки по n букв каждый, заменяем каждую букву соответствующим ей элементом из $\mathbb{Z}_m$, образуем транспонированные вектор-столбцы и применяем данное линейное преобразование к каждому блоку $\mathbf{a}$. В этом контексте используются матрицы *инволюций* $\mathbf{K}$, являющиеся своими обратными. Они определяются условиями $\mathbf{K}^2 = \mathbf{I}$ или $\mathbf{K} = \mathbf{K}^{-1}$. (Операции выполняются по модулю m.) Заметим, что поскольку $\mathbf{K}^2 = \mathbf{I}$, то $\det^2(\mathbf{K}) = 1$, откуда следует, что $\det(\mathbf{K}) = \pm 1$. Пытаясь определить матрицы инволюций, когда размерность n равна 2 и $m = 26$, мы видим, что если $\det(\mathbf{K}) = +1$, то можно получить только восемь таких матриц. Ясно, что случай, когда $\det(\mathbf{K}) = -1$ гораздо интереснее (см. также упр. 5 к этому разделу). В этом случае имеется 736 матриц инволюции, и криптограмма может быть дешифрована, даже если не известно ни одного слова минимальной длины из открытого текста. Определяются и пробуются все матрицы инволюций. Для больших n дешифровка этим методом проб и ошибок уже невозможна.

Пример. Рассмотрим сообщение «computer algebra», где буквам $a, b, \ldots, z$ открытого текста присвоены номера $0, 1, \ldots, 25$, и матрицу инволюции

$$\mathbf{K} = \begin{bmatrix} 2 & 7 \\ 7 & 24 \end{bmatrix};$$

вектор $\mathbf{d}$ полагается нулевым. Шифрованный текст принимает вид «ymzcr yxuze oirza», причем последний символ открытого текста оста-

ется без изменения. Пара «со», эквивалентная вектору $(2, 14)$, отображается на вектор $\mathbf{K}(2, 14)^T = (4+98, 14+336) \equiv (24, 12) \pmod{26}$, который соответствует паре «ym», и т.д.

Заменяя постоянную матрицу $\mathbf{K}$ матрицей с переменными элементами, мы получаем разновидность предложенной выше схемы. Детали этого подхода могут быть найдены в книге Лидла и Пилца (Lidl, Pilz, 1984), другие интересные схемы описаны также Кришнамурти и Рамахандраном (Krishnamurthy, Ramachandran, 1980).

Прежде чем приступить к обсуждению современных криптосистем, нам понадобятся некоторые понятия, связанные с надежностью таких систем. В оценках надежности любой системы необходимо предполагать худший случай, т.е. противник может иметь доступ к другой информации, кроме шифра. Соответственно мы рассматриваем такие случаи:

Анализ только шифрованного текста. При такой атаке на систему противник имеет доступ только к перехваченному шифрованному тексту.

Анализ известного открытого текста. Теперь противник имеет несколько пар открытый текст–шифрованный текст, с которыми он может работать.

Анализ выбранного открытого текста. Противник может передавать системе неограниченные порции открытого текста и может наблюдать соответствующий шифрованный текст (это самая серьезная атака на системы).

Из предыдущего обсуждения видно, что только система одноразового блокнота *безоговорочно надежна*; это означает, что независимо от того, какие вычислительные мощности использует противник, он не может взломать систему. Однако чтобы избежать недостатков одноразового блокнота, достигается компромисс, а именно на практике рассматривается *не безоговорочно надежная* система в предположении, что перехватчик может успешно анализировать шифрованный текст с использованием невероятно мощного компьютера. Основная идея здесь — ограничить мощность противника осуществимыми вычислениями. Для понимания смысла этого утверждения нам понадобятся некоторые основные факты теории сложности вычислений (Lewis, Papadimitriou, 1978).

Математические проблемы могут быть первично разделены на две основные категории: *разрешимые* и *неразрешимые* проблемы. Пример неразрешимой проблемы — проблема «останова» машины Тьюринга, которая эквивалентна следующей: «Цирюльник бреет всех тех, кто

не бреется сам; бреет ли он себя?». (*Ответ*: Если бреет, то не бреет, а если не бреет, то бреет, т.е. у задачи нет ответа.)

Разрешимые проблемы могут быть далее подразделены на следующие общие категории:

Доказуемо трудные проблемы, имеющие лишь экспоненциальное, т.е. вида $O(2^n)$, время счета. Пример такой проблемы — арифметика Пресбургера вещественных чисел. Здесь мы хотим выяснить, верна ли формула, в предположении, что мы не можем перемножать две переменные (а можем только умножать переменную на скаляр) и что мы имеем только квантор $\exists$.

P-проблемы, которые имеют полиномиальное, т.е. вида $O(n^k)$, время счета. Пример такой проблемы — задача о цепях Эйлера из теории графов, когда мы хотим найти путь в графе, проходящий по каждому отрезку один раз.

NP-проблемы (недетерминистические полиномиальные), для которых известны только экспоненциальные алгоритмы, но не доказано, что алгоритмов с полиномиальным временем не существует. Очевидно, что множество P-проблем образует подмножество множества NP-проблем, но вопрос, верно ли, что $P = NP$ — наибольшая открытая проблема в теоретической информатике. Характеристическое свойство NP-проблем таково: тогда как найти решение такой проблемы очень трудно, если оно найдено, то его очень легко проверить за полиномиальное время; это свойство будет использоваться ниже.

NP-полные проблемы, которые образуют маленькое подмножество NP-проблем, характеризуемое свойством, что если какая-либо одна из них решена с помощью эффективного алгоритма, то все остальные проблемы в классе NP могут быть решены эффективно. Пример — задача о гамильтоновых цепях, также из теории графов, когда мы хотим найти путь в графе, проходящий через каждую вершину точно один раз.

Воспользуемся теперь упомянутыми выше идеями современной криптографии (Feistel, 1973; Lempel, 1979). Нас интересуют блочные шифры, действующие на группах битов. Чтобы продемонстрировать, как такие операции выполняются электронным устройством, рассмотрим случай, когда мы имеем только три бита (с помощью трех битов мы может представлять всего восемь объектов). Устройство называется ящиком подстановки или S-ящиком и показано на рис. 4.2.3.

На рис. 4.2.3 мы видим, что устройство подстановки состоит из двух переключателей. Первый преобразует последовательность трех

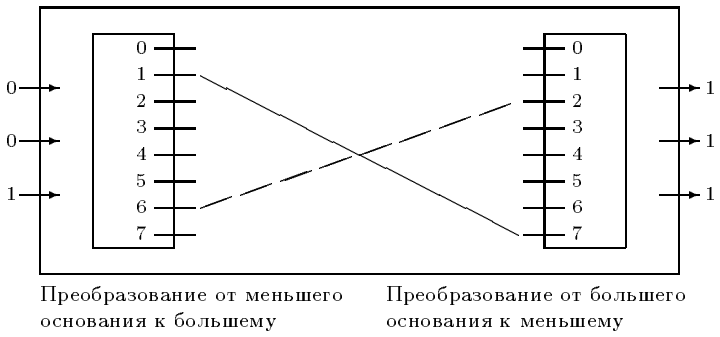


Рис. 4.2.3.

Ящик подстановки (S-ящик) для блочных шифров: соединения между двумя переключателями можно рассматривать как ящик перестановки (P-ящик).

битов в соответствующее ей значение по основанию восемь, подавая, таким образом, энергию на любую из восьми выходных линий. Эти восемь линий могут быть соединены с вторым переключателем любым из $8! = 40\ 320$ способов. Нам предоставляется решить, какую из этих 40 320 перестановок проводов нужно осуществить между первым и вторым переключателями. Роль второго переключателя состоит в том, что он должен преобразовать ввод, представляемый одной цифрой по основанию восемь, снова в 3-битовый выходной сигнал.

Предположим теперь, что мы увеличили число входных битов для S-ящика с трех до пяти, так что мы можем представлять все буквы английского алфавита, т.е. a соответствует 00000, b — 00001 и т.д., и наконец, z — 11001. Тогда существует 32! возможных конфигураций соединения двух переключателей. Однако получающийся шифр все еще остается очень слабым; он не может противостоять анализу частот символов. Проблема состоит в том, что, несмотря на большое количество возможных конфигураций соединения, имеется только 32 возможных входа и выхода. Таким образом, нам необходимо большое количество входов и выходов, настолько большое, что для любого противника практически невозможно разобрать все конфигурации. Например, если мы имеем ящик с 128 входами и выходами, анализатор должен иметь дело с 2^{128} возможными блоками ввода/вывода, т.е. с настолько большим их числом, что частотный анализ больше уже не осуществим. Конечно, недостаток этой схемы в том, что она требует 2^{128} соединений между переключателями (в P-ящике), что технологически невозможно (в настоящее время). Так что должен быть достигнут компромисс.

Ясно, что одно устройство с многими входами и выходами само является P-ящиком; на рис. 4.2.3 он имеет восемь входов и выходов. Несмотря на то что имеется 40 320 возможных соединений, истинное найти очень легко, просто подавая на вход устройства только один бит, равный 1, а остальные равные 0 и наблюдая, где 1 появляется на выходе. В нашем примере мы можем определить соединения после семи проб. Отметим, что S-ящик — это *нелинейное* устройство, а P-ящик — *линейное*.

Чтобы улучшить эту схему, мы должны ввести так называемые *системы шифров-произведений*, комбинирующие P- и S-ящики. Впервые системы шифров-произведений были предложены Шенноном (Shannon, 1948, 1949), они состоят из слоев P- и S-ящиков. Предположим, что P-ящики имеют по 15 входов и выходов и что S-ящики имеют их только по 3; тогда за каждым P-ящиком следует пять S-ящиков, и операция в системе шифров-произведений выполняется при условии, что вход состоит из 14 нулей и одной 1, следующим образом: первый ящик, P, передает *единственную* 1 некоторому ящику S, который, являясь нелинейным устройством, может из одной 1 получить до трех 1. Эти единицы подаются затем на следующий P-ящик, который перетасовывает их и подает на следующие S-ящики, и процесс повторяется. В конце выход содержит сбалансированное число нулей и единиц. Эти идеи проиллюстрированы на рис. 4.2.4.

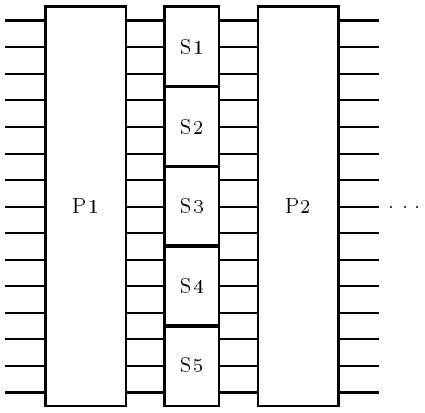


Рис. 4.2.4.

Система шифров-произведений, где P-ящики имеют 15 входов и выходов, а S-ящики имеют их только по 3.

Рис. 4.2.4 иллюстрирует принцип, на котором основана IBM-система LUCIFER. P-ящики в системе LUCIFER имеют или 64, или

128 входов и выходов, а S-ящики — только 4. Конечно, цель разработчика состоит в том, чтобы сделать для противника как можно более сложной задачу проследить обратный путь и таким образом восстановить ключи перестановок на S и P. Система LUCIFER является блочным шифром высокой пробы, однако, в настоящее время она не считается надежной системой.

В 1977 г. Национальное бюро стандартов выпустило Стандарт шифрования данных (DES) с намерением создать криптографический стандарт, используемый для надежной передачи всяких данных, кроме данных, связанных с национальной безопасностью. DES-алгоритм — это блочный шифр, разработанный фирмой IBM, на основе S/P-схемы типа описанной выше системы LUCIFER. DES шифрует 64-битовые блоки открытого текста, используя 64-битовые ключи (56 битов ключа и 8 проверочных битов). Шифрование осуществляется за 16 отдельных этапов, причем на каждом этапе используется шифр-произведение под управлением 48-битового ключа. То есть на всех этапах используются различные 48-битовые ключи K_i, $i = 1, 2, \ldots, 16$, получаемые в соответствии с некоторым алгоритмом программы планирования из внешнего ключа K.

Хотя с точки зрения теории сложности DES выглядит надежным, размер ключа этого стандарта подвергается критике [см. (Diffie, Hellman, 1977)]. Проблема состоит в том, что 56-битовый ключ взламывается путем «анализа известного открытого текста» противником с использованием больших, но реальных, вычислительных ресурсов. Это делается методом грубой силы. Предположим, например, что известное сообщение M зашифровано при помощи DES с ключом K и что $E(K, M) = C$. Чтобы определить K, криптоаналитик декодирует C с каждым из 2^{56} возможных ключей. Определив M, криптоаналитик прекращает работу и объявляет K. Хотя этот исчерпывающий поиск выглядит невозможным, решительный противник может построить компьютер специального назначения, который выполняет параллельно миллион проверок за сравнительно короткое время (< 10 часов). Однако, вероятно, увеличение размера ключа с 56 до 128 исключит исчерпывающий поиск из употребления.

Неудовлетворение стандартом DES дало импульс дальнейшим исследованиям (Diffie, Hellman, 1976) и привело к открытию асимметричных систем кодирования, также известных как *криптосистемы открытого ключа*. Отметим, что до сих пор все рассматриваемые системы были симметричными в том смысле, что как отправитель, так и получатель обладали одним и тем же ключом, который должен быть надежным.

4.2.2. Асимметричные криптосистемы (открытого ключа)

Понятие криптосистем открытого ключа было впервые введено Диффи и Хеллманом в 1976 г. Согласно этим схемам, каждый пользователь помещает в открытый каталог процедуру шифрования E (чтобы другие подписчики использовали ее для кодирования сообщений, адресованных этому пользователю), но держит в секрете детали соответствующей процедуры D дешифровки. [Представляют интерес статьи (Ecker, 1982; Hellman, 1979; Simmons, 1979).] Очевидно, что эти методы основаны на наблюдении, что процедура шифрования и соответствующий ключ не должны быть такими же, как процедура дешифровки и соответствующий ей ключ. Чтобы такая система имела право на существование, должен найтись простой метод, позволяющий получать процедуры E и D друг из друга. Желательно, чтобы процедуры E и D обладали следующими свойствами:

1. Если M — открытый текст сообщения, то E и D должны быть такими, что
$$D[E(M)] = M,$$
т.е. дешифровка зашифрованного текста M дает M.
2. Как E, так и D могут быть легко вычислены.
3. Знание E не приводит к легкому способу вычисления D. [Очень неэффективный способ вычисления D состоит в том, чтобы проверять $E(M) = C$ для всех возможных сообщений M.]
3. Для каждого сообщения M должно быть $E[D(M)] = M$. Это полезно для реализации подписей и будет разъяснено ниже.

Например, если пользователь B хочет послать частное сообщение M пользователю A, то пользователь B находит процедуру E_{A} в открытом каталоге пользователя A (как в телефонном справочнике) и передает $C = E_{\mathrm{A}}(M)$ воткрытую, зная, что только A может декодировать это, пользуясь секретной процедурой D_{A}.

Пользователь B может также «подписать» сообщение, посылаемое пользователю A, так, чтобы A убедился в его подлинности. Чтобы подписать сообщение M, пользователь B сначала вычисляет зависящую от сообщения подпись
$$S = D_{\mathrm{B}}(M),$$
затем он находит в каталоге E_{A} и передает $C = E_{\mathrm{A}}(S)$. Теперь только A может извлечь S из C, вычислив $D_{\mathrm{A}}(C) = S$ и затем использовав E_{B} из каталога, чтобы получить сообщение M, т.е.
$$E_{\mathrm{B}}(S) = E_{\mathrm{B}}[D_{\mathrm{B}}(M)] = M.$$

Поскольку только пользователь B мог создать текст S, который декодируется в M процедурой E_{B}, пользователь A знает, что сообщение M могло быть послано только пользователем B. Криптосистема RSA (названная в честь ее создателей Ривеста, Шамира и Адлемана (Rivest, Shamir, Adleman, 1978), описанная ниже, представляет очень хорошую технику для подписей.

В качестве применения предложенной схемы рассмотрим случай, когда подписан договор между странами A и B о запрещении ядерных испытаний, и для того чтобы контролировать соблюдение договора, достигнуто соглашение об установке на территории противной стороны сейсмических инструментов для фиксации любых отклонений и, следовательно, обнаружения подземных испытаний. В то время как можно защитить сами инструменты (они могут саморазрушаться, если кто-то попытается их вскрыть), невозможно защитить канал, по которому передается информация, поскольку можно перерезать провода и посылать ложную информацию. Кроме того, если сообщения посылаются в шифрованном виде, то страна-хозяин может предполагать, что кроме сейсмических данных передается дополнительная незаконная информация.

Проблемы, стоящие перед двумя странами, могут быть решены путем использования цифровой системы подписи. Сейсмическая станция страны A содержит компьютер, который преобразует сообщение M в $S = D_{\mathrm{A}}(M)$. Страна B не может заменить S каким-нибудь S', поскольку с большой вероятностью сообщение $E_{\mathrm{A}}(S') = M'$ будет бессмысленным. Однако страна A *передает* стране B процедуру E_{A}, которую B может использовать, чтобы получить $E_{\mathrm{A}}(S) = M$ и быть таким образом уверенной, что послано только законное сообщение.

Предоставив полное описание этих элегантных понятий, Диффи и Хеллман не описали практической реализации криптосистемы открытого ключа, указав, однако, что такая реализация может быть выполнена с помощью вычислительно сложной задачи типа обращения односторонних функций.

Функция f называется *односторонней*, если она обратима и ее легко вычислить, но определить f^{-1}, исходя из полного описания функции f, вычислительными средствами невозможно. Функция f называется односторонней *ловушкой*, если f^{-1} легко вычисляется при наличии информации о «ловушке» (а именно секретного ключа дешифровки) и если без этой информации функция f односторонняя.

Первая криптосистема открытого ключа (Merkle, Hellman, 1978) базировалась на *NP-задаче рюкзака или подмножеств*, которая может быть сформулирована следующим образом: даны число N и k

чисел $\{n_1, n_2, \ldots, n_k\}$; найти подмножество в $\{n_1, n_2, \ldots, n_k\}$, сумма которого равна N, если такое подмножество существует.

Пример. Рассмотрим множество {1292, 2089, 2110, 625, 283, 1599, 3759, 1315, 250, 2460} из 10 ($k = 10$) чисел и $N = 8329$. Какие из этих чисел дают в сумме N? В общем, и худшем, случае мы должны сделать 2^k проб; в нашем случае методом проб и ошибок мы находим, что $8329 = 2110 + 3759 + 2460$.

Отметим характеристическое свойство NP-проблем; именно, в то время как трудно найти числа 2110, 2460 и 3759, как только они найдены, мы можем легко проверить (просто сложив их), что они действительно решают нашу проблему. Более того, проблема рюкзака также NP-полна. (Очевидно, что при стремлении k к бесконечности в приведенном выше примере невозможно решить проблему рюкзака методом поиска.)

В общем случае, если мы определим *скалярное* произведение любых двух n-мерных векторов $\mathbf{n} = (n_1, n_2, \ldots, n_k)$ и $\mathbf{x} = (x_1, x_2, \ldots, x_k)$ как $\mathbf{n} \cdot \mathbf{x} = n_1x_1 + n_2x_2 + \cdots + n_kx_k$, то проблема рюкзака может быть представлена в виде $N = \mathbf{n} \cdot x$, где $x_i = 1$, если i-я компонента вектора $\mathbf{n}$, n_i, входит в сумму, и $x_i = 0$ в противном случае.

Рассмотрим теперь частный случай проблемы рюкзака, где данные числа образуют *супервозрастающую последовательность*, т.е. каждое число n_i больше, чем сумма всех предшествующих элементов. В этом случае мы имеем простую проблему рюкзака, поскольку за линейное время можно определить, имеется ли решение, и если оно существует, то простой алгоритм найдет его.

Пример. Рассмотрим множество {5, 10, 20, 42, 90, 205, 500} из семи чисел ($k = 7$) и $N = 305$. Решить эту проблему рюкзака — все равно, что найти решение $x_i = 0$ или 1, $i = 1, 2, \ldots, 7$, уравнения

$$305 = 5x_1 + 10x_2 + 20x_3 + 42x_4 + 90x_5 + 205x_6 + 500x_7.$$

Это, однако, тривиально, поскольку легко видеть, что $x_7 = 0$ и $x_6 = 1$. (Поскольку сумма всех предшествующих чисел меньше 205, x_6 должен равняться 1, в противном случае мы никогда не получим сумму 305.) Проблема принимает вид

$$100 = 5x_1 + 10x_2 + 20x_3 + 42x_4 + 90x_5,$$

и продолжая аналогичным образом, мы получаем $x_5 = 1$, $x_4 = x_3 = 0$, $x_2 = 1$ и $x_1 = 0$. Другими словами, нам требуется в общем случае только k вычитаний и сравнений, и, следовательно, эта версия проблемы рюкзака может быть решена за полиномиальное время (линейное).

Таким образом, основная идея использования рюкзака для криптосистем состоит в преобразовании простого рюкзака в сложный — рюкзак с ловушкой. Это осуществляется путем выбора двух достаточно больших целых чисел m и w, $(m, w) = 1$ и образования затем нового рюкзака с ловушкой $\mathbf{n} = (n_1, n_2, \ldots, n_k)$ из данного супервозрастающего вектора рюкзака $\mathbf{n}^s = (n_1^s, n_2^s, \ldots, n_k^s)$ с помощью формулы $n_i \equiv w n_i^s \pmod{m}$, $i = 1, 2, \ldots, k$. Модулярная арифметика перетасовывает числа, и, следовательно, значения n_i^s не образуют больше супервозрастающей последовательности. Вектор рюкзака $\mathbf{n}$ публикуется и образует открытый ключ, а вектор $\mathbf{n}^s$ вместе с числами m, w и w^{-1}, мультипликативным обратным к $w \pmod{m}$, *держатся в секрете* человеком, ожидающим прихода сообщения.

Подведем итоги:

Получатель знает $\mathbf{n}^s$, m, w и w^{-1}, образующие секретный ключ, отправитель знает сообщение M, каждый знает открытый ключ $\mathbf{n}$.

Если $M = (x_1, x_2, \ldots, x_{bk})$ — сообщение в двоичном виде, то оно зашифровывается b величинами $c_i = n_1 \cdot x_{(i-1)k+1} + n_2 \cdot x_{(i-1)k+2} + \cdots + n_k \cdot x_{ik}$, которые и передаются. Перед несанкционированным перехватчиком стоит задача решения сложной проблемы рюкзака с суммами c_i и вектором $\mathbf{n}$. Однако, умножая c_i на w^{-1}, предполагаемый получатель легко вычисляет $c_i^s \equiv w^{-1} c_i \pmod{m}$, $i = 1, 2, \ldots, b$, и преобразует трудные проблемы рюкзака с суммами c_i и вектором $\mathbf{n}$ в простые рюкзаки с суммами c_i^s и вектором $\mathbf{n}^s$.

Неизвестно, насколько надежными являются системы, основанные на проблеме рюкзака. Надежность этих криптосистем базируется на анализе времени вычислений худшего случая проблемы рюкзака. Например, если $k = 1000$, то понадобится самое большее 2^{1000} проб для решения проблемы. Однако может также случиться, что решение будет найдено после всего нескольких проб, в случае чего система не будет так надежна, как мы думаем. Первоначально считалось, что повторная шифровка увеличивает надежность рюкзака с ловушкой. Однако в настоящее время известны атаки на простые и дважды итерированные рюкзаки. (Чтобы получить дважды итерированный рюкзак, выбираем другую пару чисел m_1, w_1 и получаем из $\mathbf{n}$ новый вектор; процесс может быть повторен сколько угодно раз.) Представляет интерес статья Меркля и Хеллмана (Merkle, Hellman, 1981).

Операции этого типа для криптосистем разъяснены ниже. Однако следует помнить, что выбранные значения слишком малы, чтобы

обеспечить какую-либо надежность, т.е. они слишком малы, чтобы убедить, что вычислительная часть недоступна для несанкционированного перехватчика.

Пример. Выберем следующее соответствие между буквами алфавита и двоичными 5-наборами:

a	00000	b	00001	c	00010	d	00011	e	00100	f	00101
g	00110	h	00111	i	01000	j	01001	k	01010	l	01011
m	01100	n	01101	o	01110	p	01111	q	10000	r	10001
s	10010	t	10011	u	10100	v	10101	w	10110	x	10111
y	11000	z	11001	,	11010	.	11011	!	11100	?	11101
$''$	11110	+	11111								

где + обозначает пробел. Мы выберем также секретную информацию $\mathbf{n}^s = (1, 7, 13, 28, 52), m = 111$ и $w = 55$, в случае чего $w^{-1} \pmod{111} = 109$; тогда открытый ключ — это $\mathbf{n} = (55, 52, 49, 97, 85)$, где $n_i \equiv n_i^s w \pmod{m}$. Теперь сообщение «computer algebra», представляемое двоичной последовательностью 00010 01110 01100 01111 10100 10011 00100 10001 00000 01011 00110 00100 00001 10001 00000 ($= x_1 x_2 \ldots x_{75}$), передается как последовательность чисел 97, 198, 101, 283, 104, 237, 49, 140, 0, 234, 146, 49, 85, 140, 0, где $97 = 55 \cdot 0 + 52 \cdot 0 + 49 \cdot 0 + 97 \cdot 1 + 85 \cdot 0$ и т.д. Отметим, что в общем случае каждое передаваемое число равно $c_i = n_1 \cdot x_{(i-1)k+1} + n_2 \cdot x_{(i-1)k+2} + \cdots + n_k \cdot x_{ik}$, т.е. первоначальное сообщение в двоичном виде разбивается на группы из k битов. Несанкционированный перехватчик знает c_i и открытый ключ $\mathbf{n}$, но для больших k он должен перепробовать огромное число случаев, чтобы расшифровать сообщение (2^k в худшем случае).

Получатель, обладающий супервозрастающим вектором $\mathbf{n}^s$, должен просто вычислить

$$c_i^s \equiv c_i w^{-1} \pmod{m}$$

и решить легкую задачу рюкзака $c_i^s = \mathbf{n}^s \cdot \mathbf{x}$. Это так, потому что если выбрать $m > \sum n_i^s$, то

$$\begin{aligned}
c_i^s &\equiv [n_1 \cdot x_{(i-1)k+1} + n_2 \cdot x_{(i-1)k+2} + \cdots + n_k \cdot x_{ik}] w^{-1} \pmod{m} \\
&\equiv w^{-1} n_1 \cdot x_{(i-1)k+1} + w^{-1} n_2 \cdot x_{(i-1)k+2} + \cdots + w^{-1} n_k \cdot x_{ik} \pmod{m} \\
&\equiv w^{-1} w n_1^s \cdot x_{(i-1)k+1} + w^{-1} w n_2^s \cdot x_{(i-1)k+2} + \cdots + w^{-1} w n_k^s \cdot x_{ik} \pmod{m} \\
&\equiv n_1^s \cdot x_{(i-1)k+1} + n_2^s \cdot x_{(i-1)k+2} + \cdots + n_k^s \cdot x_{ik} \pmod{m} \\
&= n_1^s \cdot x_{(i-1)k+1} + n_2^s \cdot x_{(i-1)k+2} + \cdots + n_k^s \cdot x_{ik}.
\end{aligned}$$

Так, продолжая наш пример, мы сначала заметим, что в нашем случае $m > \sum n_i^s$ и получатель вычисляет 15 величин c_i^s, которые равны 28, 48, 20, 100, 14, 81, 13, 53, 0, 87, 41, 13, 52, 53, 0. Затем он без труда решает соответствующие проблемы рюкзака с вектором $\mathbf{n}^s = (1, 7, 13, 28, 52)$ и получает переданное сообщение; например, 28 соответствует двоичному 5-набору 00010 $(= c)$, 48 соответствует 01110 $(= o)$, 20 соответствует 01100 $(= m)$ и т.д.

Существует вариант описанной криптосистемы, базирующийся на *мультипликативном рюкзаке*. Здесь дается число N и k взаимно простых чисел $\{n_1, n_2, \ldots, n_k\}$, а мы хотим найти, если оно существует, подмножество множества $\{n_1, n_2, \ldots, n_k\}$, такое, что *произведение* этих чисел равно N.

Завершая обсуждение криптосистем рюкзака, следует заметить, что они не обладают свойством непосредственной цифровой подписи. Это связано с тем, что не всякое целое число c в области значений криптограмм может быть представлено в виде суммы некоторого подмножества чисел n_i, т.е. отображение не является отображением «на». Подробности можно найти в литературе.

Продолжим теперь обсуждение RSA-криптосистем, основанных на степенной функции, используемой как односторонняя функция. Успех этого метода зависит от сложности нахождения сомножителей больших целых чисел. Если бы эту проблему удалось решить за полиномиальное время, RSA-криптосистемы были бы скомпрометированы.

Для лучшего понимания RSA-криптосистем читателю следует еще раз просмотреть разд. 2.3; в частности, полезными являются определение функции ϕ в разд. 2.3.1 и теорема Эйлера 2.3.14, утверждающая, что если $(a, m) = 1$, то $a^{\phi(m)} \equiv 1 \pmod{m}$. Кроме того, в разд. 2.3.1 мы видели, что $\phi(n) = n(1 - 1/p_1)(1 - 1/p_2)\ldots(1 - 1/p_k)$ для любого целого n, где p_i — простые числа, входящие в разложение числа n в произведение степеней простых чисел. Для нас представляет интерес частный случай $n = pq$, где p и q — два различных простых числа. Из последней формулы мы получаем $\phi(pq) = (p - 1)(q - 1)$. Покажем, как работает RSA-криптосистема. Получатель вычисляет следующие величины:

p, q: Выбраны два больших простых числа, которые держатся в *секрете*.

n: Произведение pq, которое помещается в открытый каталог.

E: Случайное целое число, также размещаемое в открытом каталоге. [E используется для кодирования сообщений, посылаемых

получателю, который должен убедиться, что E взаимно просто с числом $\phi(n) = (p-1)(q-1)$. Это сделать легко, поскольку получатель знает числа p и q, а gcd вычисляется быстро.]

D: Целое число, используемое получателем для декодирования. D — мультипликативное обратное числа E по модулю $(p-1)(q-1)$ — также держится *в секрете*; т.е. $DE \equiv 1 \pmod{(p-1)(q-1)}$. Обратное существует, поскольку $\gcd[E,(p-1)(q-1)] = 1$. Снова эти вычисления быстрые, поскольку получатель знает p и q.

Причины, накладывающие технические ограничения на различные целые числа, будут разъяснены ниже.

Подведем итоги.

Получатель знает p, q и D, образующие секретный ключ, отправитель знает сообщение M, каждый знает открытый ключ n и E.

Сообщение M, передаваемое получателю, прежде всего преобразуется в цифровую форму некоторой стандартной несекретной процедурой. В частности, можно использовать присвоения $a = 00$, $b = 01$, $c = 02$, $d = 03$, $e = 04$, ..., $z = 25$, . $= 26$, , $= 27$, ? $= 28$, ; $= 29$; т.е. каждая буква заменяется двузначным числом. Например, сообщение «computer algebra» должно быть переписано в виде «02141215201904170011 0604011700». Конечно, если сообщение длинное, то оно разбивается на блоки; длина каждого блока такова, что его численное значение *не превосходит* n. Важно, чтобы каждый блок M_i сообщения находился в области $0 \le M_i \le n-1$, поскольку в противном случае невозможно отличить его от любого большего целого числа, сравнимого с ним по модулю n. Практически модуль n выбирается большим, порядка 200 цифр, так что могут использоваться блоки размером до 10^{200}.

Отправитель берет каждый блок сообщения M_i, смотрит на открытые ключи E и n и передает $C_i \equiv M_i^E \pmod{n}$, где $0 \le M_i \le n-1$.

Получатель получает C_i и вычисляет $(C_i)^D \pmod{n}$. Заметим, однако, что по определению мы имеем $ED \equiv 1 \pmod{\phi(n)}$, откуда следует, что $ED = 1 + k\phi(n)$ для некоторого целого k. Таким образом,

$$\begin{aligned}(C_i)^D &\equiv (M_i)^{ED} \\ &= (M_i)^{[1+k\phi(n)]} \\ &\equiv M_i \pmod{n},\end{aligned}$$

где последнее равенство получено из теоремы Эйлера 2.3.14. Таким образом, получатель определил блок первоначального сообщения M_i.

Операции криптосистемы такого типа описаны ниже. Однако, как и в случае криптосистемы рюкзака, следует иметь в виду, что выбранные значения слишком малы, чтобы обеспечить какую-либо надежность, т.е. они слишком малы, чтобы гарантировать, что часть вычислений невыполнима для несанкционированного перехватчика.

Пример. Как упоминалось выше, используя присвоения $a = 00$, $b = 01$, $c = 02$, $d = 03$, $e = 04$, ..., $z = 25$, $. = 26$, $,= 27$, $? = 28$, $;= 29$, сообщение «computer algebra» мы переписываем в виде «0214121520 1904170011 0604011700». Выбирая $p = 3$, $q = 11$, мы имеем $n = 3 \cdot 11 = 33$ и $\phi(n) = 20$; более того, мы выбираем $E = 7$, и в этом случае $D = 3$. Теперь каждый блок сообщения M_i состоит из двузначного числа, и зашифрованное сообщение получается по формуле $C_i \equiv (M_i)^7 \pmod{33}$. А именно, мы получаем последовательность чисел $29, 20, 12, 27, 26,\ \ 13, 16, 08, 00, 11,\ \ 30, 16, 01, 08, 00$, которая и передается. Для того чтобы восстановить блок переданного сообщения (который в нашем случае представляет просто букву), получатель должен возвести каждое двузначное число в куб (modulo 33); это мы оставляем читателю в качестве упражнения. (В разд. 2.3.2 мы рассматривали эффективные алгоритмы для возведения целых чисел в степень, а также как для вычисления мультипликативного обратного целого числа по модулю другого целого числа; читателю следует еще раз просмотреть эти алгоритмы.)

Несанкционированный перехватчик сообщения в рассмотренном примере должен вычислить D, мультипликативное обратное числа E по модулю $(p-1)(q-1)$. Однако, чтобы сделать это, перехватчик должен найти p и q по $n = pq$, находящемуся в открытом каталоге. Таким образом ему требуется алгоритм с полиномиальным временем для разложения больших целых чисел. Отметим, что p и q выбираются так, чтобы у них было одинаковое количество цифр, поэтому у n цифр вдвое больше. Более того, они должны быть выбраны так, чтобы у $p-1$ был достаточно большой сомножитель, обозначим его f, и у $f-1$ также должен быть достаточно большой сомножитель. Аналогичные ограничения накладываются на q. Это гарантирует, что открытый текст не может быть найден с помощью итераций шифрования быстрее, чем случайным поиском; см. (Blakley et al., 1979; Herlestam, 1978; Williams et al., 1979). (Дешифрование *итерациями* — это процесс, когда шифрованный текст C последовательно шифруется снова, до тех пор, пока не получим вновь C, т.е. полагаем $C_1 = C$ и вычисляем $C_{i+1} \equiv (C_i)^E \pmod{n}$, пока не получим $C_{i+1} = C$. Тогда $C_i = M$.)

Другой способ, которым перехватчик может пытаться решить проблему, состоит в нахождении $\phi(n)$, поскольку в этом случае D может быть легко вычислено из соотношения $ED \equiv 1 \pmod{\phi(n)}$. Однако следующие соображения показывают, что этот подход не проще, чем попытки разложить n; если известно $\phi(n)$, то p и q могут быть найдены следующим образом. Из соотношений $\phi(n) = (p-1)(q-1) = pq - (p+q) + 1 = n - (p+q) + 1$ видно, что по $\phi(n)$ легко находится $p+q$. Более того, соотношения $(p-q)2 = (p+q)2 - 4pq = (p+q)2 - 4n$ показывают, что, зная n и $p+q$, легко получить $p-q$. Тогда p и q даются формулами

$$p = (1/2)[(p+q) + (p-q)],$$
$$q = (1/2)[(p+q) - (p-q)].$$

Таким образом, любая попытка найти $\phi(n)$ эквивалентна решению сложной проблемы разложения n на множители.

Предположим, что современные алгоритмы разложения на множители могут разлагать целые числа до 200 десятичных цифр за несколько часов на самых быстрых машинах; тогда если n имеет примерно 2000 десятичных цифр, то можно с уверенностью утверждать, что его невозможно разложить на множители за какое-либо разумное время. Однако, по-видимому, слишком рано говорить, выдержат ли RSA-криптосистемы проверку временем.

Существует также криптосистема с открытым ключом, основанная на сложности общей проблемы декодирования для линейных кодов, исправляющих ошибки; более подробно см. (Lempel, 1979, pp. 297–298). Более того, существуют распределенные системы с открытым ключом. Назначение такой системы состоит в том, чтобы позволить каждой паре пользователей надежно обменяться собственным ключом на ненадежном канале для использования в стандартной криптосистеме; см. (Merkle, 1978; Pohlig et al., 1978).

Упражнения

Раздел 4.1

1. Докажите, что в векторном пространстве размерности n над $GF(2)$ расстояние Хэмминга — это метрика, а вес Хэмминга — норма.

2. Завершите доказательство лемм 4.1.5 и 4.1.9.

3. Покажите, что если кодовое расстояние $\geq r+t+1$, то этот код может исправлять $\leq r$ ошибок и обнаруживать $r+t$ ошибок.

4. Постройте код, состоящий из восьми слов длины 7, такой, что расстояние между любыми двумя различными кодовыми словами не меньше 4.

Раздел 4.1.1

1. Пусть

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

— порождающая матрица двоичного (5,2)-кода. Определите его проверочную матрицу, кодовые слова и лидеры смежных классов для этого кода.

2. При помощи (7,4)-кода Хэмминга с проверочной матрицей

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

закодируйте сообщения $(0,1,1,0)$ и $(1,0,1,0)$. Кроме того, предполагая, что случилось не более одной ошибки, декодируйте векторы (**a**) $\mathbf{v} = (1,1,0,0,0,0,0)$, (**b**) $\mathbf{v} = (1,0,0,1,0,1,0)$ и (**c**) $\mathbf{v} = (1,1,0,1,0,1,1)$, которые были получены по зашумленному каналу.

3. Модификация проверочной матрицы $\mathbf{H}$ из упр. 2 позволяет нам обнаруживать наличие двух ошибок, а также исправлять одну ошибку. А именно, добавляя столбец нулей в начале (слева) и заполняя верхнюю строку единицами, мы получаем

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

Теперь если $\mathbf{v}$ — полученный вектор, $\mathbf{u}$ — переданный и $\mathbf{e} = \mathbf{v} - \mathbf{u}$, то:

a. Если все координаты $\mathbf{e}$ равны нулю, то при передаче не было ошибок.

b. Если $\mathbf{e}$ имеет одну ненулевую координату, то встретилась одна ошибка, и она находится так же, как в упр. 2.

c. Если $\mathbf{e}$ имеет два ненулевых элемента, то случились две ошибки.

В последнем случае $\mathbf{vH}^T = \mathbf{eH}^T$ — сумма двух столбцов матрицы $\mathbf{H}$, но определить, какие два столбца матрицы $\mathbf{H}$ дают эту сумму, нельзя. Можно, однако, утверждать, что $\mathbf{vH}^T$ не является столбцом матрицы $\mathbf{H}$, поскольку сумма двух столбцов матрицы $\mathbf{H}$ всегда имеет верхний элемент $= 0$.

Декодируйте векторы

$$\mathbf{v} = (1,0,1,1,0,1,1,1),$$
$$\mathbf{v} = (0,0,0,1,1,1,1,1),$$
$$\mathbf{v} = (1,0,1,1,1,0,0,1),$$
$$\mathbf{v} = (1,0,1,1,0,1,0,1),$$

предполагая, что в каждом из них имеются 0, 1 или 2 ошибки.

Существуют ли в (8,4)-коде Хэмминга полученные слова, для которых получатель может с уверенностью утверждать, что они имеют по крайней мере 3 ошибки?

4. Покажите, что $(2^m - 1, 2^m - 1 - m)$-код Хэмминга совершенен.

5. Двоичные коды можно обобщить на коды, которые являются векторными пространствами над любым конечным полем $GF(q)$. Пусть

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 2 & 1 \end{bmatrix}$$

— порождающая матрица троичного (4,2)-кода C. Найдите его проверочную матрицу.

Раздел 4.1.2

1. Определите все кодовые слова кода с порождающим полиномом $g(x) = x^3 + x + 1$ над $GF(2)$, если длина сообщений равна 4. Имеет ли $(0,1,1,1,0,1,1)$ обнаруживаемые ошибки?

2. Полином $g(x) = x^6 + x^5 + x^4 + x^3 + 1$ является порождающим полиномом циклического кода над $GF(2)$ с блоками длины 15. Найдите проверочный полином и порождающую и проверочную матрицы этого кода. Сколько ошибок может исправлять этот код?

3. Проверьте, что двоичный полином $x^4 + x^3 + x^2 + x + 1$ неприводим, и, используя это, постройте матрицу $\mathbf{H}$ для БЧХ-кода,

исправляющего две ошибки, с блоками длины 15, имеющими 7 информационных и 8 проверочных битов.

4. Рассмотрим циклический (15,7)-код с $g(x) = x^8+x^7+x^6+x^4+1$. Является ли $x^{14} + x^5 + x + 1$ кодовым словом, и если нет, то каков его синдром?

5. Покажите, что схема, изображенная на рис. 4.1.5, действительно производит кодовые слова кода, для которого $h(x)$ является проверочным полиномом. Постройте такую схему кодирования для частного случая двоичного (7,4)-кода Хэмминга с $g(x) = x^3 + x + 1$.

6. [Кодирование циклическим (n, k)-кодом, использующим регистры с $n - k$ разрядами.] Пусть $f_0(x)$ — полином степени n, коэффициенты которого при k наивысших степенях x, а именно $x^{n-1}, x^{n-2}, \ldots, x^{n-k}$, являются k информационными битами, а остальные коэффициенты равны 0. Если $g(x)$ — полином степени $n - k$, то по алгоритму деления получаем $f_0(x) = g(x)q(x) + r(x)$, $\deg[r(x)] < n - k$. Тогда $f_0(x) - r(x)$ — полином степени $\leq n - 1$, делящийся на $g(x)$, который можно рассматривать как кодовое слово циклического (n, k)-кода, порожденного полиномом $g(x)$. Спроектируйте схему кодирования, использующую регистр с $n - k$ разрядами, основанную на приведенных рассуждениях.

7. Спроектируйте схему для обнаружения ошибок циклического кода.

Раздел 4.2.1

1. Докажите, что отображение $a \to an + k \pmod m$ из $\mathbb{Z}_m$ в себя инъективно тогда и только тогда, когда $\gcd(m, n) = 1$.

2. Используйте «computer» как ключ в коде Виженера и зашифруйте слово «cryptography».

3. Получите случайную последовательность 0/1 подбрасыванием монетки и используйте эту последовательность в одноразовом блокноте, чтобы зашифровать сообщение «secret code».

4. Используя данный в тексте квадрат Плайфера, зашифруйте слово «university».

5. Покажите, что если $\det(\mathbf{M}) \equiv -1 \pmod{26}$, то $\mathbf{M}$ — матрица инволюции тогда и только тогда, когда $m_{11}+m_{22} \equiv 0 \pmod{26}$. Постройте матрицу инволюции, у которой $m_{11} = 2$.

6. Пусть $\mathbb{Z}_{26}$ — алфавит открытого и шифрованного текстов в

шифре Хилла. Пусть

$$\mathbf{K} = \begin{bmatrix} 2 & 1 \\ -3 & -2 \end{bmatrix}$$

— матрица шифрования. Зашифруйте слово «department». Чему равна обратная к $\mathbf{K}$ матрица?

7. В 26-буквенном алфавите, описанном в тексте, воспользуйтесь модулярным шифром с $n = 7$ и $k = 19$, чтобы зашифровать сообщение «congratulations».

8. Пользуясь таблицей относительных частот на рис. 4.2.2, дешифруйте следующее сообщение, про которое известно, что оно зашифровано с использованием модулярного шифра при $n = 1$ (известного также как *сдвиг*):

ymjkw jvzjs hdrjy mtisj jixqt slhnu mjwyj cyxyt btwp.

9. Предположим, что мы все еще работаем с 26-буквенным алфавитом, но теперь $n \neq 1$ в модулярном шифре. Более того, наиболее часто в шифрованном тексте встречается буква «j», а вторая по частоте — буква «e». Пользуясь информацией из рис. 4.2.2, что в английском языке «e» и «t» — две наиболее часто встречающиеся буквы, выведите формулу для дешифровки сообщения. (*Указание.* Если p — символ открытого текста, а c — символ шифрованного текста, то вам нужно найти n', k' такие, что $p \equiv c \cdot n' + k' \pmod{m}$; n' — это мультипликативный обратный к $n \pmod{m}$, а $k' = -n \cdot k$, где n, k — параметры модулярного шифра.)

10. **a.** Сколько модулярных шифров имеется в n-буквенном алфавите в случае $n = 1$?

b. В общем случае сколько модулярных шифров имеется в m-буквенном алфавите?

c. Сколько модулярных шифров имеется, когда $m = 24, 26, 29, 32$?

11. Предположим, что мы используем модулярный шифр ($n \neq 1$) в m-буквенном алфавите. Буква называется *неподвижной*, если $a \to a \cdot n + k \equiv a \pmod{m}$.

a. Покажите, что если m — простое число, то всегда существует неподвижная буква.

b. Покажите, что если в нашем модулярном шифре $k = 0$, то (для любого m) имеется по крайней мере одна неподвижная буква, более того, если m четно (и $k = 0$), то имеются по крайней мере две неподвижные буквы.

Раздел 4.2.2

1. Пользуясь криптосистемой рюкзака, зашифруйте слово «university». Используйте те же присвоения букв и те же параметры, что и в соответствующем примере в тексте.
2. Завершите пример, данный в тексте, для RSA-криптосистемы.
3. В упр. 19 разд. 2.3.3 мы видели, что если мы найдем основание b, такое, что n псевдопросто, но *не* строго псевдопросто по основанию b, то мы можем быстро найти нетривиальный сомножитель числа n. Объясните, как бороться с этим при выборе $n = pq$ в RSA-криптосистеме.
4. Для каждой из следующих задач рюкзака определите, является ли последовательность супервозрастающей и сколько решений она имеет (если имеет вообще).
   a. $\{2, 3, 6, 17, 33, 67\}$, $N = 41$.
   b. $\{1, 3, 5, 11, 23, 47\}$, $N = 69$.
   c. $\{2, 3, 6, 12, 22, 49\}$, $N = 61$.
5. Покажите, что супервозрастающая последовательность с наименьшими значениями n_i — это последовательность $n_i = 2^i$. Покажите затем, что соответствующая задача рюкзака $\mathbf{N} = \mathbf{n^S} \cdot \mathbf{x}$ всегда имеет решение n, а именно $n = N$, и что нет никакой другой супервозрастающей последовательности, для которой соответствующая задача рюкзака всегда имеет решение.
6. Покажите, что любая последовательность положительных целых чисел $\{n_i\}$, в которой $n_{i+1} \geq 2n_i$ для всех i, является супервозрастающей.
7. (Бросание монеты на большом расстоянии с использованием двулистной односторонней функции.) Предположим, что при подготовке международного футбольного матча представители двух команд решили *без проведения личной встречи* «бросить монетку», чтобы определить, в какой стране провести игру.

   Под двулистной односторонней функцией мы подразумеваем алгоритм, который:

   a. по данному ключу E подходящего типа строит функцию $f : P \to C$, такую, что для каждого элемента $c \in C$ существует ровно два элемента $p_1, p_2 \in P$, таких, что $f(p_i) = c$ и
   b. по данному ключу D («обратному» к E) и $c \in C$ можно найти оба значения $p_1, p_2 \in P$, такие, что $f(p_i) = c$.

Мы предполагаем, что, зная только E, вычислительно невозможно найти D (см. ниже упр. 8). Отсюда следует, что для данного $p_1 \in P$ можно найти «сопутствующий» элемент p_2 такой, что $f(p_1) = f(p_2) = c$, если известны и E, и D.

Предположим, что представители r_B и r_G бразильской и немецкой национальных футбольных команд хотят воспользоваться этим набором, чтобы бросить монетку. Представитель r_B генерирует пару ключей E и D и посылает E (но *не* D) представителю r_G. Разработайте «приличную» процедуру, т.е. такую, где каждый игрок имеет шансы 50%–50% «выиграть», и которая адекватно защищена против мошенничества. (Вам нужно определить, что значит «выиграть».)

8. В упр. 23 разд. 3.3.1 был представлен алгоритм решения сравнения $x^2 \equiv a \pmod{p}$ для любого простого числа p и квадратичного вычета a. (См. также упр. 1 по программированию разд. 3.3.1.) Предположим, однако, что нет хорошего алгоритма решения сравнения $x^2 \equiv a \pmod{n}$, где a — квадратичный вычет по модулю n, а $n = pq$, произведение двух различных больших простых чисел, если нам не известно разложение n на множители (если оно известно, то для нахождения квадратного корня по модулю n по данным квадратным корням по модулям p и q используется греко-китайский алгоритм). Предположим, что не оба числа p и q сравнимы с по модулю 4, и (как в упр. 7) пусть $E = n$ и $D = [p, q]$, и пусть $P = C$ — множество пар $(x, -x)$ вычетов по модулю n. Рассмотрите функцию $f : x \to x^2 \pmod{n}$ и покажите, что таким образом получается пример из упр. 7 реализации бросания монеты на большом расстоянии.

9. (RSA-криптосистема не всегда скрывает сообщения.) Пусть n — произвольное целое число, являющееся произведением простых (т.е. n свободно от квадратов). Пусть d и e — положительные целые числа, такие, что $(p-1)|(de-1)$ для любого простого p, такого, что $p|n$. (Например, пусть $de \equiv 1 \pmod{\phi(n)}$.) Покажите, что $a^{de} \equiv a \pmod{n}$ для любого целого a (имеет ли оно общий сомножитель с n или нет).

Упражнения по программированию

Раздел 4.1

1. Простой код для передачи числовых данных $x_1x_2\ldots x_n$, $0 \le x_i \le 9$ для всех i, получается прибавлением одной проверочной цифры d_c, состоящей из числа единиц суммы

$$x_1 + 2x_2 + x_3 + 2x_4 + \cdots + X_n, \text{ где } X_n = \begin{cases} x_n, & \text{если } n \text{ нечетно,} \\ 2x_n, & \text{если } n \text{ четно.} \end{cases}$$

Тогда кодовое слово равно $x_1x_2\ldots x_nd_c$. Этот код может обнаруживать многие ошибки перестановок, типичные ошибки, получающиеся при копировании данных вручную.

a. Если полученное слово равно 12347, то определите, имеет ли место ошибка.

b. Какой код получится для цифр 13579? Чему будет равно полученное слово, если цифры 7 и 9 будут переставлены?

c. Используя предложенную схему, напишите программу, обнаруживающую ошибки в полученных строках, содержащих до 10 цифр.

Раздел 4.1.1

1. Дана проверочная $(r \times n)$-матрица $\mathbf{H}$ для (n, k)-кода с исправлением одной ошибки с $r \le 3$ и $n \le 2^r - 1$. Напишите процедуры, чтобы:

a. Распечатывать множество двоичных k-векторов $\mathbf{H}$-кода и кодовое слово для каждого вектора.

b. Декодировать получаемые двоичные n-векторы.

Раздел 4.2.2

1. Напишите процедуру для шифровки/дешифровки текста, используя криптосистему с открытым ключом типа рюкзака.

2. Напишите процедуру для шифровки/дешифровки текста, используя RSA-криптосистему с открытым ключом.

[Отметим, что в некоторых случаях эта схема может оставлять сообщения без изменений. Чтобы убедиться в этом, положим $n = 15$, $E = 3$. Сколько существует x, таких, что $0 \le x \le n - 1$ и $x^E \equiv x \pmod{n}$? Что можно сказать в общем случае?]

Литература

Afrati F. *Introduction to the theory of information* (in Greek). National Technical University of Athens, Greece, 1985.

Berlekamp E.R. *Algebraic coding theory.* McGraw-Hill, New York, 1968. [Имеется перевод: Берлекэмп Э. Алгебраическая теория кодирования. М.: Мир, 1974.]

Blake I.F. Codes and designs. *Mathematics Magazine* **52**, 81–95, 1979.

Blakley G.R., Borosh I. Rivest-Shamir-Adleman public key cryptosystems do not always conceal messages. *Computers and Mathematics with Applications* **5**, 169–178, 1979.

Bose R.C., Ray-Chaudhuri D.K. On a class of error correcting binary group codes. *Information and Control* **3**, 68–79, 279–290, 1960.

Childs L. *A concrete introduction to higher algebra.* Springer-Verlag, New York, 1979.

Diffie W., Hellman M.E. New directions in cryptography. *IEEE transactions on Information Theory* **IT-22**, 644–654, 1976.

Diffie W., Hellman M.E. Exhaustive cryptanalysis of the NBS data encryption standard. *Computer* **10**, 74–84, June 1977.

Ecker A. Ueber die mathematis einiger Chiffrierverfahren. *Computing* **29**, 277–287, 1982.

Feistel H. Cryptography and computer privacy. *Scientific American* **228**, 15–23, May 1973.

Gass F. Solving a Jules Verne cryptogram. *Mathematics Magazine* **59**, 3–11, 1986.

Hamming R.W. Error detecting and error correcting codes. *Bell System Technical Journal* **29**, 147–150, 1950.

Hellman M.E. The mathematics of public-key cryptography. *Scintific American* **241**, 130–139, August 1979.

Herlestam T. Critical remarks on some public-key cryptosystems. *BIT* **18**, 493–496, 1978.

Hill L.S. Cryptography in an algebraic alphabet. *American Mathematical Monthly* **36**, 306–312, 1929.

Hill L.S. Concerning certain linear transformation apparatus of cryptography. *American Mathematical Monthly* **38**, 135-154, 1931.

Hocquenghem A. Codes correcteurs d'erreurs. *Chiffres* **2**, 147–156, 1959.

Kahn D. *The Codebreakers.* MacMillan, New York, 1967.

Krishnamurthy E.V., Ramachandran V. A cryptographic system based on finite field transforms. *Proceedings of the Indian Academy of Sciences (Math. Sci.)* **89**, 75–93, 1980.

Lempel A. Cryptology in transition. *ACM Computing Surveys* **11**, 280–303, 1979.

Levinston N. Coding theory: a counter to G.H. Hardy's conception of applied mathematics. *American Mathematical Monthly* **77**, 249–258, 1970.

Lewis H.R., Papadimitriou C.H. The efficiency of algorithms. *Scientific American*, 96–109, January 1978.

Lidl R., Pilz G. *Applied abstract algebra.* Springer-Verlag, New York, 1984.

Mackiw G. *Applications of abstract algebra.* Wiley, New York, 1985.

MacWilliams F.J., Sloane N.J.A. *The theory of error-correcting codes, Part I and II.* North-Holland, New York, 1977.

Merkle R.C. Secure communications over insecure channels. *Communications of the ACM* **21**, 294–299, April 1978.

Merkle R.C., Hellman M.E. Hiding information and signatures in trapdoor knapsacks. *IEEE Transactions on Information Theory* **IT-24**, 525–530, 1978.

Merkle R.C., Hellman M.E. On the security of multiple encryption. *Communications of the ACM* **24**, 465–467, July 1981.

Peterson W.W., Weldon E.J. *Error-correcting codes*, 2 ed. MIT press, Cambridge MA, 1972.

Pohlig S.C., Hellman M.E. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Transactions on Information Theory* **IT-24**, 106–110, 1978.

Rivest R.L., Shamir A., Adleman L. A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM* **21**, 120–126, February 1978.

Shannon C.E. A mathematical theory of communication. *Bell System Technical Journal* **27**, 379–423, 623–656, 1948.

Shannon C.E. Communication theory of secrecy systems. *Bell System Technical Journal* **28**, 656–715, 1949.

Simmons G.J. Cryptology: The mathematics of secure communication. *Mathematical Intelligencer* **1**, 233–246, 1979.

Sinkov A. *Elementary cryptanalysis — a mathematical approach.* The New Mathematical Library no. 22, Mathematical Association of America, Washington, D.C., 1968.

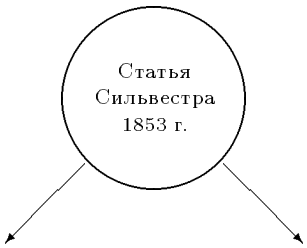
Wakerly J. *Error detecting codes, self-checking circuits and applications.* North-Holland, New York, 1978.

Williams H.C., Schmid B. Some remarks concerning the M.I.T. public-key cryptosystem. *BIT* **19**, 525–538, 1979.

Глава 5

Наибольшие общие делители полиномов над целыми числами и последовательности полиномиальных остатков

В этой главе мы продолжаем обсуждать вычисление наибольших общих делителей (gcd) полиномов и последовательностей полиномиальных остатков (PRS). (Читателю следует в этом месте вновь просмотреть разд. 3.2.1 и 3.2.2.) Однако, как отмечено в заглавии, мы ограничиваем изучение полиномами над кольцом целых чисел, где ключевой проблемой является ограничение роста коэффициентов. Мы детально исследуем два классических метода, существующие в литературе: метод Сильвестра–Габихта псевдоделения субрезультантных PRS (состоящий из двух алгоритмов), предложенный Сильвестром в 1853 г. и дополненный Габихтом в 1948 г., и метод матричной триангуляризации субрезультантных PRS, разработанный автором в 1986 г. (Akritas, 1986, 1988) (см. также историческое замечание 1 и рис. 5.1).



Метод псевдоделения субрезультантных PRS, предложенный Сильвестром в 1853 г. и дополненный Габихтом в 1948 г.

Метод матричной триангуляризации субрезультантных PRS, разработанный автором в 1986 г.

Рис. 5.1.

Обзор исторического развития двух классических методов субрезультантных PRS. Метод, разработанный Сильвестром, следует использовать только для полных PRS, в то время как методом Габихта следует пользоваться, когда PRS неполна. Метод матричной триангуляризации может быть использован в обоих случаях.

Как мы убедимся позже, при использовании метода матричной

триангуляризации коэффициенты полиномиальных остатков в определенных случаях меньше коэффициентов, полученных методом Сильвестра–Габихта; следовательно, первый метод лучше.

5.1

Введение и обоснование

Как мы уже видели, $\mathbb{Z}[x]$ не является ни полем, ни евклидовой областью, однако по теореме 3.2.10 — это область с однозначным разложением на множители, т.е. каждый необратимый элемент (по существу) единственным образом может быть представлен в виде произведения неприводимых полиномов. (Напомним, что каждое поле является областью с однозначным разложением на множители, в которой каждый ненулевой элемент является обратимым (*единицей*) и нет простых элементов. Целые числа образуют область с однозначным разложением на множители, где обратимыми являются ± 1, а простыми — $\pm 2, \pm 3, \pm 5, \ldots$.)

В разд. 3.2.3 мы сказали, что полином в кольце $\mathbb{Q}[x]$ называется *примитивным*, если его коэффициенты взаимно просты; это определение, так же как и утверждение теоремы 3.2.12, переносится на полиномы над произвольной областью с однозначным разложением на множители. Имеет место также следующая

Теорема 5.1.1. Пусть J — область с однозначным разложением на множители и $p(x) \in J[x]$ — ненулевой полином. Тогда $p(x)$ может быть единственным образом представлен в виде произведения $p(x) = c \cdot p'(x)$, где $c \in J$, а полином $p'(x)$ примитивен. Это разложение единственно с точностью до единиц кольца J, т.е. если $p(x) = c_1 \cdot p_1'(x) = c_2 \cdot p_2'(x)$, то $c_1 = bc_2$ и $p_2'(x) = bp_1'(x)$, где b — единица кольца J.

Доказательство. Доказательство очевидным образом следует из результатов гл. 3. □

Константа c в теореме 5.1.1 — это наибольший общий делитель коэффициентов полинома $p(x)$, называемый *содержанием* полинома $p(x)$ и обозначаемый $\mathrm{cont}[p(x)]$; очевидно, что тогда полином $p'(x)$, называемый *примитивной частью* полинома $p(x)$ и обозначаемый $\mathrm{pp}[p(x)]$, является примитивным полиномом в $J[x]$. [Обычно $\mathrm{pp}[p(x)]$ определяют так, чтобы его старший коэффициент был положителен.]

Поэтому любой ненулевой полином $p(x)$ в $J[x]$ имеет единственное представление вида (с точностью до обратимых элементов)

$$p(x) = \mathrm{cont}[p(x)] \cdot \mathrm{pp}[p(x)].$$

Например, рассмотрим два полинома $p_1(x) = 5x+10$ и $p_2(x) = -3x^2+9$ в $\mathbb{Z}[x]$. Тогда

$$\begin{aligned} &\mathrm{cont}[p_1(x)] = 5, && \mathrm{pp}[p_1(x)] = x+2, \\ &\mathrm{cont}[p_2(x)] = -3, && \mathrm{pp}[p_2(x)] = x^2-3, \\ &\mathrm{cont}[p_1(x)\cdot p_2(x)] = -15, && \mathrm{pp}[p_1(x)\cdot p_2(x)] = x^3+2x^2-3x-6. \end{aligned}$$

Отметим, что мы воспользовались теоремой 3.2.12.

Обратимся теперь к задаче нахождения наибольшего общего делителя двух полиномов $p_1(x), p_2(x)$ в кольце $\mathbb{Z}[x]$, являющемся не полем, а только областью с однозначным разложением на множители. Из приведенных выше рассуждений ясно, что мы можем вывести следующие важные соотношения:

$$\begin{aligned} \mathrm{cont}\{\gcd[p_1(x), p_2(x)]\} &= \gcd\{\mathrm{cont}[p_1(x)], \mathrm{cont}[p_2(x)]\}, \\ \mathrm{pp}\{\gcd[p_1(x), p_2(x)]\} &= \gcd\{\mathrm{pp}[p_1(x)], \mathrm{pp}[p_2(x)]\}. \end{aligned}$$

Поэтому задача нахождения наибольшего общего делителя произвольных полиномов сводится к задаче нахождения наибольшего общего делителя примитивных полиномов.

5.1.1. Общий обзор классических алгоритмов PRS

Рассмотрим два примитивных ненулевых полинома $p_1(x)$ и $p_2(x)$ в $\mathbb{Z}[x]$, у которых $\deg[p_1(x)] = m$ и $\deg[p_2(x)] = n$, $m > n$. Поскольку алгоритм **PDF** (из разд. 3.1) деления полиномов с остатком требует точной делимости на старший коэффициент полинома $p_2(x)$, $\mathrm{lc}[p_2(x)]$, обычно этот процесс невозможно выполнить для полиномов $p_1(x)$ и $p_2(x)$ над целыми числами, не ослабляя требования делимости. Поэтому мы вводим процесс *псевдоделения*, который всегда дает нам *псевдочастное* и *псевдоостаток* (prem), коэффициенты которых являются целыми числами.

Псевдоделение означает предварительное умножение полинома $p_1(x)$ на $\{\mathrm{lc}[p_2(x)]\}^{m-n+1}$, а затем применение алгоритма **PDF**, когда известно, что все частные существуют, т.е.

$$\{\mathrm{lc}[p_2(x)]\}^{m-n+1} p_1(x) = p_2(x)q(x) + r(x), \quad \deg[r(x)] < \deg[p_2(x)],$$

где $q(x)$ и $r(x)$ — псевдочастное и псевдоостаток соответственно.

Пример. Пользуясь псевдоделением в $\mathbb{Z}[x]$, разделим $p_1(x) = x^4 - 7x + 7$ на $p_2(x) = 3x^2 - 7$. Для того чтобы вычислить $q(x)$ и $r(x)$, предварительно умножим $p_1(x)$ на $3^{4-2+1} = 27$, а затем, применяя **PDF**, получаем $q(x) = 9x^2 + 21$ и $r(x) = -189x + 336$. Читатель может убедиться, что **PDF** не будет работать, если мы предварительно домножим $p_1(x)$ только на 3.

Поэтому, пытаясь вычислить наибольший общий делитель полиномов $p_1(x)$ и $p_2(x)$, мы должны убедиться, что выполнимы все деления полиномов, встречающиеся в этом процессе, т.е. мы должны, используя псевдоделения, сформировать последовательность полиномиальных остатков. Таким образом, мы приходим к следующему обобщенному алгоритму Евклида для полиномов.

GEA-P. Обобщенный алгоритм Евклида для полиномов над целыми числами (**G**eneralized **E**uclidean **A**lgorithm for **P**olynomials over the Integers)

Вход: $p_1(x), p_2(x)$ — ненулевые полиномы в $\mathbb{Z}[x]$; $\deg[p_1(x)] = n_1$, $\deg[p_2(x)] = n_2$, $n_1 \geq n_2$.

Выход: $\gcd[p_1(x), p_2(x)]$, наибольший общий делитель полиномов $p_1(x)$ и $p_2(x)$.

1. [Вычисление gcd содержаний] $c := \gcd\{\mathrm{cont}[p_1(x)], \mathrm{cont}[p_2(x)]\}$. (Здесь мы используем известный алгоритм Евклида для вычисления наибольшего общего делителя двух целых чисел.)

2. [Вычисление примитивных частей] $p_1'(x) := p_1(x)/\mathrm{cont}[p_1(x)]$; $p_2'(x) := p_2(x)/\mathrm{cont}[p_2(x)]$.

3. [Построение PRS] Вычислить $p_1'(x), p_2'(x), p_3(x), \ldots, p_h(x)$.

4. [Выход] Если $\deg[p_h(x)] = 0$, то вернуть $\gcd[p_1(x), p_2(x)] := c$, иначе вернуть $\gcd[p_1(x), p_2(x)] := c \cdot \mathrm{pp}[p_h(x)]$.

Ясно, что время работы этого алгоритма зависит от того, насколько эффективно мы можем вычислять последовательность полиномиальных остатков $p_1'(x), p_2'(x), p_3(x), \ldots, p_h(x)$. Заметим, что если $n_i = \deg[p_i(x)]$, то в общем случае мы можем утверждать, что члены этой последовательности удовлетворяют соотношениям

$$\{\mathrm{lc}[p_{i+1}(x)]\}^{n_i - n_{i+1}+1} p_i(x) = p_{i+1}(x)q_i(x) + \beta_i p_{i+2}(x),$$
$$\deg[p_{i+2}(x)] < \deg[p_{i+1}(x)],$$

где $i = 1, 2, \ldots, h-1$ для некоторого h. [Разумеется, $p_i(x) := p_i'(x)$, $i = 1, 2$, где $p_i'(x)$, $i = 1, 2$ определены на шаге 2 алгоритма **GEA-P**].

Если дан метод выбора коэффициентов β_i, то выписанное соотношение дает алгоритм построения PRS; разумеется, условие завершения этого семейства алгоритмов — равенство нулю псевдоостатка.

Ниже мы рассматриваем различные алгоритмы, полученные для разных значений β_i. Кроме того, мы кратко опишем метод матричной триангуляризации субрезультантных PRS, существенно отличающийся от всех остальных методов тем, что он не зависит от выбора элементов β_i; вместо явного псевдоделения полиномов метод матричной триангуляризации приводит матрицу к верхней треугольной форме, и все члены PRS получаются из строк итоговой матрицы.

Евклидов алгоритм PRS. Здесь $\beta_i = 1$ для всех $i = 1, 2, \ldots, h-1$, т.е. каждый псевдоостаток используется в том виде, в котором он получен. Это один из худших методов построения PRS, приводящий к экспоненциальному росту коэффициентов.

Пример. Рассмотрим полиномы $p_1(x) = x^3 - 7x + 7$, $p_2(x) = 3x^2 - 7$ в $\mathbb{Z}[x]$. Очевидно, что $\operatorname{cont}[p_1(x)] = \operatorname{cont}[p_2(x)] = 1$ и $p_i(x) = p_i'(x)$, $i = 1, 2$. Мы имеем такую последовательность:

$$\begin{aligned}
p_1(x) &= x^3 - 7x + 7, \\
p_2(x) &= 3x^2 - 7, & q_1(x) &= 3x, \\
p_3(x) &= -42x + 63, & q_2(x) &= -126x - 189, \\
p_4(x) &= -441, & q_3(x) &= 18522x - 27783,
\end{aligned}$$

полученную при выполнении следующих псевдоделений:

$$\begin{aligned}
(3)^2 p_1(x) &= p_2(x) \cdot (3x) + (-42x + 63), \\
(-42)^2 p_2(x) &= p_3(x) \cdot (-126x - 189) + (-441), \\
(-441)^2 p_3(x) &= p_4(x) \cdot (18522x - 27783) + 0.
\end{aligned}$$

Из шага 4 алгоритма **GEA-P** следует, что $\gcd[p_1(x), p_2(x)] = 1$. Отметим также, что в последнем псевдоделении коэффициенты имеют 8 десятичных цифр, поскольку $(-441)^2 p_3(x) = -8168202x + 12252303$.

Последовательность полиномиальных остатков этого примера называется *полной*, потому что степень каждого ее члена на единицу меньше степени предыдущего; два первых члена могут, конечно, иметь одинаковые степени. В противном случае последовательность называется *неполной*. Заметим, что не существует способа сказать a priori, будет PRS полной или неполной. См. также (Barnett, 1970, 1974; Brown et al., 1971; Collins, 1966, 1971).

Экспоненциальный рост коэффициентов членов PRS в приведенном примере обусловлен тем, что полиномы этой последовательности не являются примитивными, т.е. то, что мы не избавляемся от их содержания, дает вредный эффект. Эта ситуация исправляется ниже.

Алгоритм примитивных PRS. В этом случае $\beta_i = \mathrm{cont}\{\mathrm{prem}[p_i(x), p_{i+1}(x)]\}$, $i = 1, 2, \ldots, h-1$, где «prem» обозначает псевдоостаток, т.е. теперь мы удаляем содержание $(i+2)$-го члена PRS до того, как мы используем его. [Напомним, что для данного $p(x)$ удобно определять $\mathrm{pp}[p(x)]$ так, чтобы старший коэффициент был положительным.]

Пример. Рассмотрим те же полиномы, что и в предыдущем примере: $p_1(x) = x^3 - 7x + 7$, $p_2(x) = 3x^2 - 7$ в $\mathbb{Z}[x]$, где снова $p_i(x) = p_i'(x)$, $i = 1, 2$. Теперь мы получаем

$$\begin{aligned}
&p_1(x) = x^3 - 7x + 7, && && \\
&p_2(x) = 3x^2 - 7, && q_1(x) = 3x, && \\
&p_3(x) = 2x - 3, && q_2(x) = 6x + 9, && \beta_1 = -21, \\
&p_4(x) = 1, && q_3(x) = 2x - 3, && \beta_2 = -1,
\end{aligned}$$

что достигается выполнением следующих псевдоделений:

$$\begin{aligned}
(3)^2 p_1(x) &= p_2(x) \cdot (3x) + (-21)(2x - 3), \\
(2)^2 p_2(x) &= p_3(x) \cdot (6x + 9) + (-1), \\
(1)^2 p_3(x) &= p_4(x) \cdot (2x - 3) + 0.
\end{aligned}$$

Из этого примера ясно видно, что этот алгоритм настолько хорош, насколько этого можно добиться в отношении роста коэффициентов членов PRS. Однако на каждом шаге требуется вычислять один или более наибольших общих делителей коэффициентов, а эти вычисления становятся все более сложными по мере роста коэффициентов. Поэтому нам хотелось бы найти способ избежать большей части этих вычислений и все-таки резко уменьшить рост коэффициентов по сравнению с тем, который происходит в евклидовом алгоритме PRS. Это может быть достигнуто использованием либо метода псевдоделения Сильвестра–Габихта, либо метода матричной триангуляризации субрезультантных PRS; детальное описание этих методов может быть найдено в разд. 5.2 и 5.3 соответственно.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS. С этим методом связаны два алгоритма: алгоритм Сильвестра *редуцированных (субрезультантных)* PRS, если последовательность полна, и алгоритм Габихта *субрезультантных* PRS, если последовательность неполна. (См. историческое замечание 1.)

В соответствии с этим методом в случае полной PRS мы выбираем

$$\begin{aligned} &\beta_1 = 1, \\ &\beta_i = \{\mathrm{lc}[p_i(x)]\}^2, \quad i = 2, 3, \ldots, h-1 \end{aligned} \tag{S}$$

[алгоритм Сильвестра редуцированных (субрезультантных) PRS], в то время как в случае неполных PRS мы полагаем

$$\begin{aligned} &\beta_1 = (-1)^{n_1 - n_2 + 1}, \\ &\beta_i = (-1)^{n_i - n_{i+1} + 1} \, \mathrm{lc}[p_i(x)] \cdot H_i^{n_i - n_i + 1}, \quad i = 2, 3, \ldots, h-1 \end{aligned} \tag{H}$$

(алгоритм Габихта субрезультантных PRS); здесь

$$\begin{aligned} &H_2 = \{\mathrm{lc}[p_2'(x)]\}^{n_1 - n_2}, \quad \text{где } p_2'(x) = \mathrm{pp}[p_2(x)] \text{ и} \\ &H_i = \{\mathrm{lc}[p_i(x)]\}^{n_{i-1} - n_i} H_{i-1}^{1-(n_{i-1} - n_i)}, \quad i = 3, \ldots, h-1. \end{aligned}$$

Выбор значений в соотношениях (S) и (H) разъясняется в разд. 5.2.1 и 5.2.3 соответственно. В случае полных PRS метод Габихта сводится к методу Сильвестра. В общем случае соотношения (S) могут быть записаны в виде

$$\begin{aligned} &\beta_1 = 1, \\ &\beta_i = \{\mathrm{lc}[p_i(x)]\}^{n_{i-1} - n_i + 1}, \quad i + 2, 3, \ldots, h-1, \end{aligned} \tag{S$'$}$$

известном также как алгоритм *Сильвестра редуцированных* (субрезультантных) PRS.

Заметим, что в обоих приведенных выше алгоритмах последовательности полиномиальных остатков вычислялись с помощью классического алгоритма деления полиномов **PDF**; более того, вместо вычисления содержания $(i+2)$-го члена PRS мы просто делили его коэффициенты на β_i, зная, что это деление может быть выполнено *без остатка*. (Почему это так, мы увидим в разд. 5.2.1 и 5.2.3). В обоих приведенных алгоритмах мы имели также равенство $\beta_1 = 1$, означавшее, что полином $p_3(x)$ не может быть редуцирован, и читателю следует помнить об этом.

Как уже упоминалось, Габихт обобщил результат Сильвестра, и поэтому его алгоритм PRS может применяться также для полных PRS (конечно, за счет дополнительной стоимости). Однако алгоритм Сильвестра PRS [(S) или (S′)] работает не очень хорошо, если наша PRS неполна, т.е. рост может быть экспоненциальным, хотя и не таким быстрым, как в случае алгоритма Евклида PRS. (Напомним, что не существует способа a priori узнать, будет PRS полной или неполной.)

Пример. Снова рассмотрим полиномы $p_1(x) = x^3 - 7x + 7$, $p_2(x) = 3x^2 - 7$ в $\mathbb{Z}[x]$, где $\operatorname{cont}[p_1(x)] = \operatorname{cont}[p_2(x)] = 1$ и $p_i(x) = p_i'(x)$, $i = 1, 2$. Имеем

$$
\begin{aligned}
&p_1(x) = x^3 - 7x + 7, \\
&p_2(x) = 3x^2 - 7, && q_1(x) = 3x, \\
&p_3(x) = -42x + 63, && q_2(x) = -126x - 189, \\
&p_4(x) = -49, && q_3(x) = 18522x - 27783.
\end{aligned}
$$

Поскольку мы имеем дело с полной PRS, оба алгоритма в методе Сильвестра–Габихта псевдоделения субрезультантных PRS порождают одну и ту же последовательность, которая получается такой же, как и для евклидова алгоритма PRS. Однако отметим, что теперь $p_4(x) = -49$, а не -441, поскольку $p_4(x)$ разделили на $9 = \{\operatorname{lc}[p_2(x)]\}^2$. [Здесь $p_4(x)$ — моном, но это несущественно.]

Метод матричной триангуляризации субрезультантных PRS. В отличие от приведенных выше алгоритмов PRS метод матричной триангуляризации не использует явного деления полиномов. Чтобы посмотреть, как он работает, рассмотрим два полинома в $\mathbb{Z}[x]$: $p_1(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0$ и $p_2(x) = d_m x^m + d_{m-1}x^{m-1} + \cdots + d_0$, $c_n \neq 0$, $d_m \neq 0$, $n \geq m$, для которых мы хотим вычислить последовательность полиномиальных остатков. Прежде всего построим матрицу, соответствующую результанту в форме Сильвестра (определение см. в разд. 5.2.2 и 5.3.3), являющуюся в нашем случае выписанной ниже квадратной матрицей порядка $2n$ [$p_2(x)$ преобразуется

в полином степени n добавлением нулевых коэффициентов]:

$$\begin{bmatrix} c_n & c_{n-1} & \dots & c_0 & 0 & 0 & \dots & 0 \\ d_n & d_{n-1} & \dots & d_0 & 0 & 0 & \dots & 0 \\ 0 & c_n & \dots & \cdot & c_0 & 0 & \dots & 0 \\ 0 & d_n & \dots & \cdot & d_0 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & c_n & c_{n-1} & \cdot & \dots & c_0 \\ 0 & \dots & 0 & d_n & d_{n-1} & \cdot & \dots & d_0 \end{bmatrix}.$$

Затем эту матрицу мы приводим к верхней треугольной форме, используя преобразования, сохраняющие целочисленность матрицы, и все члены (полной или неполной) PRS получаются из строк результирующей матрицы с помощью доказанной автором теоремы (теорема 5.3.6); время вычислений метода матричной триангуляризации — *то же самое*, что и у метода Сильвестра–Габихта псевдоделения.

Заметим, что сильвестрова форма результанта используется впервые в настоящем столетии; она была погребена в статье Сильвестра 1853 г. и только однажды была использована Ван Влеком в 1899 г.

В следующем примере коэффициенты полиномов, получающихся с помощью метода матричной триангуляризации субрезультантных PRS, меньше коэффициентов, полученных методом псевдоделения Сильвестра–Габихта.

Пример. Снова рассмотрим полиномы $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$. Как мы уже видели, оба алгоритма метода Сильвестра–Габихта псевдоделения субрезультантных PRS дают полиномы $p_3(x) = -42x + 63$, $p_4(x) = -49$, а метод матричной триангуляризации дает $p_3(x) = -14x + 21$, $p_4(x) = -49$ (детали реализации обсуждаются в разд. 5.3.3). Соответствующая результирующая матрица:

$$\begin{bmatrix} 1 & 0 & -7 & 7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 & 0 \\ 0 & 0 & 9 & 0 & -21 & 0 \\ 0 & 0 & 0 & -42 & 63 & 0 \\ 0 & 0 & 0 & 0 & 196 & -294 \\ 0 & 0 & 0 & 0 & 0 & -49 \end{bmatrix} *$$

Помеченная звездочкой строка означает, что при выборе ведущего элемента третья строка переставлена с четвертой. Коэффициенты полинома $p_3(x) = -14x + 21$ получены из третьей строки *до* перестановки, и мы видим, что они в три раза меньше полученных методом псевдоделения Сильвестра–Габихта. Это происходит потому, что

частное от псевдоделения $p_1(x)$ на $p_2(x)$ равно $3x$ и *не содержит свободного члена*; это означает, что нам не нужно умножать $p_1(x)$ на 3^2, достаточно на 3. (Однако нет способа узнать об этом до выполнения деления.)

Следующим рассмотрим пример неполной PRS.

Пример. Если мы рассмотрим полиномы $p_1(x) = x^5 + 5x^4 + 10x^3 + 5x^2 + 5x + 2$ и $p_2(x) = x^4 + 4x^3 + 6x^2 + 2x + 1$, то верхняя треугольная форма матрицы, соответствующей форме Сильвестра результанта, равна

$$\begin{bmatrix} 1 & 5 & 10 & 5 & 5 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & 3 & 4 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 7 & 1 & 3 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & -2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 9 & -6 & -3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 58 & 50 & 18 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -266 & -112 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -756 & -532 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -980 \end{bmatrix} *$$

Помеченная $*$ строка означает, что имеет место перестановка строк. По теореме 5.3.6 члены неполной PRS, порожденной полиномами $p_1(x)$ и $p_2(x)$, суть $3x^2 - 2x - 1$, $-266x - 112$ и -980.

5.1.2. Неклассический подход: модулярный алгоритм для наибольшего общего делителя

Как уже говорилось, в следующих разделах мы детально рассмотрим два метода субрезультантных PRS. Однако непосредственно сейчас мы представим алгоритм, который читатель должен был предвидеть.

Ключевое для этого алгоритма наблюдение состоит в том, что вычисление наибольшего общего делителя двух полиномов над конечным полем не приводит к росту, поскольку коэффициенты не могут превосходить размер модуля. Поэтому мы можем преобразовать задачу нахождения наибольшего общего делителя двух полиномов над целыми числами к нескольким задачам нахождения наибольшего общего делителя двух полиномов над некоторыми конечными полями. По теореме 5.1.2, сформулированной ниже, ответы к этим задачам

затем интерполируются (с использованием греко-китайского алгоритма), чтобы получить истинный наибольший общий делитель над целыми числами. Это в общих чертах составляет *модулярный* gcd-*алгоритм*. В следующих рассуждениях $p^{(p)}(x)$ обозначает полином $p(x)$ в $\mathbb{Z}_p[x]$.

Теорема 5.1.2. Пусть $p_1(x)$ и $p_2(x)$ — два полинома в $\mathbb{Z}[x]$, и $p_h(x) = \gcd[p_1(x), p_2(x)]$. Тогда для всех простых p, таких, что p не делит $\mathrm{lc}[p_1(x)] \cdot \mathrm{lc}[p_2(x)]$, имеет место неравенство $\deg[p_h^{(p)}(x)] \geq \deg[p_h(x)]$, где $p_h^{(p)}(x) = \gcd[p_1(x), p_2(x)]$ над $\mathbb{Z}_p[x]$, и существует только конечное число простых p, таких, что $\deg[p_h^{(p)}(x)] > \deg[p_h(x)]$.

Доказательство. Поскольку p не делит $\mathrm{lc}[p_1(x)] \cdot \mathrm{lc}[p_2(x)]$, то, следовательно, p не делит старший коэффициент полинома $p_h(x)$, а значит, первое утверждение тривиально. Второе утверждение вытекает из наблюдения, что степень полинома $p_h^{(p)}(x)$ может превосходить степень полинома $p_h(x)$, только если p делит один из старших коэффициентов предыдущих членов PRS, а это может случиться только для конечного числа простых чисел. □

Простые числа p, для которых $\deg[p_h^{(p)}(x)] > \deg[p_h(x)]$, называются «несчастливыми» и отбрасываются в описанном ниже алгоритме; более того, с целью эффективности каждое простое число p выбирается меньшим машинного слова. В модулярном gcd-алгоритме используются также следующие факты:

(1) Если полином $p_1(x)$ неприводим над целыми по модулю m, где m не делит $\mathrm{lc}[p_1(x)]$, то $p_1(x)$ неприводим также над целыми числами; по существу это теорема 3.2.16.

(2) Если полиномы $p_1(x)$ и $p_2(x)$ взаимно просты над целыми по модулю m, где m не делит $\mathrm{lc}[p_1(x)]$ и не делит $\mathrm{lc}[p_2(x)]$, то полиномы $p_1(x)$ и $p_2(x)$ взаимно просты над целыми числами; это следует непосредственно из теоремы 5.1.2.

M. Модулярный алгоритм (**M**odular Algorithm)

Вход: Два примитивных полинома $p_1(x), p_2(x)$ от одной переменной с целыми коэффициентами.

Выход: Полином $p_h(x) = \gcd[p_1(x), p_2(x)]$ над целыми числами.

1. [Инициализация] Положить $c := \gcd\{\mathrm{lc}[p_1(x)], \mathrm{lc}[p_2(x)]\}$. Выбрать простое p, которое не делит $\mathrm{lc}[p_1(x)] \cdot \mathrm{lc}[p_2(x)]$, и вычислить *нормированный* полином $p_h^{(p)}(x) = \gcd[p_1(x), p_2(x)]$ в $GF(p)$. Положить $d := \deg[p_h^{(p)}(x)]$; если $d = 0$, то вернуть $p_h(x) := 1$.

2. [Готово?] Положить $g(x) := c \cdot p_h^{(p)}(x) \pmod p$.
Если $\mathrm{pp}[g(x)] | p_1(x)$ и $\mathrm{pp}[g(x)] | p_2(x)$ (где оба деления выполняются над целыми числами), то вернуть $p_h(x) := \mathrm{pp}[g(x)]$.

3. [Новое простое] Выбрать новое простое число p_1, которое не делит $\mathrm{lc}[p_1(x)] \cdot \mathrm{lc}[p_2(x)]$, и вычислить нормированный полином $p_h^{(p_1)}(x) = \gcd[p_1(x), p_2(x)]$ в $GF(p_1)$; если $\deg[p_h^{(p_1)}(x)] = 0$, то вернуть $p_h(x) := 1$.

4. [Несчастливое простое?] Если $\deg[p_h^{(p_1)}(x)] > d$, то p_1 — несчастливое простое число; перейти к шагу 3. Если $\deg[p_h^{(p_1)}(x)] < d$, то p — несчастливое простое число; положить $p_h^{(p)}(x) := p_h^{(p_1)}(x)$, $d := \deg[p_h^{(p)}(x)]$, $p := p_1$ и перейти к шагу 2.

5. [Цикл по коэффициентам] В этой точке $\deg[p_h^{(p_1)}(x)] = \deg[p_h^{(p)}(x)]$. Положить $p'_h(x) := 0$. Для $i := 0, 1, \ldots, d$ выполнять: применить **GCRA2** к коэффициентам при x^i в $p_h^{(p)}(x)$ и $p_h^{(p_1)}(x)$ по модулю p и p_1; пусть q — результат интерполяции; положить $p'_h(x) := p'_h(x) + qx^i$. [При использовании симметричной системы вычетов коэффициенты полинома $p'_h(x)$ по абсолютной величине $\leq \lfloor p \cdot p_1/2 \rfloor$; заметим также, что полином $p'_h(x)$ *нормирован*.]

6. [Обновление] Положить $p := p \cdot p_1$, $p_h^{(p)}(x) := p'_h(x)$ и перейти к шагу 2.

Анализ времени работы алгоритма ***M***. Из теоремы 5.1.2 следует, что приведенный выше алгоритм завершается, поскольку число несчастливых простых чисел конечно. [См. также (Brown, 1971, th. 1), где дана оценка числа возможных несчастливых простых чисел.] Браун показал, что преимущество модулярного алгоритма над классическими проявляется только тогда, когда данные полиномы достаточно большие и достаточно плотные. В разреженном случае (много отсутствующих членов) преимущество анализировать трудно.

Пример предложен чуть дальше. Предыдущий алгоритм не использует априорную оценку числа различных используемых простых чисел; интересная версия алгоритма **M**, использующая такую оценку, приводится ниже.

M1. Модулярный алгоритм, версия 1 (**M**odular Algorithm, Version **1**)

Вход: Два примитивных полинома $p_1(x), p_2(x)$ от одной переменной с целыми коэффициентами.

Выход: Полином $p_h(x) = \gcd[p_1(x), p_2(x)]$ над целыми числами.

1. [Инициализация] Выбрать простое p, которое не делит $\mathrm{lc}[p_1(x)] \cdot \mathrm{lc}[p_2(x)]$, и вычислить $p_h^{(p)}(x) = \gcd[p_1(x), p_2(x)]$ в $GF(p)$. Положить $d := \deg[p_h^{(p)}(x)]$; если $d = 0$, то вернуть $p_h(x) := 1$. Положить $B := \max[|p_1(x)|_\infty, |p_2(x)|_\infty]$.

2. [Новое простое] Выбрать новое простое число p_1, которое не делит $\mathrm{lc}[p_1(x)] \cdot \mathrm{lc}[p_2(x)]$. Вычислить $p_h^{(p_1)}(x) = \gcd[p_1(x), p_2(x)]$ в $GF(p_1)$. Если $\deg[p_h^{(p_1)}(x)] = 0$, то вернуть $p_h(x) := 1$.

3. [Несчастливое простое?] Если $\deg[p_h^{(p_1)}(x)] > d$, то p_1 — несчастливое простое число; перейти к шагу 2. Если $\deg[p_h^{(p1)}(x)] < d$, то p — несчастливое простое число; положить $p_h^{(p)}(x) := p_h^{(p_1)}(x)$, $d := \deg[p_h^{(p)}(x)]$, $p := p_1$ и перейти к шагу 2.

4. [Цикл по коэффициентам] В этой точке $\deg[p_h^{(p_1)}(x)] = \deg[p_h^{(p)}(x)]$. Положить $p'_h(x) := 0$. Для $i := 0, 1, \ldots, d$ выполнять: применить **GCRA2** к коэффициентам при x^i в $p_h^{(p)}(x)$ и $p_h^{(p_1)}(x)$ по модулю p и p_1 соответственно; пусть q — результат интерполяции, положить $p'_h(x) := p'_h(x) + qx^i$. (Использовать симметричную систему вычетов.)

5. [Готово?] Положить $p_h(x) := p'_h(x)$, $p := p \cdot p_1$. Если $p > 2B$, то вернуть $p_h(x) := \mathrm{pp}[p_h(x)]$, иначе перейти на шаг 2.

В действительности значение B, используемое в приведенном выше алгоритме, — это оценка максимума абсолютной величины коэффициентов полинома $p_h(x)$. Такая оценка может быть получена из теоремы А.О. Гельфонда (*Трансцендентные и алгебраические числа* М.: Гостехиздат, 1952), но в практических вычислениях вместо этого используется максимум абсолютных величин коэффициентов полиномов $p_1(x)$ и $p_2(x)$. Выбирается значение $2B$, так как оно работает в большинстве случаев (Brown, 1971). Ясно, что существуют исключения; например, если $p(x) = x^8+2x^7+3x^6+4x^5+5x^4+4x^3+3x^2+2x+1$ и $q(x) = x-1$, то $p(x)q(x) = x^9+x^8+x^7+x^6+x^5-x^4-x^3-x^2-x-1$, и мы имеем дело со случаем, когда коэффициенты делителя больше коэффициентов делимого.

Заметим, что обе версии модулярного gcd-алгоритма легко обобщить для обработки полиномов от многих переменных. Можно также воспользоваться алгоритмами подъема, описанными в гл. 6, чтобы найти наибольший общий делитель двух полиномов над $\mathbb{Z}$. Это — так называемый расширенный gcd-алгоритм Цассенхаузена [**E**xtended

Zassenhaus **GCD** (**EZGCD**)], который не описывается здесь, но подробности можно найти в литературе (Miola et al., 1974; Moses et al., 1973; Yun, 1974).

Пример. Найдем наибольший общий делитель полиномов $p_1(x) = 9x^5 + 6x^4 + 3x^3 + 3x^2 + 2x + 1$ и $p_2(x) = 3x^4 + 5x^3 + 6x^2 + 3x + 1$.

Используя **M**, заметим сначала, что $c = 3$, а затем для $p = 5$ получим $p_1^{(5)}(x) = 4x^5+x^4+3x^3+3x^2+2x+1$, $p_2^{(5)}(x) = 3x^4+x^2+3x+1$ и $p_h^{(5)}(x) = \gcd[p_1(x), p_2(x)] = 2x^2+3x+4 = x^2+4x+2$ в $GF(5)$. Затем, на шаге 2 вычисляем полином $g(x) = 3x^2+2x+1$, который делит как $p_1(x)$, так и $p_2(x)$. Поэтому $p_h(x) = 3x^2 + 2x + 1$.

Используя **M1**, сначала вычислим $p_h^{(5)}(x) = \gcd[p_1(x), p_2(x)] = 2x^2 + 3x + 4$ в $GF(5)$, а затем $p_h^{(7)}(x) = \gcd[p_1(x), p_2(x)] = 5x^2 + x + 4$ в $GF(7)$. Поскольку $5 \cdot 7 = 35 > 2B = 18$, этих двух простых чисел достаточно. С помощью греко-китайской теоремы об остатках мы из полиномов $p_h^{(5)}(x)$ и $p_h^{(7)}(x)$ получаем полином $g(x) = 12x^2 + 8x + 4$ и $p_h(x) = \mathrm{pp}[g(x)] = 3x^2 + 2x + 1$.

5.2

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

В этом разделе мы детально изучаем метод Сильвестра–Габихта псевдоделения субрезультантных PRS. По существу мы собираемся показать, почему значения β_i, указанные в разд. 5.1.1 в связи с алгоритмами Сильвестра и Габихта PRS, делят $p_{i+2}(x)$, $(i+2)$-й член PRS. В этом месте читателя просят вновь просмотреть свойства определителей в приложении в конце этой книги.

5.2.1. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

В своей статье 1853 г. Сильвестр разработал метод вычисления полной последовательности полиномиальных остатков над целыми числами; более того, он сохранял коэффициенты членов PRS *насколько возможно малыми*, удаляя их *распределенные сомножители* (используя терминологию Сильвестра), а *не* их содержание (операция, включающая вычисление наибольшего общего делителя целых

чисел). Следует также отметить, что Сильвестр интересовался вычислением последовательностей Штурма (рассматриваемых в гл. 7), где вычисляется противоположный для каждого полиномиального остатка в PRS, т.е. $p_i(x) = p_{i+1}(x)q_i(x) - p_{i+2}(x)$. Поэтому определители в его доказательстве в точности совпадают с определителями, используемыми ниже, за исключением того, что в них переставлены вторая и третьи строки (Akritas, 1987). (Результанты и субрезультанты определяются в следующем разделе.)

Теорема 5.2.1 (Сильвестр 1853). Пусть $p_1(x), p_2(x), p_3(x), \ldots, p_h(x)$ — *полная* последовательность полиномиальных остатков, $p_i(x) \in \mathbb{Z}[x]$ для $i = 1, 2, \ldots, h$. Тогда $\{\mathrm{lc}[p_i(x)]\}^2 | p_{i+2}(x)$, $1 < i \leq h-2$, т.е. квадрат старшего коэффициента полинома $p_i(x)$ является делителем полинома $p_{i+2}(x)$.

Доказательство. Пусть $a, b, c, d, \ldots$ — коэффициенты какого-либо делимого $p_{i-1}(x)$, а $\alpha, \beta, \gamma, \delta, \ldots$ — делителя $p_i(x)$, где $a = \mathrm{lc}[p_{i-1}(x)]$ и $\alpha = \mathrm{lc}[p_i(x)]$. Тогда, как легко видеть, коэффициенты остатка $p_{i+1}(x)$, образующие второй делитель, равны

$$\left(\frac{1}{a^2}\right) \det \begin{bmatrix} a & b & c \\ \alpha & \beta & \gamma \\ 0 & \alpha & \beta \end{bmatrix}, \quad \left(\frac{1}{a^2}\right) \det \begin{bmatrix} a & b & d \\ \alpha & \beta & \delta \\ 0 & \alpha & \gamma \end{bmatrix},$$
$$\left(\frac{1}{a^2}\right) \det \begin{bmatrix} a & b & e \\ \alpha & \beta & \varepsilon \\ 0 & \alpha & \delta \end{bmatrix}, \ldots .$$

[Читателю следует проверить, что эти определители — коэффициенты остатка $p_{i+1}(x)$.] Таким же образом, полагая

$$m := \det \begin{bmatrix} a & b & c \\ \alpha & \beta & \gamma \\ 0 & \alpha & \beta \end{bmatrix},$$

видим, что каждый коэффициент второго остатка будет иметь форму блочного определителя [заметим, что, присваивая значение переменной m, нам не нужно делить определитель на a^2 (почему?)]:

$$\left(\frac{1}{m^2}\right) \det \begin{bmatrix} & \alpha & & & \beta & & & \gamma & \\ a & b & c & a & b & d & a & b & e \\ \alpha & \beta & \gamma & \alpha & \beta & \delta & \alpha & \beta & \varepsilon \\ 0 & \alpha & \beta & 0 & \alpha & \gamma & 0 & \alpha & \delta \\ & & & a & b & c & a & b & d \\ & 0 & & \alpha & \beta & \gamma & \alpha & \beta & \delta \\ & & & 0 & \alpha & \beta & 0 & \alpha & \gamma \end{bmatrix},$$

где выписанное выше выражение — это $\mathrm{lc}[p_{i+2}(x)]$. (3×3-элементы в выражении для выписанного выше определителя сами являются определителями.) Опуская общий множитель $(1/m^2)$ и разлагая по первому столбцу, мы видим, что выписанный выше определитель равен

$$\alpha \left\{ \det \begin{bmatrix} a & b & d \\ \alpha & \beta & \delta \\ 0 & \alpha & \gamma \end{bmatrix} \cdot \det \begin{bmatrix} a & b & d \\ \alpha & \beta & \delta \\ 0 & \alpha & \gamma \end{bmatrix} - \det \begin{bmatrix} a & b & c \\ \alpha & \beta & \gamma \\ 0 & \alpha & \beta \end{bmatrix} \cdot \det \begin{bmatrix} a & b & e \\ \alpha & \beta & \varepsilon \\ 0 & \alpha & \delta \end{bmatrix} \right\}$$

$$- \det \begin{bmatrix} a & b & c \\ \alpha & \beta & \gamma \\ 0 & \alpha & \beta \end{bmatrix} \cdot \left\{ \beta \det \begin{bmatrix} a & b & d \\ \alpha & \beta & \delta \\ 0 & \alpha & \gamma \end{bmatrix} - \gamma \det \begin{bmatrix} a & b & c \\ \alpha & \beta & \gamma \\ 0 & \alpha & \beta \end{bmatrix} \right\}.$$

Снова разлагая, мы видим, что для некоторых выражений A, B первый член имеет вид

$$\alpha^2 A + \alpha(a\beta\gamma \cdot a\beta\gamma - a\beta^2 \cdot a\beta\delta),$$

в то время как второй член имеет вид (разлагаем и упрощаем выражение в первых скобках)

$$\alpha^2 B - \alpha(\gamma^2 - \beta\delta) a^2 \beta^2.$$

Следовательно, после сокращения весь определитель имеет вид $\alpha^2(A+B)$, и мы видим, что α^2 будет входить в качестве сомножителя в этот и остальные коэффициенты полинома $p_{i+2}(x)$. □

Заметим, что в теореме 5.2.1 мы не можем редуцировать коэффициенты полинома $p_3(x)$; предложение применяется к коэффициентам полиномов $p_4(x), p_5(x)$ и т.д.

Сильвестр указывает, что «тот же самый явный метод может применяться, чтобы показать, что если степень первого делителя была бы на ϵ единиц, а не на одну, меньше степени первого делимого, то α^{e+1} содержалось бы в каждом члене второго вычета; однако сложность доказательства этим методом возрастает с возрастанием ϵ» (Sylvester, 1853, p. 419).

Пример. Рассмотрим снова полиномы $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$ в $\mathbb{Z}[x]$. Как уже говорилось, последовательность полиномиальных остатков, полученная с помощью евклидова алгоритма PRS, — это $p_1(x) = x^3 - 7x + 7$, $p_2(x) = 3x^2 - 7$, $p_3(x) = -42x + 63$, $p_4(x) = -441$.

Пользуясь теоремой 5.2.1, вычислим коэффициенты полинома $p_3(x)$ как определители, полученные из матрицы

$$\begin{bmatrix} 1 & 0 & -7 & 7 \\ 3 & 0 & -7 & 0 \\ 0 & 3 & 0 & -7 \end{bmatrix}.$$

Эти определители суть

$$\det \begin{bmatrix} 1 & 0 & -7 \\ 3 & 0 & -7 \\ 0 & 3 & 0 \end{bmatrix} = -42 \quad \text{и} \quad \det \begin{bmatrix} 1 & 0 & 7 \\ 3 & 0 & 0 \\ 0 & 3 & -7 \end{bmatrix} = 63.$$

(В действительности они являются субрезультантами и определяются в следующем разделе.) Заметим, что в этом месте мы не можем выполнить какое-либо деление, поскольку $p_3(x)$ — это первый остаток.

Подобным образом единственный коэффициент полинома $p_4(x)$ получается из определителя

$$(1/3^2) \det \begin{bmatrix} 3 & 0 & -7 \\ -42 & 63 & 0 \\ 0 & -42 & 63 \end{bmatrix} = (1/9) \cdot (-441) = -49.$$

Легко видеть, что в этом случае мы можем разделить без остатка 441 на 3^2, где $3 = \mathrm{lc}[p_2(x)]$, и получить гораздо меньший коэффициент.

Используя соотношение (S′) из разд. 5.1.1, получаем следующий алгоритм:

SRSPRS. Редуцированные (субрезультантные) PRS Сильвестра (**S**ylvester's **r**educed (**s**ubresultant) **PRS**)

Вход: Два ненулевых полинома $p_1(x), p_2(x)$ в $\mathbb{Z}[x]$, $n_1 = \deg[p_1(x)] \geq n_2 = \deg[p_2(x)]$.

Выход: Редуцированная (субрезультантная) PRS полиномов $p_1(x)$ и $p_2(x)$.

1. [Инициализация] $i := 1$; $c := 1$; $n_0 := n_1$.

2. [Готово?] Если $p_{i+1}(x) = 0$, то выход.

3. [Вычисление псевдоостатка] Вывести $p_{i+2}(x) := \mathrm{prem}[p_i(x), p_{i+1}(x)]/c^{n_{i-1}-n_i+1}$; положить $i := i + 1$, $c := \mathrm{lc}[p_i(x)]$ $n_{i+1} := \deg[p_{i+1}(x)]$ и перейти к шагу 2.

Анализ времени работы алгоритма ***SRSPRS***. Время работы этого алгоритма равно

$$t_{\mathbf{SRSPRS}}[p_1(x), p_2(x)] = O\{n^4 L^2 [|p(x)|_\infty]\},$$

где $|p(x)|_\infty = \max(|p_1(x)|_\infty, |p_2(x)|_\infty)$. Доказательство этого факта использует результанты (обсуждаемые в следующем разделе) и опускается, поскольку оно совпадает с доказательством анализа времени вычислений алгоритма **HSPRS**, приведенном в разд. 5.2.3. (Кстати, оно совпадает со временем вычислений классического PRS-алгоритма.)

Теорема Сильвестра указывает отношение делимости, существующее между *некоторыми* членами *полных* PRS, в качестве результата которого мы можем легко получить существующие отношения делимости. Однако для того, чтобы увидеть, почему работает алгоритм Габихта субрезультантных PRS (который, напоминаем, используется в неполных случаях), нам понадобится более общее отношение делимости, т.е. нам требуется отношение делимости между *любыми* членами PRS. На это новое отношение делимости намекнул Сильвестр, но впервые оно было представлено Габихтом в 1948 г. Прежде, однако, нам потребуются некоторые дополнительные понятия, обсуждаемые в следующем разделе.

5.2.2. Результант двух полиномов

Результанты и субрезультанты существенны для глубокого понимания PRS-алгоритм Габихта, и в этом разделе мы изучаем их свойства. (Сильвестр неявно использовал их.) Великолепной работой является (Netto, 1896); см. также (Bocher, 1907) и (Lipson, 1981).

Рассмотрим два полинома в $\mathbb{Z}[x]$:

$$p_1(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_0, \quad c_n \neq 0,\ n > 0,$$
$$p_2(x) = d_m x^m + d_{m-1} x^{m-1} + \cdots + d_0, \quad d_m \neq 0,\ m > 0,$$

и найдем условия, при выполнении которых они имеют общий делитель или (по крайней мере один) общий корень.

Обозначим корни полинома $p_1(x) = 0$ через $\alpha_1, \alpha_2, \ldots, \alpha_n$, а полинома $p_2(x) = 0$ — через $\beta_1, \beta_2, \ldots, \beta_m$. Тогда для того, чтобы полиномы $p_1(x)$ и $p_2(x)$ имели общий корень, некоторое α_i должно быть равно

некоторому β_j и должно обращаться в нуль следующее произведение, обозначаемое через $\operatorname{res}[p_1(x), p_2(x)]$:

$$\begin{aligned}\operatorname{res}[p_1(x), p_2(x)] = c_n^m d_m^n &(\alpha_1 - \beta_1)(\alpha_1 - \beta_2)\dots(\alpha_1 - \beta_m)\\ &(\alpha_2 - \beta_1)(\alpha_2 - \beta_2)\dots(\alpha_2 - \beta_m)\\ &\vdots\\ &(\alpha_n - \beta_1)(\alpha_n - \beta_2)\dots(\alpha_n - \beta_m);\end{aligned}$$

это и есть то условие, которое мы ищем. [Читателю следует отметить, что между $\operatorname{res}[p_1(x), p_2(x)]$ и переменной res, использованной в разд. 2.4 и 3.1.4, нет никакой связи; эта переменная обозначена сокращением слова result (результат).] Группируя n строк и m столбцов и принимая во внимание, что

$$\begin{aligned}p_1(x) &= (-1)^n c_n(\alpha_1 - x)(\alpha_2 - x)\dots(\alpha_n - x),\\ p_2(x) &= d_m(x - \beta_1)(x - \beta_2)\dots(x - \beta_m),\end{aligned}$$

имеем

$$\begin{aligned}\operatorname{res}[p_1(x), p_2(x)] &= c_n^m p_2(\alpha_1)p_2(\alpha_2)\dots p_2(\alpha_n)\\ &= (-1)^{mn} d_m^n p_1(\beta_1)p_1(\beta_2)\dots p_1(\beta_m). \qquad \text{(R)}\end{aligned}$$

Таким образом, мы видим, что множитель $c_n^m d_m^n$ необходим, чтобы сделать $\operatorname{res}[p_1(x), p_2(x)]$ целой функцией коэффициентов c_i, d_j. Очевидно, что

$$\operatorname{res}[p_1(x), p_2(x)] = (-1)^{mn} \operatorname{res}[p_2(x), p_1(x)],$$

и это выражение мы называем *результантом* полиномов $p_1(x)$ и $p_2(x)$. Поэтому нами доказана следующая

Теорема 5.2.2. Обращение в нуль результанта $\operatorname{res}[p_1(x), p_2(x)]$ — это необходимое и достаточное условие для того, чтобы уравнения $p_1(x) = 0$ и $p_2(x) = 0$ имели общий корень.

Некоторые свойства результантов. Если $p_2(x) = p_3(x)p_4(x)$, где $\deg[p_3(x)] = m_1$ и $\deg[p_4(x)] = m_2$, то из соотношения (R) вытекает

$$\begin{aligned}\operatorname{res}[p_1(x), p_3(x)p_4(x)] &= c_n^{m_1+m_2}[p_3(\alpha_1)p_4(\alpha_1)]\dots[(p_3(\alpha_n)p_4(\alpha_n)]\\ &= c_n^{m_1} p_3(\alpha_1)\dots p_3(\alpha_n) \cdot c_n^{m_2} p_4(\alpha_1)\dots p_4(\alpha_n).\end{aligned}$$

То есть

$$\operatorname{res}[p_1(x), p_3(x)p_4(x)] = \operatorname{res}[p_1(x), p_3(x)] \cdot \operatorname{res}[p_1(x), p_4(x)]. \qquad \text{(R1)}$$

Более того, если c — константа, то из (R) получаем

$$\operatorname{res}[p_1(x), p_2(x) + cp_1(x)] = c_n^m [p_2(\alpha_1) + cp_1(\alpha_1)] \dots [p_2(\alpha_n) + cp_1(\alpha_n)],$$

а поскольку $p_1(\alpha_i) = 0,\ i = 1, 2, \dots, n$, то

$$\operatorname{res}[p_1(x), p_2(x) + cp_1(x)] = \operatorname{res}[p_1(x), p_2(x)]. \qquad \text{(R2)}$$

Наконец, если $p_2(x) = x^m$, то все значения β_i равны нулю, и мы получаем специальный вид

$$\operatorname{res}[p_1(x), x^m] = (-1)^{mn}[p_1(0)]^m = (-1)^{mn} c_0^m. \qquad \text{(R3)}$$

Представление результанта. Мы не будем вычислять результанты по формуле (R), поскольку тогда нам потребовалось бы прежде вычислить корни полинома $p_1(x)$ или $p_2(x)$. Вместо этого мы представим результант как определитель, который нетрудно посчитать. Чтобы сделать это, рассмотрим следующую систему $(m + n)$ уравнений [мы все еще имеем дело с данными выше двумя полиномами $p_1(x)$ и $p_2(x)$ степеней n и m соответственно]:

$$\begin{aligned}
&[c_0 - p_1(x)] + c_1 x + \dots + c_n x^n &&= 0,\\
&[c_0 - p_1(x)]x + c_1 x^2 + \dots + c_n x^{n+1} &&= 0,\\
&\qquad\qquad \ddots\\
&[c_0 - p_1(x)]x^{m-1} + c_1 x^m + \dots + c_n x^{m+n-1} &&= 0,\\
d_0 \quad &+ d_1 x + \dots + d_m x^m &&= 0,\\
&d_0 x + d_1 x^2 + \dots + d_m x^{m+1} &&= 0,\\
&\qquad\qquad \ddots\\
&d_0 x^{n-1} + d_1 x^n + \dots + d_m x^{m+n-1} &&= 0.
\end{aligned}$$

Мы рассматриваем это как систему линейных уравнений от $(m + n)$ неизвестных $1, x, x^2, \dots, x^{m+n-1}$. Первые m уравнений в действительности являются тождествами, получаемыми из определения $p_1(x)$; последние n выполняются только для $x = \beta_1, \beta_2, \dots, \beta_m$. Переставляя столбцы и строки, мы получаем уравнение степени m от

$v = p_1(\beta_1), p_1(\beta_2), \ldots, p_1(\beta_m)$, которое может быть представлено определителем

$$\det \begin{bmatrix} c_n & c_{n-1} & \ldots & & c_0 - v & 0 & \ldots & 0 \\ 0 & c_n & c_{n-1} & \ldots & & c_0 - v & \ldots & 0 \\ & & & & \ddots & & & \\ 0 & 0 & \ldots & & c_n & c_{n-1} & \ldots & c_0 - v \\ d_m & d_{m-1} & \ldots & d_0 & 0 & 0 & \ldots & 0 \\ 0 & d_m & d_{m-1} & \ldots & d_0 & 0 & \ldots & 0 \\ & & & & \ddots & & & \\ 0 & 0 & \ldots & d_m & d_{m-1} & & \ldots & d_0 \end{bmatrix} = 0, \quad \left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} (m \text{ строк}) \quad \left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} (n \text{ строк})$$

и корни этого уравнения определяют величины $p_1(\beta_1), p_1(\beta_2), \ldots, p_1(\beta_m)$. Воспользовавшись тем, что если мы имеем уравнение

$$a_m v^m + a_{m-1} v^{m-1} + a_{m-2} v^{m-2} + \cdots + a_0 = 0,$$

то произведение всех его корней равно $(-1)^m a_0 / a_m$; мы видим, что если мы разделим наше уравнение степени m от v на коэффициент при v^m, то свободный член будет равен $(-1)^m p_1(\beta_1) p_1(\beta_2) \ldots p_1(\beta_m)$. Поскольку в нашем случае коэффициент при $(-v)^m$ равен $(-1)^{mn} d_m^n$ (проверьте!), получаем следующее соотношение (заметим, что мы положили $v = 0$, и следовательно, определитель — это свободный член уравнения степени m):

$$\det \begin{bmatrix} c_n & c_{n-1} & \ldots & & c_0 & 0 & \ldots & 0 \\ 0 & c_n & c_{n-1} & \ldots & & c_0 & \ldots & 0 \\ & & & & \ddots & & & \\ 0 & 0 & \ldots & & c_n & c_{n-1} & \ldots & c_0 \\ d_m & d_{m-1} & \ldots & d_0 & 0 & 0 & \ldots & 0 \\ 0 & d_m & d_{m-1} & \ldots & d_0 & 0 & \ldots & 0 \\ & & & & \ddots & & & \\ 0 & 0 & \ldots & d_m & d_{m-1} & & \ldots & d_0 \end{bmatrix} \quad \left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} (m \text{ строк}) \quad \left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} (n \text{ строк})$$

$$= (-1)^{mn} d_m^n p_1(\beta_1) p_1(\beta_2) \ldots p_1(\beta_m) = c_n^m p_2(\alpha_1) p_2(\alpha_2) \ldots p_2(\alpha_n)$$
$$= \mathrm{res}_{\mathrm{B}}[p_1(x), p_2(x)].$$

Таким образом мы выразили результант в виде определителя, который может быть легко вычислен; далее мы будем обозначать эту форму результанта через $\mathrm{res}_{\mathrm{B}}[p_1(x), p_2(x)]$ в честь итальянофранцузского математика девятнадцатого века Ди Брюно, который широко ею пользовался.

Субрезультанты — получение коэффициентов членов PRS. Введем определение субрезультантов, играющих важную роль в наших исследованиях PRS-алгоритмов. Рассмотрим *полную* последовательность полиномиальных остатков $p_1(x), p_2(x), p_3(x), \ldots, p_h(x)$, где

$$\begin{aligned} p_1(x) &= c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0, \quad c_n \neq 0, \\ p_2(x) &= d_{n-1}x^{n-1} + d_{n-2}x^{n-2} + \cdots + d_0, \quad d_{n-1} \neq 0. \end{aligned} \tag{BT}$$

Как упоминалось в доказательстве теоремы 5.2.1, коэффициенты полинома $p_3(x)$ степени $n-2$ задаются определителями

$$\det \begin{bmatrix} c_n & c_{n-1} & c_{n-2} \\ d_{n-1} & d_{n-2} & d_{n-3} \\ 0 & d_{n-1} & d_{n-2} \end{bmatrix}, \quad \det \begin{bmatrix} c_n & c_{n-1} & c_{n-3} \\ d_{n-1} & d_{n-2} & d_{n-4} \\ 0 & d_{n-1} & d_{n-3} \end{bmatrix} \text{ и т.д.},$$

где первый определитель — это старший коэффициент полинома $p_3(x)$, второй определитель — это коэффициент при x^{n-3} и т.д. По причинам, которые объясняются ниже, эти определители называются *субрезультантами*. [Мы знаем, что для последовательных членов PRS, $p_i(x)$, $i > 3$, каждый определитель делится на $\{\mathrm{lc}[p_{i-2}(x)]\}^2$.] Сравнивая эти определители с результантом полиномов $p_1(x)$ и $p_2(x)$, представленным в виде определителя порядка $2n-1$, мы легко видим, что коэффициенты полинома $p_3(x)$ получаются следующим образом. Поскольку степень полинома $p_3(x)$ равна $n-2$, удалим из матрицы $\mathbf{M}_B$, соответствующей $\mathrm{res_B}[p_1(x), p_2(x)]$, последние $n-2$ (из $n-1$) строк значений c, последние $n-2$ (из n) строк значений d и последние $n-2$ столбца. Таким образом, у нас осталась $3 \times (n+1)$-матрица $\mathbf{M}'_B$, т.е. $\mathbf{M}'_B$ имеет 3 строки и $n+1$ столбцов $\mathbf{M}'_B(j)$, $j = 0, 1, 2, \ldots, n$. Последовательно вычисляем $n-1$ коэффициентов полинома $p_3(x)$ из определителя (субрезультанта) порядка 3, первые два столбца которого всегда совпадают с $\mathbf{M}'_B(0), \mathbf{M}'_B(1)$, а третий последовательно равен $\mathbf{M}'_B(j)$, $j = 2, 3, \ldots, n$; при $j = 2$ получаем старший коэффициент полинома $p_3(x)$. [Это в точности то, что Сильвестр делал в теореме 5.2.1, чтобы вычислить коэффициенты *только* полинома $p_3(x)$.]

Этот процесс можно *обобщить* для вычисления коэффициентов любого члена PRS, $p_i(x)$, степени $n-i+1$. Мы просто должны удалить из матрицы $\mathbf{M}_B$, соответствующей $\mathrm{res_B}[p_1(x), p_2(x)]$, последние $n-i+1$ строк значений c, последние $n-i+1$ строк значений d и последние $n-i+1$ столбцов, оставив $(2i-3) \times (n+i-2)$-матрицу $\mathbf{M}'_B$ [$\mathbf{M}'_B$ имеет $(i-2)$ строк элементов c и $(i-1)$ строк элементов d]. Тогда

$n-i+2$ коэффициентов полинома $p_i(x)$ последовательно вычисляются из определителя (субрезультанта) порядка $(2i-3)$, первые $2(i-2)$ столбцов которого всегда равны $\mathbf{M}'_B(0), \mathbf{M}'_B(1), \ldots, \mathbf{M}'_B[2(i-2)-1]$, а последний последовательно равен $\mathbf{M}'_B(j)$, $j = 2(i-2), (2i-3), \ldots, n+i-3$. Этот общий способ вычисления коэффициентов любого члена полной PRS был открыт в 1859 г. Ди Брюно [см. (Muir, 1920, p. 327–328)], т.е. Ди Брюно обнаружил, что для полиномов (BT) $p_i(x)$, i-й член PRS, равен

$$\lambda_i \sum_{0 \le m \le n-i+1} x^{n-i+1-m} \cdot$$

$$\det \begin{bmatrix} c_n & c_{n-1} & & c_{n-2} & \ldots & c_{n-2(i-2)+1} & c_{n-2(i-2)-m} \\ 0 & c_n & & c_{n-1} & \ldots & c_{n-2(i-2)+2} & c_{n-2(i-2)-m+1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & c_n & \ldots & c_{n-i+2} & c_{n-i+1-m} \\ d_{n-1} & d_{n-2} & & d_{n-3} & \ldots & d_{n-2(i-2)} & d_{n-2(i-2)-1-m} \\ 0 & d_{n-1} & & d_{n-2} & \ldots & d_{n-2(i-2)+1} & d_{n-2(i-2)-1-m+1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & d_{n-1} & \ldots & d_{n-i+2} & d_{n-i+1-m} \end{bmatrix},$$

где λ_i — «распределенный» множитель, а $c_{-k} = 0$ для любого значения k.

Если последовательность полиномиальных остатков неполная или если разность степеней полиномов $p_1(x)$ и $p_2(x)$ больше единицы, то мы можем в основном пользоваться описанным выше подходом для вычисления коэффициентов любого члена PRS; однако, как уже упоминалось, распределенный множитель не так-то легко определить, и это будет обсуждаться ниже. Кроме того, если при вычислениях степень полинома $p_i(x)$ окажется меньше, чем $n-i+1$, например $n-i+1-k$, и мы вычисляем коэффициенты полинома $p_i(x)$, удаляя из матрицы $\mathbf{M}_B$, соответствующей $\mathrm{res_B}[p_1(x), p_2(x)]$, последние $n-i+1-k$ строк элементов c и d и последние $n-i+1-k$ столбцов, то мы получаем полином, *пропорциональный* полиному, который мы получили бы, удаляя последние $n-i+1$ строк элементов c и d и последние $n-i+1$ столбцов. Однако, удаляя из $\mathrm{res_B}[p_1(x), p_2(x)]$ последовательно последние $n-i, \ldots, n-i+1-(k-1)$ строк элементов c и d и последние $n-i, \ldots, n-i+1-(k-1)$ столбцов, *мы получаем нулевые полиномы*. Эти факты доказаны в разд. 5.2.3.

В нашем обсуждении теоремы 5.2.1 мы уже упоминали, что Сильвестр интересовался вычислением противоположных к полиномиальным остаткам (последовательностью Штурма). Так, для полиномов

(BT) он вычислял коэффициенты полинома $p_3(x)$ как

$$\det\begin{bmatrix} c_n & c_{n-1} & c_{n-2} \\ 0 & d_{n-1} & d_{n-2} \\ d_{n-1} & d_{n-2} & d_{n-3} \end{bmatrix}, \quad \det\begin{bmatrix} c_n & c_{n-1} & c_{n-3} \\ 0 & d_{n-1} & d_{n-3} \\ d_{n-1} & d_{n-2} & d_{n-4} \end{bmatrix} \text{ и т.д.}$$

Чтобы иметь возможность получать противоположные к коэффициентам любого члена PRS [для полиномов (BT)] способом, подобным описанному выше, мы должны пользоваться другой формой результанта, а именно

$$\mathrm{res}_{\mathrm{T}}[p_1(x), p_2(x)] = \det\begin{bmatrix} c_n & c_{n-1} & \dots & & c_0 & 0 & \dots & 0 \\ 0 & c_n & c_{n-1} & \dots & & c_0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \ddots & \vdots & \ddots & \vdots \\ .....0 & 0 & \dots & & c_n & c_{n-1} & \dots & c_0 \\ 0 & 0 & \dots & & & d_{n-1} & \dots & d_0 \\ \vdots & \vdots & \vdots & & \ddots & \vdots & \ddots & \vdots \\ .....0 & d_{n-1} & \dots & & d_0 & 0 & \dots & 0 \\ d_{n-1} & & \dots & d_0 & 0 & 0 & \dots & 0 \end{bmatrix}.$$

Индекс T стоит в честь итальянского математика прошлого века H. Труди, давшего первое систематическое изложение *биградиентов* [см. (Muir, 1902, p. 329)]; термином *биградиент* обозначаем форму результанта, обозначенную нами $\mathrm{res}_{\mathrm{T}}[p_1(x), p_2(x)]$.

Теперь, чтобы вычислить коэффициенты любого члена $p_i(x)$ полной PRS, где $\deg[p_i(x)] = n - i + 1$, мы удаляем из матрицы $\mathbf{M}_T$, соответствующей $\mathrm{res}_{\mathrm{T}}[p_1(x), p_2(x)]$, *верхние* $n - i + 1$ строк элементов c, последние $n - i + 1$ строк элементов d и *первые* $n - i + 1$ столбцов. При этом остается $(2i - 3) \times (n + i - 2)$-матрица $\mathbf{M}'_T$, с которой мы работаем так же, как в случае $\mathrm{res}_{\mathrm{B}}[p_1(x), p_2(x)]$.

Отметим, что

$$\mathrm{res}_{\mathrm{B}}[p_1(x), p_2(x)] = (-1)^{n(n-1)/2}\,\mathrm{res}_{\mathrm{T}}[p_1(x), p_2(x)].$$

Представим теперь теорему Труди в ее первоначальном виде и обозначениях.

Теорема 5.2.3 (Trudi, 1862). Коэффициенты r-го остатка R, возникающие в ходе выполнения процесса деления Штурма для полиномов

$$a_0 x^m + \cdots + a_m, \quad b_0 x^n + \cdots + b_n$$

(т.е. мы вычисляем противоположные для остатков, полученных евклидовым алгоритмом PRS), равны последовательным определителям массива (биградиента)

$$\left[\left[\begin{matrix}(a_0, a_1, \ldots, a_m)_r \\ (b_0, b_1, \ldots, b_n)\end{matrix}\right]\right],$$

разделенным на произведение квадратов первых коэффициентов всех предыдущих остатков, на b_0^{m-n+1} и на знаковый множитель $(-1)^{(m-n)(m-n-1)/2}$.

Доказательство. Доказательство можно найти в книге (Householder, 1970). □

Пример. Для полиномов $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$ мы имеем две следующие матрицы для соответствующих результантов:

$$\mathbf{M}_B = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 7 \\ 3 & 0 & -7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 \\ 0 & 0 & 3 & 0 & -7 \end{bmatrix}, \quad \mathbf{M}_T = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 7 \\ 0 & 0 & 3 & 0 & -7 \\ 0 & 3 & 0 & -7 & 0 \\ 3 & 0 & -7 & 0 & 0 \end{bmatrix}.$$

Коэффициенты полинома $p_3(x)$ степени 1 получаются следующим образом.

Используя $\mathbf{M}_B$:

$$\det \begin{bmatrix} 1 & 0 & -7 \\ 3 & 0 & -7 \\ 0 & 3 & 0 \end{bmatrix} = -42, \quad \det \begin{bmatrix} 1 & 0 & -7 \\ 3 & 0 & 0 \\ 0 & 3 & -7 \end{bmatrix} = 63$$

и, таким образом, $p_3(x) = -42x + 63$.

Используя $\mathbf{M}_T$:

$$\det \begin{bmatrix} 1 & 0 & -7 \\ 0 & 3 & 0 \\ 3 & 0 & -7 \end{bmatrix} = 42, \quad \det \begin{bmatrix} 1 & 0 & 7 \\ 0 & 3 & -7 \\ 3 & 0 & 0 \end{bmatrix} = -63$$

и $p_3(x) = -42x + 63$.

Что касается полинома $p_4(x)$, являющегося константой, то мы просто вычисляем результант и получаем $\mathrm{res_B}[p_1(x), p_2(x)] = -49$ и $\mathrm{res_T}[p_1(x), p_2(x)] = 49$. Читатель может сравнить эти результаты с результатами, полученными в примере разд. 5.2.1. (Заметим, что в этом случае, чтобы получить значение ± 49, определитель не нужно ни на что делить.)

Дополнительное свойство результантов. Мы представим сейчас теорему, утверждающую, что результант двух полиномов может быть выражен через эти полиномы.

Теорема 5.2.4. Пусть J — коммутативное кольцо с единицей, а $p(x)$ и $q(x)$ — полиномы в кольце $J[x]$ положительной степени. Тогда в $J[x]$ существуют полиномы $r(x)$ и $s(x)$, такие, что $p(x)r(x) + q(x)s(x) = \mathrm{res}_{\mathrm{B}}[p(x), q(x)]$, причем $\deg[r(x)] < \deg[q(x)]$ и $\deg[s(x)] < \deg[p(x)]$. (Очевидно, что подобный результат имеет место и для $\mathrm{res}_{\mathrm{T}}[p(x), q(x)]$.)

Доказательство. Пусть $m = \deg[p(x)]$ и $n = \deg[q(x)]$. Для $1 \leq i < m + n$ домножим i-й столбец матрицы $\mathbf{M}_B$ на x^{m+n-i} и прибавим его к последнему столбцу. В результате получим новую матрицу $\mathbf{M}'_B$, определитель которой равен определителю матрицы $\mathbf{M}_B$. Последний столбец матрицы $\mathbf{M}'_B$ состоит из полиномов $x^{n-1}p(x), x^{n-2}p(x), \ldots, p(x), x^{m-1}q(x), x^{m-2}q(x), \ldots, q(x)$. Разлагая определитель по последнему столбцу, получаем тождество $p(x)r(x) + q(x)s(x) = \det(\mathbf{M}'_B) = \mathrm{res}_{\mathrm{B}}[p(x), q(x)]$, в котором коэффициенты полиномов $r(x)$ и $s(x)$ являются алгебраическими дополнениями элементов последнего столбца матрицы $\mathbf{M}'_B$, а следовательно, и $\mathbf{M}$ и поэтому принадлежат J. □

Эта теорема напоминает нам о соотношении, существующем между двумя полиномами и их наибольшим общим делителем. Для более подробного исследования снова рассмотрим два полинома в $\mathbb{Z}[x]$:

$$p(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0, \quad c_n \neq 0,\ n > 0,$$
$$q(x) = d_m x^m + d_{m-1}x^{m-1} + \cdots + d_0, \quad d_m \neq 0,\ m > 0,$$

и *предположим, что у них есть общий корень* α. Тогда

$$p(x) = (x - \alpha)p_1(x), \quad q(x) = (x - \alpha)q_1(x),$$

где $p_1(x), q_1(x)$ в $\mathbb{Z}[x]$, $\deg[p_1(x)] = n - 1,\ \deg[q_1(x)] = m - 1$.

Из теорем 5.2.2 и 5.2.4 легко следует, что

$$p(x)q_1(x) - q(x)p_1(x) = 0. \tag{CR}$$

Обратно, существование соотношения выписанного вида, в котором $p_1(x), q_1(x)$ лежат в $\mathbb{Z}[x]$, $\deg[p_1(x)] = n - 1,\ \deg[q_1(x)] = m - 1$, — это доказательство существования общего корня у полиномов $p(x)$ и $q(x)$. [Это так, поскольку $q_1(x)$ не может делиться на все линейные

сомножители $(x - \beta_\lambda)$ полинома $q(x)$, а поэтому существует по крайней мере один линейный сомножитель полинома $q(x)$, появляющийся также и в $p(x)$.]

Пусть теперь

$$p_1(x) = u_{n-1}x^{n-1} + u_{n-2}x^{n-2} + \cdots + u_0, \quad u_{n-1} \neq 0,$$
$$q_1(x) = v_{m-1}x^{m-1} + v_{m-2}x^{m-2} + \cdots + v_0, \quad v_{m-1} \neq 0,$$

и заменим в (CR) все полиномы соответствующими их выражениями. Тогда, группируя все коэффициенты при равных степенях x и полагая все их равными 0, мы получаем систему уравнений, выписанную на стр. 321 (где мы предполагаем $n \geq m$).

Это система $m + n$ линейных уравнений от $m + n$ неизвестных $u_{n-1}, \ldots, u_0, v_{m-1}, \ldots, v_0$. Существование общего корня означает, что у этой системы уравнений имеется ненулевое решение, следовательно, ее определитель должен равняться нулю. Переставляя строки и столбцы, видим, что выписанный определитель является результантом полиномов $p(x)$ и $q(x)$, $\mathrm{res_B}[p(x), q(x)]$.

Этот процесс можно продолжить. Если степени ненулевых полиномов $p_1(x)$ и $q_1(x)$ в (CR) равны $(n-2)$ и $(m-2)$ соответственно, то полиномы $p(x)$ и $q(x)$ имеют общий сомножитель второй степени, так что (CR) становится необходимым и достаточным условием существования двух общих корней полиномов $p(x)$ и $q(x)$. Доказательство проходит аналогично приведенному; нам только нужно положить $u_{n-1} = v_{m-1} = 0$ как в выражениях для $p_1(x), q_1(x)$, так и в соответствующей системе уравнений. Теперь в системе $(m+n-1)$ уравнений от $(m+n-2)$ неизвестных, и для того, чтобы существовало ненулевое решение, все определители порядка $(m+n-2)$, полученные из матрицы

$$\det \begin{bmatrix} c_n & c_{n-1} & \ldots & c_0 & 0 & \ldots \\ 0 & c_n & \ldots & c_1 & c_0 & \\ & & \ddots & & & \\ d_m & d_{m-1} & \ldots & d_0 & 0 & \ldots \\ 0 & d_m & \ldots & d_0 & 0 & \ldots \\ & & \ddots & & & \end{bmatrix}, \quad \begin{matrix} \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} (m-1) \text{ строк} \\ \left.\vphantom{\begin{matrix}a\\a\\a\end{matrix}}\right\} (n-1) \text{ строк} \end{matrix}$$

должны обращаться в нуль; в этой матрице $(m+n-1)$ столбцов. Очевидно, что мы можем продолжить этот процесс.

Существование наибольшего общего делителя заданной степени. Воспользуемся полученными результатами, чтобы найти необходимые и достаточные условия существования наибольшего общего делителя заданной степени полиномов $p(x)$ и $q(x)$. Читателю следует тщательно изучить доказательство следующей теоремы.

Степень				
$n+m-1$	$c_n v_{m-1}$		$-d_m u_{n-1}$	
$n+m-2$	$c_{n-1} v_{m-1}$	$+c_n v_{m-2}$	$-d_{m-1} u_{n-1} - d_m u_{n-2}$	
$\vdots$	$\vdots$		$\vdots$	
$n-1$	$c_{n-m} v_{m-1}$	$+c_{n-m+1} v_{m-2} + \ldots + c_{n-1} v_0$	$-d_0 u_{n-1} - d_1 u_{n-2} - \ldots - d_m u_{n-m-1}$	
$n-2$	$c_{n-m-1} v_{m-1}$	$+c_{n-m} v_{m-2} + \ldots + c_{n-2} v_0$	$-d_0 u_{n-2} - \ldots - d_m u_{n-m-2}$	
$\vdots$	$\vdots$		$\ddots$	
$m-1$	$c_0 v_{m-1}$	$+c_1 v_{m-2} + \ldots + c_{m-1} v_0$	$-d_0 u_{m-1} - \ldots - d_{m-1} u$	
$m-2$		$c_0 v_{m-2} + \ldots + c_{m-2} v_0$	$-d_0 u_{m-2} - \ldots - d_{m-2} u$	
$\vdots$		$\ddots$	$\ddots$	
0		$c_0 v_0$		$-d_0 u$

Теорема 5.2.5. Степень наибольшего общего делителя двух полиномов $p(x)$ и $q(x)$, $\deg[p(x)] = n$, $\deg[q(x)] = m$, из кольца $\mathbb{Z}[x]$ равна индексу первого ненулевого определителя среди

$$\mathrm{res}_{\mathrm{B}}[p(x), q(x)] = \mathrm{res}_{\mathrm{B}}^{(0)}[p(x), q(x)], \mathrm{res}_{\mathrm{B}}^{(1)}[p(x), q(x)], \mathrm{res}_{\mathrm{B}}^{(2)}[p(x), q(x)], \ldots,$$

где $\mathrm{res}_{\mathrm{B}}^{(i)}[p(x), q(x)]$ — субрезультант, получающийся из результанта $\mathrm{res}_{\mathrm{B}}[p(x), q(x)]$, если мы удалим последние i строк коэффициентов как $p(x)$, так и $q(x)$, и последние $2i$ столбцов, т.е. $\mathrm{res}_{\mathrm{B}}^{(i)}[p(x), q(x)]$ — это определитель порядка $(m + n - 2i)$ и называется (как мы убедимся, по очевидным причинам) i-м *главным субрезультантным коэффициентом*.

Доказательство. Выведем сначала условия, необходимые для того, чтобы полиномы $p(x)$ и $q(x)$ имели наибольший общий делитель степени 1. Пусть

$$\begin{aligned} p(x) &= c_n x^n + c_{n-1} x^{n-1} + \cdots + c_0, \quad c_n \neq 0,\ n > 0, \\ q(x) &= d_m x^m + d_{m-1} x^{m-1} + \cdots + d_0, \quad d_m \neq 0,\ m > 0. \end{aligned}$$

Из предыдущих рассуждений нам известно, что если $\mathrm{res}_{\mathrm{B}}[p(x), q(x)] = 0$, то $p(x)$ и $q(x)$ имеют делитель (по крайней мере) первой степени. Предположим, что этот случай действительно имеет место, и рассмотрим полиномы

$$\begin{aligned} p^{(1)}(x) &= v_{m-2} x^{m-2} + \cdots + v_0, \quad v_{m-2} \neq 0, \\ q^{(1)}(x) &= u_{n-2} x^{n-2} + \cdots + u_0, \quad u_{n-2} \neq 0, \end{aligned}$$

с неизвестными коэффициентами $u_{n-2}, \ldots, u_0, v_{m-2}, \ldots, v_0$. Тогда из уравнения

$$p(x)p^{(1)}(x) - q(x)q^{(1)}(x) = g_{m+n-2} x^{m+n-2} + g_{m+n-3} x^{m+n-3} + \cdots + g_0$$

получаем следующую систему уравнений:

$$\begin{aligned} c_n v_{m-2} - d_m u_{n-2} &= g_{m+n-2}, \\ c_{n-1} v_{m-2} + c_n v_{m-3} - d_{m-1} u_{n-2} - d_m u_{n-3} &= g_{m+n-3}, \\ &\vdots \\ c_0 v_1 + c_1 v_0 - d_0 u_1 - d_1 u_0 &= g_1, \\ c_0 v_0 - d_0 u_0 &= g_0. \end{aligned}$$

Не считая последнего уравнения, мы имеем $(m+n-2)$ линейных уравнений от $(m+n-2)$ неизвестных. Поэтому если соответствующий определитель $\mathrm{res}_{\mathrm{B}}^{(1)}[p(x),q(x)]$ порядка $m+n-2$ не равен нулю, то мы можем определить неизвестные так, чтобы

$$g_{m+n-2}=g_{m+n-3}=\cdots=g_2=0 \text{ и } g_1=\mathrm{res}_{\mathrm{B}}^{(1)}[p(x),q(x)].$$

В этом случае

$$p(x)p^{(1)}(x)-q(x)q^{(1)}(x)=\mathrm{res}_{\mathrm{B}}^{(1)}[p(x),q(x)]\cdot x+g_0.$$

Последнее соотношение означает, что $p(x)$ и $q(x)$ имеют общий делитель только первой степени (в противном случае правая часть выписанного уравнения будет делиться также на этот другой сомножитель более высокой степени). Таким образом,

$$\mathrm{res}_{\mathrm{B}}^{(0)}[p(x),q(x)]=0,\quad \mathrm{res}_{\mathrm{B}}^{(1)}[p(x),q(x)]\neq 0$$

суть необходимые и достаточные условия, чтобы степень наибольшего общего делителя полиномов $p(x)$ и $q(x)$ равнялась 1.

Если, однако, $\mathrm{res}_{\mathrm{B}}^{(1)}[p(x),q(x)]=0$, то

$$p(x)p^{(1)}(x)-q(x)q^{(1)}(x)=g_0,$$

но теперь, поскольку $p(x)$ и $q(x)$ имеют общий множитель, существует значение x, для которого левая часть последнего соотношения обращается в нуль, а следовательно, должен иметь место случай $g_0=0$. Таким образом,

$$p(x)p^{(1)}(x)-q(x)q^{(1)}(x)=0,$$

откуда мы заключаем, что полиномы $p(x)$ и $q(x)$ имеют общий множитель по крайней мере второй степени.

Поэтому если $\mathrm{res}_{\mathrm{B}}^{(0)}[p(x),q(x)]=\mathrm{res}_{\mathrm{B}}^{(1)}[p(x),q(x)]=0$, то мы полагаем

$$\begin{aligned}p^{(2)}(x)&=v_{m-3}x^{m-3}+\cdots+v_0,\quad v_{m-3}\neq 0,\\ q^{(2)}(x)&=u_{n-3}x^{n-3}+\cdots+u_0,\quad u_{n-3}\neq 0,\end{aligned}$$

с неизвестными коэффициентами $u_{n-3},\ldots,u_0,v_{m-3},\ldots,v_0$, и снова из уравнения

$$p(x)p^{(2)}(x)-q(x)q^{(2)}(x)=g_{m+n-3}x^{m+n-3}+g_{m+n-4}x^{m+n-4}+\cdots+g_0$$

мы получаем следующую систему уравнений:

$$\begin{aligned} c_n v_{m-3} - d_m u_{n-3} &= g_{m+n-3}, \\ c_{n-1} v_{m-3} + c_n v_{m-4} - d_{m-1} u_{n-3} - d_m u_{n-4} &= g_{m+n-4}, \\ &\vdots \end{aligned}$$

Не считая *двух* последних строк, мы имеем систему $(m+n-4)$ линейных уравнений от $(m+n-4)$ неизвестных. Если соответствующий определитель

$$\operatorname{res}_{\mathrm{B}}^{(2)}[p(x), q(x)] = \det \begin{bmatrix} c_n & c_{n-1} & \dots & c_0 & 0 & \dots \\ 0 & c_n & \dots & c_1 & c_0 & \\ & & \ddots & & & \\ d_m & d_{m-1} & \dots & d_0 & 0 & \dots \\ 0 & d_m & \dots & d_1 & d_0 & \dots \\ & & \ddots & & & \end{bmatrix}$$

не равен нулю, то мы можем определить неизвестные так, чтобы

$$g_{m+n-3} = g_{m+n-4} = \dots = 0, \quad g_2 = res_{\mathrm{B}}^{(2)}[p(x), q(x)].$$

В этом случае

$$p(x)p^{(2)}(x) - q(x)q^{(2)}(x) = \operatorname{res}_{\mathrm{B}}^{(2)}[p(x), q(x)] \cdot x^2 + g_1 x + g_0.$$

Это означает, что $p(x)$ и $q(x)$ имеют общий делитель только степени 2 (если существует делитель более высокой степени, то он должен делить также правую часть выписанного соотношения).

Поэтому

$$\operatorname{res}_{\mathrm{B}}^{(0)}[p(x), q(x)] = \operatorname{res}_{\mathrm{B}}^{(1)}[p(x), q(x)] = 0, \quad \operatorname{res}_{\mathrm{B}}^{(2)}[p(x), q(x)] \neq 0$$

суть необходимые и достаточные условия, чтобы степень наибольшего общего делителя полиномов $p(x)$ и $q(x)$ равнялась 2. Продолжая таким же образом, мы закончим доказательство теоремы. □

В качестве следствия теоремы 5.2.5 мы получаем

Следствие 5.2.6. Необходимым и достаточным условием взаимной простоты двух полиномов от одной переменной является необращение в нуль их результанта.

Чрезвычайно важное значение для материала, излагаемого в следующем разделе, имеет i-й главный субрезультантный коэффициент $\mathrm{res}_{\mathrm{B}}^{(i)}[p(x), q(x)]$, кратко обозначаемый там R_i; см. также приведенный ниже пример.

Результаты, аналогичные полученным выше, могут быть получены для $\mathrm{res}_{\mathrm{T}}[p(x), q(x)]$, где используются те же обозначения (причем различие знаков игнорируется), а именно имеет место

Определение 5.2.7. i-й главный субрезультантный коэффициент $\mathrm{res}_{\mathrm{T}}^{(i)}[p_1(x), p_2(x)]$ двух полиномов $p_1(x), p_2(x)$ от одной переменной — это определитель, получающийся из результанта $\mathrm{res}_{\mathrm{T}}[p_1(x), p_2(x)]$ этих полиномов, если отбросить первые i строк и столбцов и последние i строк и столбцов.

Пример. Для полиномов $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$ мы получаем следующие субрезультанты $R_i = \mathrm{res}_T^{(i)}[p_1(x), p_2(x)]$, $i = 0, 1, 2$:

$$R_0 = \begin{vmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 0 \\ 0 & 0 & 3 & 0 & -7 \\ 0 & 3 & 0 & -7 & 0 \\ 3 & 0 & -7 & 0 & 0 \end{vmatrix}, \quad R_1 = \begin{vmatrix} 1 & 0 & -7 & 0 \\ 0 & 3 & 0 & -7 \\ 0 & 0 & 3 & 0 \\ 3 & 0 & -7 & 0 \end{vmatrix}, \quad R_2 = \begin{vmatrix} 3 & 0 \\ 0 & 3 \end{vmatrix}$$

Теорема 5.2.8. Степень наибольшего общего делителя двух полиномов от одной переменной равна индексу первого ненулевого главного субрезультантного коэффициента $R_0 = \mathrm{res}_{\mathrm{T}}^{(0)}[p_1(x), p_2(x)]$, $R_1 = \mathrm{res}_{\mathrm{T}}^{(1)}[p_1(x), p_2(x)]$, $R_2 = \mathrm{res}_{\mathrm{T}}^{(2)}[p_1(x), p_2(x)], \ldots$.

Доказательство аналогично доказательству теоремы 5.2.5. □

В последнем примере мы находим, что $R_0 = \mathrm{res}_{\mathrm{T}}^{(0)}[p_1(x), p_2(x)]$, откуда заключаем, что степень наибольшего общего делителя полиномов $p_1(x)$ и $p_2(x)$ равна 0, т.е. $\gcd[p_1(x), p_2(x)] = 1$.

Имеет место также следующая

Теорема 5.2.9. Если h-й главный субрезультантный коэффициент полиномов $p_1(x)$ и $p_2(x)$ является первым ненулевым [т.е. $\mathrm{res}_{\mathrm{T}}^{(0)}[p_1(x), p_2(x)] = \mathrm{res}_{\mathrm{T}}^{(1)}[p_1(x), p_2(x)] = \cdots = \mathrm{res}_{\mathrm{T}}^{(h-1)}[p_1(x), p_2(x)] =$

0, а $\operatorname{res}_{\mathrm{T}}^{(h)}[p_1(x), p_2(x)] \neq 0]$, то наибольший общий делитель полиномов $p_1(x)$ и $p_2(x)$ может быть получен из $\operatorname{res}_{\mathrm{T}}^{(h)}[p_1(x), p_2(x)]$, если мы заменим (**a**) на $p_1(x)$ последний элемент в последней строке коэффициентов полинома $p_1(x)$, на $xp_1(x)$ — элемент, находящийся непосредственно над ним, на $x^2p_1(x)$ — элемент над этим и т.д., (**b**) на $p_2(x)$ — последний элемент в первой строке коэффициентов полинома $p_2(x)$, на $xp_2(x)$ — элемент под ним, на $x^2p_2(x)$ — элемент под этим и т.д.

Доказательство. Доказательство основано на теореме 5.2.4, и мы оставляем его читателю в качестве упражнения. □

Обобщая теорему 5.2.9, мы видим, что точно таким же образом, как описано в теореме 5.2.9, мы можем получить i-й член последовательности полиномиальных остатков из $\operatorname{res}_{\mathrm{T}}^{(n-i+1)}[p_1(x), p_2(x)]$; пример приведен ниже. Те же результаты имеют место для $\operatorname{res}_{\mathrm{B}}[p_1(x), p_2(x)]$.

Отметим также, что $|R_i| = |\operatorname{res}_{\mathrm{T}}^{(i)}[p_1(x), p_2(x)]| = |\operatorname{res}_{\mathrm{B}}^{(i)}[p_1(x), p_2(x)]|$, а потому, игнорируя различие в знаках, мы будем в обоих случаях использовать обозначение R_i.

Пример. Мы уже знаем, что $R_0 = \operatorname{res}_{\mathrm{T}}^{(0)}[p_1(x), p_2(x)] \neq 0$ для полиномов $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$; следовательно, $\deg[\gcd[p_1(x), p_2(x)]) = 0$; наибольший общий делитель получается из $\operatorname{res}_{\mathrm{T}}^{(0)}[p_1(x), p_2(x)]$ вычислением определителя

$$\det \begin{bmatrix} 1 & 0 & -7 & 7 & xp_1(x) \\ 0 & 1 & 0 & -7 & p_1(x) \\ 0 & 0 & 3 & 0 & p_2(x) \\ 0 & 3 & 0 & -7 & xp_2(x) \\ 3 & 0 & -7 & 0 & x^2p_2(x) \end{bmatrix} = 49 = p_4(x).$$

Подобным же образом мы видим, что $p_3(x)$, третий член PRS, получается из $\operatorname{res}_{\mathrm{T}}^{(1)}[p_1(x), p_2(x)]$:

$$\begin{aligned} \det \begin{bmatrix} 1 & 0 & p_1(x) \\ 0 & 3 & p_2(x) \\ 3 & 0 & xp_2(x) \end{bmatrix} &= -9p_1(x) + 3xp_2(x) \\ &\quad -9x^3 + 63x - 63 + 9x^3 - 21x \\ &= 42x - 63 = p_3(x). \end{aligned}$$

(Сравните это с результатами, полученными в примере, предшествующем теореме 5.2.4.)

В заключение этого раздела дадим следующее

Определение 5.2.10. Дискриминант $\mathrm{discr}[p(x)]$ полинома $p(x) = a\prod_{1\le i\le n}(x-\alpha_i)$ определяется следующей формулой:

$$\mathrm{discr}[p(x)] = a^{2n-2}\prod_{1\le i\le n}\prod_{i<j\le n}(\alpha_i-\alpha_j)^2.$$

Отметим, что дискриминант дает меру близости корней полинома. Это будет использоваться в методе изоляции корней. Более того, можно доказать, что для нормированного полинома $p(x)$ степени n

$$\mathrm{discr}[p(x)] = (-1)^{n(n-1)/2}\,\mathrm{res}_{\mathrm{B}}[p(x),p'(x)],$$

где $p'(x)$ — производная полинома $p(x)$. Дополнительные свойства дискриминанта можно найти в литературе (Berlecamp, 1968).

5.2.3. Алгоритм Габихта субрезультантных PRS

Как мы видели, Сильвестр указал отношение делимости, существующее между *некоторыми* членами полной последовательности полиномиальных остатков, и это явилось основанием для алгоритма редуцированных (субрезультантных) PRS. Однако, для того чтобы иметь дело с *неполными* PRS, нам требуется отношение делимости, существующее между *любыми* членами PRS. Это новое отношение делимости было сформулировано Габихтом в 1948 г. и составляет предмет данного раздела. [Удивительно простое доказательство см. в (Gonzalez et al., 1990).]

Чтобы доказать теорему Габихта, рассмотрим два полинома в $\mathbb{Z}[x]$

$$p_{n+1}(x) = c_0x^{n+1} + c_1x^n + \cdots + c_{n+1},$$
$$p_n(x) = d_0x^n + d_1x^{n-1} + \cdots + d_n$$

степеней $n+1$ и n соответственно и последовательность полиномиальных остатков $p_{n-1}(x),\ldots,p_1(x),p_0(x)$, *где индекс указывает степень соответствующего полинома*. Отметим, что Габихт интересовался в первую очередь последовательностями Штурма, и это объясняет выбор степеней первых двух полиномов. Многие из представленных ниже результатов справедливы при произвольных степенях двух первых полиномов (читатель легко может определить, когда это имеет место), но мы придерживаемся подхода Габихта.

Как мы напоминали, для полных PRS Сильвестр доказал, что $(i-2)$-й член PRS делится на квадрат i-го главного субрезультантного коэффициента:

$$\{\mathrm{res}_{\mathrm{B}}^{(i)}[p_{n+1}(x), p_n(x)]\}^2 | p_{i-2}(x). \tag{S0}$$

Другими словами, квадрат старшего коэффициента полинома $p_i(x)$ делит без остатка полином $p_{i-2}(x)$ [$\mathrm{res}_{\mathrm{B}}^{(i)}[p_{n+1}(x), p-n(x)]$ был определен в теореме 5.2.5]. Для неполных PRS, когда некоторые члены PRS отсутствуют, Габихт доказал, что

$$\{\mathrm{res}_{\mathrm{B}}^{(i+1)}[p_{n+1}(x), p_n(x)]\}^{2(i-j)} \cdot p_j(x) = p_{i+1}(x) \cdot p^{(ij)}(x) + p_i(x) \cdot q^{(ij)}(x),$$
$$0 < i < n, \quad 0 \le j < i, \tag{H0}$$

где полиномы $p^{(ij)}(x), q^{(ij)}(x)$ получены из коэффициентов полиномов $p_{i+1}(x)$ и $p_i(x)$ таким же образом (объясненным ниже), как полиномы $p^{(i)}(x), q^{(i)}(x)$ (определенные ниже в теореме 5.2.11) получены из коэффициентов полиномов $p_{n+1}(x)$ и $p_n(x)$.

Замечание. Читатель должен хорошо разобраться в различии между двумя парами полиномов с верхними индексами, т.е. $p^{(ij)}(x), q^{(ij)}(x)$ и $p^{(i)}(x), q^{(i)}(x)$; роль индекса i в них совершенно разная. Для согласованности последняя пара может также обозначаться $\{p^{(ni)}(x), q^{(ni)}(x)\}$, но мы пользуемся более простым обозначением.

В полном объеме смысл и значение соотношения (H0) будет разбираться позже. Заметим, что для неполных PRS с помощью $p^{(ij)}(x)$ и $q^{(ij)}(x)$ мы приходим к новым свойствам делимости, сформулированным в конце раздела, которые приводят к алгоритму Габихта субрезультантных PRS, упоминавшемуся в разд. 5.1.1.

Вначале будет доказана

Теорема 5.2.11. Рассмотрим последовательность полиномиальных остатков $p_{n+1}(x), p_n(x), p_{n-1}(x), \ldots, p_1(x), p_0(x)$ над целыми числами, в которой индекс обозначает степень соответствующего полинома; тогда для любого члена $p_i(x)$ этой PRS, $0 \le i < n$, мы можем найти в $\mathbb{Z}[x]$ два полинома

$$p^{(i)}(x) = a_0 x^{n-i-1} + a_1 x^{n-i-2} + \cdots + a_{n-i-1},$$
$$q^{(i)}(x) = b_0 x^{n-i} + b_1 x^{n-i-1} + \cdots + b_{n-i}$$

степеней $n-i-1$ и $n-i$ соответственно, таких, что

$$p_i(x) = p_{n+1}(x)p^{(i)}(x) + p_n(x)q^{(i)}(x).$$

Доказательство. Доказательство является конструктивным и аналогично расширенному алгоритму Евклида для целых чисел или полиномов над полем, т.е. мы требуем соблюдения соотношения $p_i(x) = p_{n+1}(x)p^{(i)}(x) + p_n(x)q^{(i)}(x)$ для любого i.

Очевидно, что $p^{(i)}(x) = 1$ и $q^{(i)}(x) = 0$ при $i = n+1$ и $p^{(i)}(x) = 0$ и $q^{(i)}(x) = 1$ при $i = n$. Как мы уже видели, чтобы вычислить $p^{(i)}(x)$ и $q^{(i)}(x)$ для $0 \leq i < n$, мы изменяем их таким же образом, как изменяем значения $p_i(x)$, а именно, если

$$p_{i-1}(x) = \{\mathrm{lc}[p_i(x)]\}^2 p_{i+1}(x) - p_i(x)q(x),$$

где $q(x)$ — частное от деления $\{\mathrm{lc}[p_i(x)]\}^2 p_{i+1}(x)$ на $p_i(x)$, то

$$p^{(i-1)}(x) = \{\mathrm{lc}[p_i(x)]\}^2 p^{(i+1)}(x) - p^{(i)}(x)q(x),$$
$$q^{(i-1)}(x) = \{\mathrm{lc}[p_i(x)]\}^2 q^{(i+1)}(x) - q^{(i)}(x)q(x). \qquad \square$$

Теорема 5.2.11 показывает, что расширенный алгоритм Евклида работает также с полиномами над целыми числами. [Если мы сравним выражения $p_i(x) = p_{n+1}(x)p^{(i)}(x) + p_n(x)q^{(i)}(x)$ в теореме 5.2.11 с выражениями вида

$$p(x)p^{(i)}(x) - q(x)q^{(i)}(x) = \mathrm{res}_{\mathrm{B}}^{(i)}[p(x), q(x)]x^i + \cdots + g_1 x + g_0,$$

полученными в доказательстве теоремы 5.2.5, мы увидим, что старший коэффициент полинома $p_i(x)$, $0 \leq i < n$ (где $p_i(x)$ — произвольный член PRS), является субрезультантом $\mathrm{res}_{\mathrm{B}}^{(i)}[p_{n+1}(x), p_n(x)]$, и поэтому он называется *главным субрезультантным коэффициентом.*]

Мы видели, что произвольный член PRS, $p_i(x)$, может быть вычислен из соответствующего главного субрезультантного коэффициента, как описано в теореме 5.2.9; подобным образом мы видим, что полиномы $p^{(i)}(x)$ и $q^{(i)}(x)$ в выражении $p_i(x) = p_{n+1}(x)p^{(i)}(x) + p_n(x)q^{(i)}(x)$ также могут быть получены из главного субрезультантного коэффициента следующим образом:

Определитель для $p^{(i)}(x)$ в точности совпадает с определителем для $p_i(x)$, кроме последнего столбца, который равен x^{n-i-1}, $\ldots, 1, 0, \ldots, 0$ (начиная сверху).

Определитель для $q^{(i)}(x)$ в точности совпадает с определителем для $p_i(x)$, кроме последнего столбца, который равен $0, \ldots, 0$, $x^{n-i-1}, \ldots, 1$ (начиная сверху).

[Итак, если нам нужны последовательности Штурма и мы пользуемся формой Брюно для определителя, то во всех определителях присутствует сомножитель $(-1)^{n(n+1)/2}$; в этом состоит различие между формами Брюно и Труди для результанта. Ниже мы этот сомножитель будем игнорировать.]

Таким образом, полиномы $p^{(i)}(x)$ и $q^{(i)}(x)$ могут быть непосредственно получены из следующих определителей порядка $2(n-i)+1$, где мы меняем местами строки и столбцы:

$$p^{(i)}(x) = \det \begin{bmatrix} x^{n-i-1} & x^{n-i-2} & \dots & 1 & 0 & 0 & \dots & 0 \\ c_0 & 0 & \dots & 0 & d_0 & 0 & \dots & 0 \\ c_1 & c_0 & \dots & 0 & d_1 & d_0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \cdot & \cdot & \cdot & c_0 & \cdot & \cdot & \cdot & d_0 \\ \cdot & \cdot & \cdot & c_1 & \cdot & \cdot & \cdot & d_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & & \dots & c_{n-i} & & & \dots & d_{n-i-1} \end{bmatrix}, \tag{H1}$$

$$q^{(i)}(x) = \det \begin{bmatrix} 0 & 0 & \dots & 0 & x^{n-i} & x^{n-i-1} & \dots & 1 \\ c_0 & 0 & \dots & 0 & d_0 & 0 & \dots & 0 \\ c_1 & c_0 & \dots & 0 & d_1 & d_0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \cdot & \cdot & \dots & c_0 & \cdot & \cdot & \dots & d_0 \\ \cdot & \cdot & \dots & c_1 & \cdot & \cdot & \dots & d_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \cdot & \cdot & \dots & c_{n-i} & \cdot & \cdot & \dots & d_{n-i-1} \end{bmatrix}.$$

Мы можем также представить соотношения (H1) в виде

$$p^{(i)}(x) = \phi_{n,i}[p_{n+1}(x), p_n(x)],$$
$$q^{(i)}(x) = \psi_{n,i}[p_{n+1}(x), p_n(x)],$$

где $\phi_{n,i}$ и $\psi_{n,i}$ рассматриваются как операторы, действующие на любой паре полиномов степеней $n+1$ и n.

Пример. Рассмотрим полиномы $p_{2+1}(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$, т.е. $n = 2$, и вычислим $p^{(1)}(x)$ и $q^{(1)}(x)$, так что

$$p_1(x) = p_{2+1}(x)p^{(1)}(x) + p_2(x)q^{(1)}(x).$$

Заметим, что в этом случае $i = 1$, и, следовательно, определители имеют порядок $2(n-i)+1 = 3$, так что

$$p^{(1)}(x) = \det \begin{bmatrix} 1 & 0 & 0 \\ 1 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix} = 9, \quad q^{(1)}(x) = \det \begin{bmatrix} 0 & x & 1 \\ 1 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix} = -3x$$

и

$$p_1(x) = p_{2+1}(x) \cdot 9 + p_2(x) \cdot (-3x) = -42x + 63,$$

что нам уже известно. [Читателю следует проверить, что мы получим те же полиномы $p^{(1)}(x)$ и $q^{(1)}(x)$, используя расширенный алгоритм Евклида для полиномов, описанный в теореме 5.2.11.]

Затем для вычисления $p^{(0)}(x)$ и $q^{(0)}(x)$, таких, что $p_0 = p_{2+1}(x)p^{(0)}(x) + p_2(x)q^{(0)}(x)$, нам нужно вычислить два определителя порядка 5:

$$p^{(0)}(x) = \det \begin{bmatrix} x & 1 & 0 & 0 & 0 \\ 1 & 0 & 3 & 0 & 0 \\ 0 & 1 & 0 & 3 & 0 \\ -7 & 0 & -7 & 0 & 3 \\ 7 & -7 & 0 & -7 & 0 \end{bmatrix} = 126x + 189$$

и

$$q^{(0)}(x) = \det \begin{bmatrix} 0 & 0 & x^2 & x & 1 \\ 1 & 0 & 3 & 0 & 0 \\ 0 & 1 & 0 & 3 & 0 \\ -7 & 0 & -7 & 0 & 3 \\ 7 & -7 & 0 & -7 & 0 \end{bmatrix} = -42x, 2 - 63x + 196,$$

следовательно,

$$p_0(x) = p_{2+1}(x)(126x + 189) + p_2(x)(-42x^2 - 63x + 196) = -49.$$

Заметим, что в приведенном примере выполняются удивительные соотношения:

$$\operatorname{lc}[p_2(x)] \operatorname{lc}[p_1(x)] = -\operatorname{lc}[p^{(0)}(x)], \quad \operatorname{lc}[p_{2+1}(x)] \operatorname{lc}[p_1(x)] = \operatorname{lc}[q^{(0)}(x)].$$

Как мы уже говорили, старший коэффициент i-го члена PRS равен $\operatorname{res}_{\mathrm{B}}^{(i)}[p_{n+1}(x), p_n(x)]$ и отличается знаком от $\operatorname{res}_{\mathrm{T}}^{(i)}[p_{n+1}(x), p_n(x)]$; игнорируя различие в знаках, мы обозначаем оба результанта R_i. Поэтому в общем случае

$$\operatorname{lc}[p_n(x)]R_i = -\operatorname{lc}[p^{(i-1)}(x)], \quad \operatorname{lc}[p_{n+1}(x)]R_i = \operatorname{lc}[q^{(i-1)}(x)]. \qquad \text{(H2)}$$

Эти формулы проверяются несложными вычислениями, если рассмотреть определители R_i, $p^{(i-1)}(x)$ и $q^{(i-1)}(x)$ вместе с тем фактом, что порядок первого из них равен $2(n-i)+1$, а двух последних $2(n-i)+3$; см. также предыдущий пример.

Таким образом, для каждого индекса i, $0 \le i < n$, мы создали множество из трех полиномов $p_i(x)$, $p^{(i)}(x)$ и $q^{(i)}(x)$; сейчас мы свяжем их попарно следующим образом.

Теорема 5.2.12 (Габихт). В предположениях теоремы 5.2.11 для $1 \le i \le n-1$

$$\det \begin{bmatrix} p^{(i)}(x) & p^{(i-1)}(x) \\ q^{(i)}(x) & q^{(i-1)}(x) \end{bmatrix} = R_i^2, \quad \det \begin{bmatrix} p^{(i)}(x) & p^{(i-1)}(x) \\ p_i(x) & p_{i-1}(x) \end{bmatrix} = R_i^2 p_n(x), \tag{H3}$$

$$\det \begin{bmatrix} q^{(i)}(x) & q^{(i-1)}(x) \\ p_i(x) & p_{i-1}(x) \end{bmatrix} = R_i^2 p_{n+1}(x).$$

Доказательство. Мы имеем

$$\begin{aligned} p_i(x) &= p_{n+1}(x)p^{(i)}(x) + p_n(x)q^{(i)}(x), \\ p_{i-1}(x) &= p_{n+1}(x)p^{(i-1)}(x) + p_n(x)q^{(i-1)}(x). \end{aligned} \tag{H4}$$

Умножая первое соотношение на $p^{(i-1)}(x)$, а второе — на $p^{(i)}(x)$, мы получаем

$$\det \begin{bmatrix} p^{(i)}(x) & p^{(i-1)}(x) \\ p_i(x) & p_{i-1}(x) \end{bmatrix} = \det \begin{bmatrix} p^{(i)}(x) & p^{(i-1)}(x) \\ q^{(i)}(x) & q^{(i-1)}(x) \end{bmatrix} p_n(x). \tag{H5}$$

Левая часть соотношения — это полином степени n, поскольку степень полинома $p_i(x)p^{(i-1)}(x)$ равна n, а степень полинома $p_{i-1}(x)p^{(i)}(x)$ — только $n-2$. Однако, поскольку степень полинома $p_n(x)$ равна n, определитель в правой части соотношения (H5) является целой константой; в действительности эта константа равна $-\operatorname{lc}\{p^{(i-1)}(x)\}R_i/\operatorname{lc}[p_n(x)]$, частному от деления старших коэффициентов полиномов $-p_i(x)p^{(i-1)}(x)$ и $p_n(x)$, которое по первой из формул (H2) равно R_i^2 (проверьте это для последнего примера). Таким образом, мы доказали два первых соотношения (H3). Последнее легко получается после умножения соотношений из (H4) на $q^{(i-1)}(x)$ и $q^{(i)}(x)$ соответственно. $\square$

[Формулы (H3) можно расширить, чтобы включить $p_{n+1}(x)$ и $p_n(x)$, полагая $p^{(n+1)}(x) = 1$, $p^{(n)}(x) = 0$, $q^{(n+1)}(x) = 0$, $q^{(n)}(x) = 1$. В этом случае формулы (H4) и (H5) остаются справедливыми; более того, если мы воспользуемся разницей в знаках, которую до сих пор игнорировали, и положим $p^{(n-1)}(x) = -d_0^2 = -R_n^2$ (посмотрите, как это получилось в последнем примере), то (H3) справедливо для $i = n$, а для $i = n+1$ положим $R_{n+1} = 1$.]

Выведем сейчас соотношение между $p_{i+1}(x)$, $p_i(x)$ и $p_{i-1}(x)$ и более общее соотношение между $p_{i+1}(x)$, $p_i(x)$ и $p_j(x)$, $j < i$ для $1 \le i \le n-1$.

Из двух последних соотношений (H3) мы получаем для $i+1$

$$\begin{aligned} p_i(x)p^{(i+1)}(x) - p_{i+1}(x)p^{(i)}(x) &= R_{i+1}^2 p_n(x), \\ p_i(x)q^{(i+1)}(x) - p_{i+1}(x)q^{(i)}(x) &= -R_{i+1}^2 p_{n+1}(x). \end{aligned}$$

Умножая первое из этих соотношений на $q^{(i-1)}(x)$, а второе — на $p^{(i-1)}(x)$, получаем

$$\begin{aligned} \det \begin{bmatrix} p^{(i+1)}(x) & p^{(i-1)}(x) \\ q^{(i+1)}(x) & q^{(i-1)}(x) \end{bmatrix} p_i(x) - \det \begin{bmatrix} p^{(i)}(x) & p^{(i-1)}(x) \\ q^{(i)}(x) & q^{(i-1)}(x) \end{bmatrix} p_{i+1}(x) \\ = R_{i+1}^2 [p_{n+1}(x)p^{(i-1)}(x) + p_n(x)q^{(i-1)}(x)], \end{aligned}$$

что в соответствии с (H3) равно

$$R_{i+1}^2 p_{i-1}(x) = -R_i^2 p_{i+1}(x) + Q_i(x)p_i(x), \tag{H6}$$

где $Q_i(x)$ — полином степени 1, который может быть выражен через коэффициенты полиномов $p_{i+1}(x)$ и $p_i(x)$ (например, коэффициент при x равен $R_i R_{i+1}$). С использованием введенных ранее операторов (H6) можно переписать в виде

$$R_{i+1}^2 p_{i-1}(x) = \phi_{i,i-1}[p_{i+1}(x), p_i(x)]p_{i+1}(x) + \psi_{i,i-1}[p_{i+1}(x), p_i(x)]p_i(x). \tag{H6$'$}$$

Основываясь на этом уравнении, рассмотрим теперь более общий случай.

Теорема 5.2.13 (Habicht, 1948). В предположениях теоремы 5.2.11, если положить $p^{(ij)}(x) = \phi_{ij}[p_{i+1}(x), p_i(x)]$ и $q^{(ij)}(x) = \psi_{ij}[p_{i+1}(x), p_i(x)]$ $(0 < i < n$ и $0 \le j < i)$, то

$$R_{i+1}^{2(i-j)} p_j(x) = p^{(ij)}(x)p_{i+1}(x) + q^{(ij)}(x)p_i(x). \tag{H7}$$

Доказательство. Пользуясь обозначениями этой теоремы, полиномы $p^{(i)}(x)$ и $q^{(i)}(x)$, определенные в теореме 5.2.11, можно записать как $p^{(ni)}(x)$ и $q^{(ni)}(x)$ соответственно. Поэтому полиномы $p^{(ij)}(x)$ и $q^{(ij)}(x)$ могут быть вычислены исходя из определителя, элементами которого являются коэффициенты полиномов $p_{i+1}(x)$ и $p_i(x)$, таким же образом, как полиномы $p^{(ni)}(x)$ и $q^{(ni)}(x)$ могут быть вычислены исходя из определителя, элементами которого являются коэффициенты полиномов $p_{n+1}(x)$ и $p_n(x)$.

Фиксировав i, образуем полиномы

$$p'_i(x) = p_i(x) \text{ и } p'_j(x) = p_{i+1}(x)p^{(ij)}(x) + p_i(x)q^{(ij)}(x),$$
$$j = i-1, \ldots, 0.$$

Заметим, что полиномы $p'_j(x)$, $j < i$, выражаются через $p_{i+1}(x)$ и $p_i(x)$, а полиномы $p_j(x)$ — через $p_{n+1}(x)$ и $p_n(x)$ (теорема 5.2.11); поэтому коэффициенты первых полиномов больше, чем коэффициенты вторых, и мы хотим доказать, что

$$p'_j(x) = R_{i+1}^{2(i-j)} p_j(x). \tag{H8}$$

Для простоты обозначений положим также $R_{i+1}^2 = r$ {напомним, что $R_{i+1} = \mathrm{lc}[p_{i+1}(x)]$} и образуем полиномы с рациональными коэффициентами

$$p''_j(x) = \left[\frac{1}{r^{(i-j)}}\right] p'_j(x), \quad j = i, \ldots, 0.$$

Обозначим старшие коэффициенты полиномов $p'_j(x)$ и $p''_j(x)$ через R'_j и R''_j соответственно. Покажем, что (H6) выполняется для любых $p''_{j+1}(x)$, $p''_j(x)$ и $p''_{j-1}(x)$, $1 \le j \le i-1$, и этим закончим доказательство теоремы.

Очевидно, что $p''_i(x) = p_i(x)$, а по (H6′) мы имеем $p''_{i-1}(x) = p_{i-1}(x)$; поэтому мы можем последовательно заключить, что также $p''_j(x) = p_j(x)$, $j = i-2, \ldots, 0$. Однако по определению операторов ϕ_{ij} и ψ_{ij} повторным применением использованных выше рассуждений мы устанавливаем, что (H6) выполняется для значений $p'_j(x)$, т.е.

$$(R'_{j+1})^2 p'_{j-1}(x) = -(R'_j) 2p'_{j+1}(x) + Q'_j(x)p'_j(x), \quad j = i-1, \ldots, 1.$$

Следовательно, разделив для каждого индекса j соответствующее соотношение на

$$r^{2(i-j-1)} r^{i-j+1} = r^{2(i-j)} r^{i-j-1},$$

получаем

$$(R''_{j+1})^2 p''_{j-1}(x) = -(R''_j)^2 p''_{j+1}(x) + Q''_j(x)p''_j(x), \quad j = i-1, \ldots, 1. \ \square$$

Эта теорема показывает, что (без вычисления наибольших общих делителей) наименьшие коэффициенты членов PRS получаются как субрезультанты результанта $\mathrm{res}_B[p_{n+1}(x), p_n(x)]$, т.е. мы получаем наименьшие коэффициенты для членов PRS только при использовании коэффициентов полиномов $p_{n+1}(x)$ и $p_n(x)$. Однако вычисление определителей является трудоемким процессом, и члены PRS вычисляются путем псевдоделения; коэффициенты членов PRS тогда получаются бóльшими, но мы уже знаем, на что их нужно делить.

Пример. Рассмотрим снова полиномы из последнего примера $p_{2+1}(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$, для которых мы уже знаем, что $p_1(x) = -42x + 63$ и $p_0(x) = -49$. Очевидно, что в этом случае $p_1(x) = p_1'(x)$, и, таким образом, мы можем вычислить $p_0'(x) = p_2(x)p^{(10)}(x) + p_1(x)q^{(10)}$. Мы имеем $i = 1$, $j = 0$ и

$$p^{(10)}(x) = \det \begin{bmatrix} 1 & 0 & 0 \\ 3 & -42 & 0 \\ 0 & 63 & -42 \end{bmatrix} = 1764,$$

$$q^{(10)}(x) = \det \begin{bmatrix} 0 & x & 1 \\ 3 & -42 & 0 \\ 0 & 63 & -42 \end{bmatrix} = 126x + 189.$$

Поэтому $p_0'(x) = 1764(3x^2 - 7) + (126x + 189)(-42x + 63) = -441$, и по теореме 5.2.13 3^2, старший коэффициент полинома $p_2(x)$, возведенный в квадрат, делит -441, т.е. $p_0(x) = p_0'(x)/9 = -441/3^2 = -49$.

В дальнейшем мы полагаем $R_{n+1} = c_0$, $R_n = d_0$ и рассмотрим, что случится, если один или несколько главных субрезультантных коэффициентов равны нулю, т.е. мы выведем формулы, составляющие основу алгоритма Габихта субрезультантных PRS [формула (H) разд. 5.1.1]. Наши исследования основаны на соотношениях (H7) и (H8).

Рассмотрим последовательность полиномиальных остатков $p_{n+1}(x), p_n(x), \ldots, p_0(x)$, где для всех i, $0 \le i \le n-1$,

$$p_i(x) = c_{i,0}x^i + \cdots + c_{i,i},$$

и пусть $p_i(x)$, $0 < i \le n$, — член этой последовательности, такой, что

$$R_{i+1} \neq 0,\ R_i = 0,\ c_{i1} = 0, \ldots, c_{i,i-j-1} = 0,\ c_{i,i-j} \neq 0,$$

т.е. степень полинома $p_i(x)$ равна $j \le i$. Тогда следующие соотношения легко выводятся из (H7), (H8) и вида полинома $p_i(x)$ [напомним,

что определители $p^{(ij)}(x)$ и $q^{(ij)}(x)$ получаются таким же образом, как и определители $p^{(i)}(x)$ и $q^{(i)}(x)$, т.е. из (H1), где n заменено на i, i на j и коэффициенты полиномов $p_{n+1}(x)$ и $p_n(x)$ — коэффициентами полиномов $p_{i+1}(x)$ и $p_i(x)$].

H-1. $p_{i-1}(x), \ldots, p_{j+1}(x)$ равны нулю.

H-2. $p^{(ij)}(x) = 0,\ q^{(ij)}(x) = R_{i+1}^{i-j} c_{i,i-j}^{i-j},$

$$R_{i+1}^{i-j} p_j(x) = c_{i,i-j}^{i-j} p_i(x) \qquad (i < n),$$
$$p_j(x) = d_{n-j}^{n-j} c_0^{n-j} p_n(x) \qquad (i = n).$$

H-3. $p^{(i,j-1)}(x) = (-1)^{i-j} R_{i+1}^{i-j} c_{i,i-j}^{i-j+2},$

$$R_{i+1}^{2(i-j+1)} p_{j-1}(x) = (-1)^{i-j} R_{i+1}^{i-j} c_{i,i-j}^{i-j+2} p_{i+1}(x) + q^{(i,j-1)}(x) p_i(x) \qquad (i < n),$$
$$p_{j-1}(x) = d_{n-j}^{n-j+2} c_0^{n-j} p_{n+1}(x) + q^{(j-1)}(x) p_n(x) \qquad (i = n).$$

Формулы пунктов **H-1, H-2** и **H-3** составляют основу алгоритма Габихта субрезультантных PRS.

Пример. Рассмотрим неполную последовательность полиномиальных остатков $p_4(x) = 2x^4+5x^3+5x^2-2x+1$, $p_3(x) = 3x^3+3x^2+3x-4$, $p_2(x) = 0x^2 - 21x + 45$, $p_1(x) =$–?–, $p_0(x) = -47049$. Мы видим, что здесь $R_{2+1} \neq 0$, но $R_2 = 0$, т.е. во введенных выше обозначениях $i = 2$ и $j = 1$. Тогда, чтобы вычислить $p_1(x)$, мы применяем случай **H-2** и получаем

$$p^{(21)}(x) = \det \begin{bmatrix} 1 & 0 & 0 \\ 3 & 0 & 0 \\ 3 & -21 & 0 \end{bmatrix} = 0,$$
$$q^{(21)}(x) = \det \begin{bmatrix} 0 & x & 1 \\ 3 & 0 & 0 \\ 3 & -21 & 0 \end{bmatrix} = 3(-21),$$

где $n := i$, $i := j$ и коэффициенты полиномов $p_4(x)$ и $p_3(x)$ заменены коэффициентами полиномов $p_3(x)$ и $p_2(x)$. Поэтому из $p_1'(x) = p_3(x)p^{(21)}(x) + p_2(x)q^{(21)}$ получаем

$$p_1'(x) = 3(-21)p_2(x), \quad \text{или, эквивалентно,} \quad 3^2 p_1(x) = 3(-21)p_2(x)$$

и

$$p_1(x) = -147x + 315,$$

т.е. $p_1(x)$ — это полином, пропорциональный полиному $p_2(x)$. Читателю предлагается в качестве упражнения вычислить $p_0(x)$, используя случай **H-3**.

Рис. 5.2.1.

Лакунарная структура PRS; отрезок длины $k+1$ представляет полином степени k (Габихт).

Представляет интерес рис. 5.2.1.

Используя (H) из разд. 5.1.1, получаем следующий алгоритм.

HSPRS. Субрезультантные PRS Габихта (**H**abicht's **S**ubresultant **PRS**)

Вход: Два ненулевых полинома $p_1(x), p_2(x)$ в $\mathbb{Z}[x]$, $\deg[p_1(x)] \geq \deg[p_2(x)]$.

Выход: Субрезультантная PRS полиномов $p_1(x)$ и $p_2(x)$.

1. [Инициализация] $i := 1$; $\beta_1 := (-1)^{\deg[p_1(x)]-\deg[p_2(x)]+1}$.

2. [Готово?] Если $p_{i+1}(x) = 0$, то выход из алгоритма.

3. [Вычисление псевдоостатка] $p_{i+2}(x) := \operatorname{prem}[p_i(x), p_{i+1}(x)]/\beta_i$, где β_i вычислено из соотношения (H) разд. 5.1.1 (см. также ниже упр. 3 к этому разделу); положить $i := i+1$ и перейти к шагу 2.

Анализ времени работы алгоритма ***HSPRS***. Пусть $n = \deg[p_1(x)]$. Тогда в худшем случае имеется n полиномиальных псевдоделений над целыми числами, которые выполняются (см. также разд. 3.1.1) за время $O[\{(n-1)\cdot 2 + (n-2)\cdot 2 + \cdots + 1\cdot 2\}\cdot L^2(a)] = O[n^2 L^2(a)]$, где a — наибольший коэффициент, появляющийся в ходе вычислений [аналогично времени, требуемому для вычисления произведения $p_1(x)\cdot p_2(x)$]. Множитель $L^2(a)$ соответствует имеющему место умножению целых чисел (коэффициентов). Наибольший коэффициент, появляющийся в ходе вычислений, $\leq \operatorname{res}_{\mathrm{B}}[p_1(x), p_2(x)]$, что в свою очередь ограничено по неравенству Адамара

$$|\operatorname{res}_{\mathrm{B}}[p_1(x), p_2(x)]| \leq \prod_{1\leq i\leq 2n}\left[\sum_{1\leq j\leq 2n} r_{ij}^2\right]^{1/2} \leq (2n)^n (|p(x)|_\infty)^{2n},$$

где $|p(x)|_\infty = \max(|p_1(x)|_\infty, |p_2(x)|_\infty)$, а r_{ij} обозначают элементы результанта. Если положить $a = (2n)^n(|p(x)|_\infty)^{2n}$, то $L^2(a) = [nL(2n) + 2nL(|p(x)|_\infty)]^2 = O(n^2L^2[|p(x)|_\infty])$, и время всех полиномиальных псевдоделений равно

$$t_{\mathbf{HSPRS}}[p_1(x), p_2(x)] = O\{n^4L^2(|p(x)|_\infty]\}.$$

5.3

Метод матричной триангуляризации субрезультантных PRS

Мы видели, что метод Сильвестра–Габихта псевдоделения субрезультантных PRS включает в себя два алгоритма: алгоритм Сильвестра редуцированных (субрезультантных) PRS (основанный на статье Сильвестра 1853 г.), очень хорошо работающий для полных PRS, и алгоритм Габихта субрезультантных PRS (основанный на статье Габихта 1948 г.), более предпочтительный для неполных PRS. Заметим, что алгоритм Габихта субрезультантных PRS может быть использован для полных PRS, но за счет дополнительных затрат. Более того, мы не можем a priori сказать, будет получающаяся PRS полной или неполной.

В этом разделе мы представляем метод матричной триангуляризации субрезультантных PRS, разработанный автором в 1986 г. Этот метод основан на полученном автором обобщении теоремы Ван Влека и с единой позиции подходит к полным и неполным PRS. Более того, коэффициенты членов PRS являются наименьшими из тех, что могут быть получены без вычисления наибольших общих делителей коэффициентов, и ясно демонстрируются свойства делимости. (В разд. 5.1.1 мы видели пример, когда коэффициенты, полученные этим новым методом, меньше коэффициентов, полученных методом PRS Сильвестра–Габихта.)

Удивительный факт о методе матричной триангуляризации субрезультантных PRS состоит в том, что в нем нет явного деления полиномов, вместо этого используется процесс гауссова исключения (см. приложение в конце книги), применяемый к единственной матрице, соответствующей результанту в форме Сильвестра (рассматриваемому ниже), и все члены PRS получаются из строк результирующей верхней треугольной матрицы с помощью теоремы 5.3.6. Следует заметить, что результант в форме Сильвестра впервые используется в современной литературе. (См. также историческое замечание 2.)

5.3.1. Гауссово исключение и полиномиальное деление

Рассмотрим полиномы $p_1(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0$, $c_n \neq 0$, $n > 0$, $p_2(x) = d_m x^m + d_{m-1}x^{m-1} + \cdots + d_0$, $d_m \neq 0$, $m > 0$, над целыми числами $n \geq m$. Из евклидовости мы знаем, что существуют единственные полиномы с целыми коэффициентами $q(x)$ и $r(x)$, такие, что $p_1(x) = p_2(x)q(x) + r(x)$, $\deg[r(x)] < \deg[p_2(x)]$ (в этом месте читателю следует посмотреть еще раз алгоритм **PDF**). Пусть $r(x) = r_{m-1}x^{m-1} + r_{m-2}x^{m-2} + \cdots + r_0$.

Мы утверждаем, что деление полиномов над полем или псевдоделение над целыми числами эквивалентно приведению матрицы $\mathbf{M}$ к верхней треугольной форме $T(\mathbf{M})$, т.е. все элементы m_{ij}, $j < i$, матрицы $T(\mathbf{M})$ равны нулю. Поскольку гауссово исключение — это процесс, приводящий данную матрицу элементарными преобразованиями над строками к верхней треугольной форме, наше утверждение состоит в том, что *полиномиальное деление эквивалентно гауссову исключению.*

Чтобы убедиться в этом, рассмотрим упомянутые выше полиномы и сформируем $(n-m+2)\times(n+1)$-матрицу $\mathbf{M}$:

$$\mathbf{M} = \left.\begin{bmatrix} d_m & d_{m-1} & \ldots & d_0 & 0 & \ldots & 0 \\ 0 & d_m & d_{m-1} & \ldots & d_0 & \ldots & 0 \\ & & & & \ddots & & \\ 0 & 0 & \ldots & d_m & d_{m-1} & \ldots & d_0 \\ c_n & c_{n-1} & & \ldots & & & c_0 \end{bmatrix}\right\} (n-m+1) \text{ строк}$$

где коэффициенты полиномов $x^{n-m}p_2(x)$, $x^{n-m-1}p_2(x)$, $\ldots$, $p_2(x)$ составляют первую, вторую, $\ldots$ и $(n-m+1)$-ю строки $(n-m+2)\times(n+1)$-матрицы; коэффициенты полинома $p_1(x)$ составляют последнюю строку этой матрицы. Если мы приведем теперь матрицу $\mathbf{M}$ к верхней треугольной форме $T(\mathbf{M})$, то увидим, что элементы последней строки преобразуются в коэффициенты остатка $r(x)$, т.е.

$$T(\mathbf{M}) = \begin{bmatrix} d_m & d_{m-1} & \ldots & d_0 & 0 & \ldots & 0 \\ 0 & d_m & d_{m-1} & \ldots & d_0 & \ldots & 0 \\ & & & & \ddots & & \\ 0 & 0 & \ldots & d_m & d_{m-1} & \ldots & d_0 \\ 0 & 0 & \ldots & 0 & r_{m-1} & \ldots & r_0 \end{bmatrix}.$$

Мы вычли здесь из последней строки матрицы $\mathbf{M}$ каждую из первых $n-m+1$ ее строк, предварительно домноженную на соответствующее число, разность стала новой последней строкой. Таким образом, мы последовательно исключили степени x до x^{m-1}.

Пример. Рассмотрим полиномы $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$. Тогда

$$\mathbf{M} = \begin{bmatrix} 3 & 0 & -7 & 0 \\ 0 & 3 & 0 & -7 \\ 1 & 0 & -7 & 7 \end{bmatrix},$$

и, приведя ее к верхней треугольной форме, получаем

$$T(\mathbf{M}) = \begin{bmatrix} 3 & 0 & -7 & 0 \\ 0 & 3 & 0 & -7 \\ 0 & 0 & -14 & 21 \end{bmatrix},$$

откуда заключаем, что псевдоостаток от деления $p_1(x)$ на $p_2(x)$ равен $r(x) = -14x + 21 = p_3(x)$. (Читателю следует заметить, что коэффициенты здесь меньше, чем полученные непосредственным псевдоделением.)

Далее мы будем иметь дело только с квадратными матрицами и будем приводить их к верхней треугольной форме, пользуясь алгоритмом Доджсона, сохраняющим целочисленность (см. историческое замечание 3). Для $n \times n$-матрицы $\mathbf{M}$ алгоритм Доджсона работает следующим образом. Положим $m_{00}^{(-1)} := 1$ и $m_{ij}^{(0)} := m_{ij}$ (элементы матрицы), $i, j = 1, \ldots, n$; тогда для $k < i, j \le n$ положим

$$m_{ij}^{(k)} := [m_{k-1,k-1}^{(k-2)}]^{-1} \det \begin{bmatrix} m_{kk}^{(k-1)} & m_{kj}^{(k-1)} \\ m_{ik}^{(k-1)} & m_{ij}^{(k-1)} \end{bmatrix}. \qquad \text{(D)}$$

То есть при $i \le k$ строки не меняются, а при $k < i$ мы получаем в соответствующих строках 0 в нижнем треугольнике. [См. также (Малашонок, 1983, 1986).]

Заметим, что нам нужен выбор ведущего элемента (перестановки строк), чтобы обеспечить условие $m_{kk}^{(k-1)} \neq 0$ для всех k, и, как мы убедимся ниже, этот выбор должен быть сделан так, чтобы он не испортил симметрию матрицы, соответствующей результанту в форме Сильвестра. Поэтому мы пользуемся *выбором ведущего элемента методом пузырька*, который определяется следующим образом: если нам нужно переставить строку i со строкой j, где $j > i + 1$, то мы позволяем строке j *подняться* до уровня строки i, последовательно меняя ее местами (попарно) с каждой из строк, лежащих над ней. Таким образом, после выбора ведущего элемента методом пузырька бывшая строка i находится непосредственно под бывшей строкой j.

В алгоритме Доджсона очень существенно, что определитель порядка 2 *без остатка* делится на $m_{k-1,k-1}^{(k-2)}$ и что элементы получающейся матрицы являются наименьшими среди тех, которые можно получить без вычисления gcd коэффициентов и без введения рациональных чисел. Чтобы в этом убедиться, рассмотрим матрицу [см. также (Малашонок, 1983)]

$$\begin{bmatrix} a & b & c & d & e & \dots \\ f & g & h & i & j & \dots \\ k & l & m & n & o & \dots \\ \dots & & & & & \end{bmatrix}.$$

После первого преобразования, $k = 1$, мы получаем матрицу

$$\begin{bmatrix} a & b & c & d & e & \dots \\ 0 & ag-bf & ah-cf & ai-df & aj-ef & \dots \\ 0 & al-bk & am-ck & an-dk & ao-ek & \dots \\ \dots & & & & & \end{bmatrix},$$

т.е. нет никакого деления, поскольку $m_{0,0}^{(-1)} = 1$. После второго преобразования, $k = 2$, мы получаем следующую матрицу (*до деления*):

$$\begin{aligned} m_{3,3}^{(2)} &= (ag-bf)(am-ck) - (al-bk)(ah-cf) \\ &= aagm - acgk - abfm + bcfk - aahl + acfl + abhk - bcfk \\ &= aagm - acgk - abfm - aahl + acfl + abhk, \\ m_{3,4}^{(2)} &= aagm - adgk - abfn - aail + adfl + abik, \dots . \end{aligned}$$

Таким образом, мы видим, что каждый элемент подматрицы можно без остатка разделить на a. Подобным образом мы сможем разделить на $(ag-bf)$ элементы новой подматрицы после следующего преобразования и т.д.

5.3.2. Гауссово исключение и результанты в формах Брюно и Труди

Напомним, что для двух данных полиномов $p(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0$ и $q(x) = d_m x^m + d_{m-1}x^{m-1} + \cdots + d_0$, $n \geq m$, матрица $\mathbf{M}_{\mathrm{B}}$,

соответствующая результанту в форме Брюно, равна

$$\mathbf{M}_{\mathrm{B}} = \left[\begin{array}{cccccccc} c_n & c_{n-1} & \dots & & c_0 & 0 & \dots & 0 \\ 0 & c_n & c_{n-1} & \dots & & c_0 & \dots & 0 \\ & & & & \ddots & & & \\ 0 & 0 & \dots & & c_n & c_{n-1} & \dots & c_0 \\ d_m & d_{m-1} & \dots & d_0 & 0 & 0 & \dots & 0 \\ 0 & d_m & d_{m-1} & \dots & d_0 & 0 & \dots & 0 \\ & & & & \ddots & & & \\ 0 & 0 & \dots & d_m & d_{m-1} & & \dots & d_0 \end{array}\right] \begin{array}{l} \left.\vphantom{\begin{array}{c}a\\a\\a\\a\end{array}}\right\}(m \text{ строк}) \\ \left.\vphantom{\begin{array}{c}a\\a\\a\\a\end{array}}\right\}(n \text{ строк}) \end{array}$$

Мы хотим доказать, что если мы приведем $\mathbf{M}_{\mathrm{B}}$ к верхней треугольной форме $T(\mathbf{M}_{\mathrm{B}})$, пользуясь только преобразованиями строк, то последняя ненулевая строка матрицы $T(\mathbf{M}_{\mathrm{B}})$ даст нам коэффициенты полинома $\gcd[p(x), q(x)]$. Заметим, что *в общем случае* при преобразовании $\mathbf{M}_{\mathrm{B}}$ в $T(\mathbf{M}_{\mathrm{B}})$ мы можем получить *только* gcd, но *ни одного* другого члена PRS, порожденной полиномами $p(x)$ и $q(x)$.

Для доказательства главного результата прежде всего нам понадобится следующая

Лемма 5.3.1 (Laidacker, 1969). Пусть $p_i(x), p_i^+(x)$ — полиномы, соответствующие i-м строкам матриц $\mathbf{M}_{\mathrm{B}}$ (определенной выше) и $T(\mathbf{M}_{\mathrm{B}})$ соответственно, где $T(\mathbf{M}_{\mathrm{B}})$ — верхняя треугольная форма матрицы $\mathbf{M}_{\mathrm{B}}$, полученная из последней только преобразованиями строк. Если k — степень полинома $p_r^+(x)$, соответствующего последней ненулевой строке матрицы $T(\mathbf{M}_{\mathrm{B}})$, то не существует полиномов вида

$$\alpha_1 p_1(x) + \alpha_2 p_2(x) + \cdots + \alpha_{m+n} p^{m+n}(x) \qquad \text{(L)}$$

степени, меньшей k, где α_i — константа, $i = 1, 2, \dots, m+n$.

Доказательство. В обеих матрицах, $\mathbf{M}_{\mathrm{B}}$ и $T(\mathbf{M}_{\mathrm{B}})$, элементы первого столбца — это коэффициенты при степени x^{m+n-1}, элементы второго столбца – это коэффициенты при степени x^{m+n-2} и т.д. Из линейной алгебры нам известно, что множество линейных комбинаций вида (L), порожденных полиномами $\{p_i(x)\}$, совпадает с множеством, порожденным полиномами $\{p_i^+(x)\}$, $i = 1, 2, \dots, m+n$, поэтому мы будем доказывать теорему для последнего множества.

Прежде всего заметим, что при $k = 0$ теорема очевидным образом верна. Предположим теперь, что она неверна при $k > 0$. Тогда существуют константы β_i, $i = 1, 2, \dots, r$, такие, что $\beta_1 p_1^+(x) +$

$\beta_2 p_2^+(x) + \cdots + \beta_r p_r^+(x) = p^+(x)$, где $\deg[p^+(x)] < k$. Так как $\deg[p_1^+(x)] = m+n-1 > k$, а также $\deg[p_1^+(x)] > \deg[p_i^+(x)]$, $i = 2, \ldots, r$, то β_1 должно равняться нулю. Рассмотрим оставшееся выражение $\beta_2 p_2^+(x) + \cdots + \beta_r p_r^+(x) = p^+(x)$; по тем же причинам, что и выше, мы заключаем, что $\beta_i = 0$ для $i = 2, 3, \ldots, r$, откуда следует, что $p^+(x)$ — нулевой полином, что противоречит предположению.

Теорема 5.3.2 (Laidacker, 1969). В предположениях леммы 5.3.1 последняя ненулевая строка матрицы $T(\mathbf{M}_\mathrm{B})$, полученной из $\mathbf{M}_\mathrm{B}$ с помощью только преобразований строк, дает коэффициенты полинома $\gcd[p(x), q(x)]$.

Доказательство. Пусть $g_k, g_{k-1}, \ldots, g_0$ — элементы последней ненулевой строки матрицы $T(\mathbf{M}_\mathrm{B})$; в этом случае $p_r^+(x) = g_k x^k + g_{k-1} x^{k-1} + \cdots + g_0$. По построению $\mathbf{M}_\mathrm{B}$ мы видим, что любой полином вида $\alpha_1 p_1(x) + \cdots + \alpha_{m+n} p^{m+n}(x)$ может быть записан в виде $p(x) g_r(x) + q(x) g_r'' x)$, где $\deg[g_r(x)] \le m-1$ и $\deg[g_r''(x)] \le n-1$. Ясно, что $p_r^+(x)$ также может быть записан в виде $p(x) g_r(x) + q(x) g_r'' x)$, тогда, очевидно, $\gcd[p(x), q(x)] | p_r^+(x)$, откуда мы заключаем, что $\deg\{\gcd[p(x), q(x)]\} \le \deg[p_r^+(x)]$. Однако по лемме 5.3.1 не существует полиномов степени $< k$, которые могут быть представлены в виде $p(x) g_r(x) + q(x) g_r''(x)$, а следовательно, наша теорема доказана. □

Пример. Рассмотрим пару полиномов $p(x) = x^3 - 7x + 7$ и $q(x) = 3x^2 - 7$, порождающую полную PRS. Тогда

$$\mathbf{M}_\mathrm{B} = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 7 \\ 3 & 0 & -7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 \\ 0 & 0 & 3 & 0 & -7 \end{bmatrix}, T(\mathbf{M}_\mathrm{B}) = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 7 \\ 0 & 0 & 14 & -21 & 0 \\ 0 & 0 & 0 & 196 & -294 \\ 0 & 0 & 0 & 0 & -49 \end{bmatrix}.$$

Из последней строки матрицы $T(\mathbf{M}_\mathrm{B})$ мы заключаем, что эти два полинома взаимно просты. Заметим также, что в $T(\mathbf{M}_\mathrm{B})$ мы теряем полином $q(x)$, и оказывается, что мы получаем полиномиальный остаток, третью строку, соответствующую полиному $14x - 21$.

Однако если мы рассмотрим пару полиномов $p(x) = x^5 + 5x^4 + 10x^3 + 5x^2 + 5x + 2$ и $q(x) = x^4 + 4x^3 + 6x^2 + 2x + 1$, порождающую

неполную PRS, то

$$\mathbf{M}_{\mathrm{B}} = \begin{bmatrix} 1 & 5 & 10 & 5 & 5 & 2 & 0 & 0 & 0 \\ 0 & 1 & 5 & 10 & 5 & 5 & 2 & 0 & 0 \\ 0 & 0 & 1 & 5 & 10 & 5 & 5 & 2 & 0 \\ 0 & 0 & 0 & 1 & 5 & 10 & 5 & 5 & 2 \\ 1 & 4 & 6 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 6 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & 6 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 4 & 6 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 2 & 1 \end{bmatrix}$$

и

$$T(\mathbf{M}_{\mathrm{B}}) = \begin{bmatrix} 1 & 5 & 10 & 5 & 5 & 2 & 0 & 0 & 0 \\ 0 & 1 & 5 & 10 & 5 & 5 & 2 & 0 & 0 \\ 0 & 0 & 1 & 5 & 10 & 5 & 5 & 2 & 0 \\ 0 & 0 & 0 & 1 & 5 & 10 & 5 & 5 & 2 \\ 0 & 0 & 0 & 0 & -19 & -22 & -13 & -12 & -4 \\ 0 & 0 & 0 & 0 & 0 & 215 & 64 & 81 & 46 \\ 0 & 0 & 0 & 0 & 0 & 0 & -306 & 36 & 116 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 756 & 532 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 980 \end{bmatrix}.$$

Учитывая, что члены PRS, порожденной двумя данными выше полиномами, суть

$$-3x^2 + 2x + 1,\ -266x - 112,\ 980,$$

мы видим, что из $T(\mathbf{M}_{\mathrm{B}})$ мы можем извлечь заключение только о $\gcd[p(x), q(x)]$.

Очевидно, что форма Труди результанта по существу совпадает с формой Брюно, за исключением того, что нижнее множество строк попарно переставлено. Таким образом, теорема 5.3.2 остается верной в этом случае, и мы можем ожидать, что получим только заключение о наибольшем общем делителе двух полиномов (который будет дан со знаком $-$).

Пример. Рассмотрим пару полиномов, исследовавшихся в предыдущем примере. Прежде всего для $p(x) = x^3 - 7x + 7$ и $q(x) = 3x^2 - 7$ мы имеем

$$\mathbf{M}_{\mathrm{T}} = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 7 \\ 0 & 0 & 3 & 0 & -7 \\ 0 & 3 & 0 & -7 & 0 \\ 3 & 0 & -7 & 0 & 0 \end{bmatrix}, \quad T(\mathbf{M}_{\mathrm{T}}) = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 \\ 0 & 1 & 0 & -7 & 7 \\ 0 & 0 & 3 & 0 & -7 \\ 0 & 0 & 0 & 42 & -63 \\ 0 & 0 & 0 & 0 & 49 \end{bmatrix}.$$

Заметим, что в $T(\mathbf{M}_{\mathrm{T}})$ предпоследняя строка соответствует полиному $42x-63=(-3)(-14x+21)$, т.е. это кратное противоположного к остатку полинома. Подобным же образом последняя строка — это наибольший общий делитель со знаком $-$.

Теперь для $p(x)=x^5+5x^4+10x^3+5x^2+5x+2$ и $q(x)=x^4+4x^3+6x^2+2x+1$ мы имеем

$$\mathbf{M}_{\mathrm{T}}=\begin{bmatrix}1&5&10&5&5&2&0&0&0\\0&1&5&10&5&5&2&0&0\\0&0&1&5&10&5&5&2&0\\0&0&0&1&5&10&5&5&2\\0&0&0&0&1&4&6&2&1\\0&0&0&1&4&6&2&1&0\\0&0&1&4&6&2&1&0&0\\0&1&4&6&2&1&0&0&0\\1&4&6&2&1&0&0&0&0\end{bmatrix}$$

и

$$T(\mathbf{M}_{\mathrm{T}})=\begin{bmatrix}1&5&10&5&5&2&0&0&0\\0&1&5&10&5&5&2&0&0\\0&0&1&5&10&5&5&2&0\\0&0&0&1&5&10&5&5&2\\0&0&0&0&1&4&6&2&1\\0&0&0&0&0&3&-5&1&1\\0&0&0&0&0&0&9&-6&-3\\0&0&0&0&0&0&0&-266&-112\\0&0&0&0&0&0&0&0&-980\end{bmatrix}.$$

Заметим, что четыре последние строки матрицы $T(\mathbf{M}_{\mathrm{T}})$ соответствуют четырем полиномам, из которых *только три последних* принадлежат PRS, порожденной $p(x)$ и $q(x)$. Таким образом, $9x^2-6x-3=(-3)(-3x^2+2x+1)$, где $-3x^2+2x+1$ — остаток от деления $p(x)$ на $q(x)$, $-266x-112=(-1)(266x+112)$, а $266x+112$ — второй полиномиальный остаток и -980 — это gcd со знаком $-$.

Из этого примера мы видим, что из $T(\mathbf{M}_{\mathrm{T}})$, так же как из $T(\mathbf{M}_{\mathrm{B}})$, мы можем получить сведения *только* о наибольшем общем делителе двух полиномов. Однако в случае полных PRS из $T(\mathbf{M}_{\mathrm{T}})$ могут быть получены противоположные к полиномиальным остаткам.

5.3.3. Гауссово исключение + результант в форме Сильвестра = метод матричной триангуляризации субрезультантных PRS

До сих пор нам встречались результанты в формах Брюно и Труди, в действительности в литературе обе эти формы обычно называются

формами Сильвестра. Однако мы будем называть результантом в форме Сильвестра описываемый ниже результант; эта форма была «погребена» в статье Сильвестра 1853 г. и упоминалась только в статье Ван Влека (Van Vleck, 1899). Сильвестр указывает (с. 426 его статьи 1853 г.), что он получил эту форму в 1839 или 1840 г., а несколькими годами позже ее воспроизвел, не зная о ней, также Кэли. Именно результант в форме Сильвестра образует основу метода матричной триангуляризации субрезультантных PRS.

Рассмотрим два полинома в $\mathbb{Z}[x]$, $p(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0$ и $q(x) = d_m x, m + d_{m-1}x^{m-1} + \cdots + d_0$, $c_n \neq 0$, $d_m \neq 0$, $n \geq m$. Тогда результант в форме Сильвестра, $\mathrm{res_S}[p(x), q(x)]$, имеет порядок $2n$ и следующий вид [$q(x)$ преобразован в полином степени n введением нулевых коэффициентов]:

$$\mathrm{res_S}[p(x), q(x)] = \det \begin{bmatrix} c_n & c_{n-1} & & \dots & c_0 & 0 & 0 & \dots & 0 \\ d_n & d_{n-1} & & \dots & d_0 & 0 & 0 & \dots & 0 \\ 0 & c_n & & \dots & & c_0 & 0 & \dots & 0 \\ 0 & d_n & & \dots & & d_0 & 0 & \dots & 0 \\ & & & & \ddots & & & \ddots & \\ 0 & \dots & 0 & c_n & c_{n-1} & & & \dots & c_0 \\ 0 & \dots & 0 & d_n & d_{n-1} & & & \dots & d_0 \end{bmatrix}. \tag{S1}$$

Сильвестр получил эту форму результанта (известную также как *элиминант*) из системы уравнений

$$\begin{aligned} p(x) &= 0, \\ q(x) &= 0, \\ x \cdot p(x) &= 0, \\ x \cdot q(x) &= 0, \\ x^2 \cdot p(x) &= 0, \\ x^2 \cdot q(x) &= 0, \\ &\vdots \\ x^{n-1} \cdot p(x) &= 0, \\ x^{n-1} \cdot q(x) &= 0 \end{aligned}$$

и указал, что если мы возьмем k пар из выписанных выше уравнений, то наивысшая степень переменной x, появляющаяся в некотором из них, равна x^{n+k-1}. Поэтому мы сможем исключить такое количество степеней x, что x^{n-k} будет наивысшей неисключенной степенью и $n - k$ будет степенью одного из членов последовательности

(Штурма) полиномов, противоположных (со знаком «минус») к полиномиальным остаткам, порожденной полиномами $p(x)$ и $q(x)$; это доказано в теореме 5.3.4. Более того, Сильвестр показал, что полиномиальные остатки, полученные таким образом, являются тем, что он назвал *упрощенными вычетами*, т.е. коэффициенты являются наименьшими из тех, что можно получить без вычисления наибольших общих делителей целых чисел и без введения рациональных чисел. Другими словами, полиномиальные остатки освобождены от соответствующих *распределенных сомножителей* (используя терминологию Сильвестра). (Этот факт очевиден после нашего обсуждения теоремы Габихта, поскольку единственные коэффициенты, входящие в этот результант, — это коэффициенты исходных полиномов, а не остатков.)

Пример. Рассмотрим полиномы $p_1(x) := p(x) = x^3 - 7x + 7$ и $p_2(x) := q(x) = 3x^2 - 7$. Тогда

$$\operatorname{ress}[p(x), q(x)] = \det \begin{bmatrix} 1 & 0 & -7 & 7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 & 0 \\ 0 & 1 & 0 & -7 & 7 & 0 \\ 0 & 0 & 3 & 0 & -7 & 0 \\ 0 & 0 & 1 & 0 & -7 & 7 \\ 0 & 0 & 0 & 3 & 0 & -7 \end{bmatrix},$$

и мы можем вычислить противоположные к коэффициентам первого полиномиального остатка (степень которого $n - k = 1$), если возьмем $k = n - 1 = 2$ пар строк. Таким образом, старший коэффициент равен

$$\det \begin{bmatrix} 1 & 0 & -7 & 7 \\ 0 & 3 & 0 & -7 \\ 0 & 1 & 0 & -7 \\ 0 & 0 & 3 & 0 \end{bmatrix} = 3 \cdot (21) - 1 \cdot 21 = 42,$$

а второй коэффициент равен

$$\det \begin{bmatrix} 1 & 0 & -7 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 1 & 0 & 7 \\ 0 & 0 & 3 & -7 \end{bmatrix} = 3 \cdot (-21) = -63,$$

т.е. первый полиномиальный остаток равен $p_3(x) = 42x - 63$; отметим разницу в знаках между $p_3(x)$, вычисленным здесь, и $p_3(x)$, вычисленным с использованием евклидова алгоритма PRS.

В качестве упражнения читателю предлагается вычислить $p_4(x)$.

В общем случае, если имеется последовательность полиномиальных остатков $p_1(x), p_2(x), p_3(x), \ldots, p_h(x)$, $\deg[p_1(x)] = n$, $\deg[p_2(x)] = m$, $n \geq m$, мы можем получить (противоположные) коэффициенты $(i+1)$-го члена PRS, $i = 0, 1, 2, \ldots, h-1$, как миноры, образованные первыми $2i$ строками матрицы (S1) с поочередным приписыванием к первым $2i-1$ столбцам [$(2i) \times (2n)$-матрицы] каждого из оставшихся. Полиномы, образованные таким образом, пропорциональны полиномам, полученным с использованием соответствующих процедур для результантов в формах Брюно и Труди; более того, в случае неполных PRS полиномы следуют образцу, данному на рис. 5.2.1 (доказательство дано ниже).

До сих пор мы называли последовательность полиномиальных остатков последовательностью Штурма, если, начиная с третьего члена, коэффициенты каждого полиномиального остатка являются противоположными к коэффициентам полиномов, полученных евклидовым алгоритмом PRS; это их главная характеристика. Однако, как мы увидим в гл. 7 об отделении корней, последовательности Штурма обладают также следующим свойством: если старший коэффициент какого-то члена равен нулю, то старшие коэффициенты предыдущего и последующего членов (полиномиальных остатков) имеют противоположные знаки.

Чтобы доказать, что члены PRS, полученные описанным выше способом, составляют последовательность Штурма, нам понадобится следующая

Лемма 5.3.3 (мультипликативные правила для вычисления произведения определителя на один из его миноров). Читателю следует посмотреть (Muir, 1882, p. 118, Art. 90). Произведение определителя на любой из его миноров M может быть выражено как сумма произведений пар миноров; эти пары формируются следующим образом. Первый сомножитель произведения получается взятием q строк, включающих строки минора M, и формированием затем каждого минора порядка q, содержащего M. Второй сомножитель любого произведения — это минор, содержащий M и дополнение до первого сомножителя. Наконец, знак любого произведения задается так: второй сомножитель преобразуется таким образом, чтобы его главная диагональ совпадала с главной диагональю двух указанных входящих в него миноров, после этого нужно взять знак $+$ или $-$ в зависимости от того, будет ли сумма номеров строк и столбцов, из которых сформирован первый сомножитель, четной или нечетной.

Пример. Рассмотрим определитель

$$\det \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ b_1 & b_2 & b_3 & b_4 & b_5 \\ c_1 & c_2 & c_3 & c_4 & c_5 \\ d_1 & d_2 & d_3 & d_4 & d_5 \\ e_1 & e_2 & e_3 & e_4 & e_5 \end{bmatrix},$$

который мы обозначаем $|a_1b_2c_3d_4e_5|$, и его минор

$$\det \begin{bmatrix} b_3 & b_4 \\ c_3 & c_4 \end{bmatrix},$$

который мы обозначаем $|b_3c_4|$. Тогда для двух различных множеств из $q = 4$ строк мы имеем

$$\begin{aligned} |b_3c_4| \cdot |a_1b_2c_3d_4e_5| =& |a_1b_2c_3d_4| \cdot |b_3c_4e_5| - |a_1b_3c_4d_5| \cdot |b_3c_4e_2| \\ &+ |a_2b_3c_4d_5| \cdot |b_3c_4e_1| \\ =& - |a_1b_2c_3e_4| \cdot |b_3c_4d_5| + |a_1b_3c_4e_5| \cdot |b_3c_4d_2| \\ &- |a_2b_3c_4e_5| \cdot |b_3c_4d_1| \\ =& \dots . \end{aligned}$$

Теорема 5.3.4 (Van Vleck, 1899). Рассмотрим в $\mathbb{Z}[x]$ полиномы $p_1(x) = c_nx^n + c_{n-1}x^{n-1} + \cdots + c_0$ и $p_2(x) = d_mx^m + d_{m-1}x^{m-1} + \cdots + d_0$, $c_n \neq 0$, $d_m \neq 0$, $n \geq m$. Тогда последовательность полиномов, образованных из первых $2i$ строк матрицы (S1), $i = 1, 2, \dots, n$, является последовательностью Штурма.

Доказательство. Прежде всего заметим, что знаки старших коэффициентов полиномов, образованных из (S1), согласуются со знаками соответствующих коэффициентов в остатках Штурма. Поэтому нам достаточно доказать, что если старший коэффициент какого-либо члена равен нулю, то старшие коэффициенты предыдущего и последующего полиномиальных остатков имеют противоположные знаки. Для этого рассмотрим старшие коэффициенты любых трех последовательных полиномиальных остатков $D_{2k-2}, D_{2k}, D_{2k+2}$, где D_k обозначает определитель $k \times k$-матрицы, образованной первыми k строками и столбцами матрицы (S1), и умножим первый остаток на последний:

$$D_{2k-2} \cdot D_{2k+2}.$$

Поскольку D_{2k-2} — минор матрицы D_{2k+2}, мы можем воспользоваться леммой 5.3.3. Легко видеть, что первый определитель D_{2k-2}

можно взять в качестве минора матрицы D_{2k+2} несколькими способами. Один из них состоит в том, чтобы опустить две первые и две последние строки, а также первый и три последних столбца. Затем мы выбираем в качестве q наших строк все строки с третьей по последнюю. Из определителей порядка q, которые можно сформировать, чтобы получить D_{2k-2}, все кроме трех содержат первый столбец, содержащий только нулевые элементы, следовательно, они обращаются в нуль. Оставшиеся три определителя получены отбрасыванием кроме первого столбца еще последнего столбца, или предпоследнего, или предпредпоследнего. Так получаются первые сомножители из множества произведений пар миноров; заметим, что первый равен D_{2k}, а третий по лемме 5.3.3 должен быть умножен на D_{2k}. Следовательно, если $D_{2k} = 0$, то нужно рассмотреть единственное произведение — оставшийся определитель, умноженный на множитель, который, как легко видеть, должен быть отрицательным. Поэтому $D_{2k-2} \cdot D_{2k+2} < 0$, если $D_{2k} = 0$. □

Из приведенных выше рассуждений видно, что результант в форме Сильвестра может дать нам последовательность Штурма двух полиномов (последовательность их полиномальных остатков, домножаемых на -1), являющуюся в общем случае полной последовательностью полиномиальных остатков. Более того, и это наиболее существенно: для того чтобы найти коэффициенты, *не требуется вычислять определители*. Это следует из того, что в исходном определителе (S1) элементы любых двух последовательных строк совпадают с элементами двух предыдущих строк. Таким образом, если элементы какой-то строки изменены путем добавления кратного предыдущей строки, то такая же замена может быть выполнена с элементами всех соответствующих строк через одну без изменения значения минора, появляющегося в качестве коэффициента в одном из полиномов последовательности Штурма. Поэтому можно привести соответствующую матрицу к верхней треугольной форме, пользуясь алгоритмом Доджсона (D) (см. разд. 5.3.1), не портя периодическую структуру определителя. Таким образом, имеет место следующая

Теорема 5.3.5 (Van Vleck, 1899). Если мы приведем $\mathbf{M}_{\mathrm{S}}$, матрицу, соответствующую результанту в форме Сильвестра, к верхней треугольной форме $T(\mathbf{M}_{\mathrm{S}})$, то *четные* строки матрицы $T(\mathbf{M}_{\mathrm{S}})$ дают коэффициенты последовательных полиномиальных остатков (Штурма). Коэффициенты, взятые из данной строки, умножаются на $(-1)^k$, где k — число отрицательных «составляющих» на главной диагонали выше рассматриваемой строки.

Ван Влек пользовался этой теоремой для вычисления полных последовательностей полиномиальных остатков, обновляя одновременно только три строки и удаляя из каждого остатка наибольший общий делитель коэффициентов, вычисления которого мы хотим избежать. Однако он не рассматривал задачу выбора ведущего элемента. В этом случае требуется проявлять осторожность в определении знака коэффициентов при осуществлении выбора ведущего элемента, поскольку этот выбор может менять знаки. Рассмотрим следующий пример.

Пример. Пусть $p_1(x) = x^3 - 7x + 7$ и $p_2(x) = 3x^2 - 7$; тогда

$$\mathbf{M}_{\mathrm{S}} = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 & 0 \\ 0 & 1 & 0 & -7 & 7 & 0 \\ 0 & 0 & 3 & 0 & -7 & 0 \\ 0 & 0 & 1 & 0 & -7 & 7 \\ 0 & 0 & 0 & 3 & 0 & -7 \end{bmatrix} \Rightarrow T(\mathbf{M}_{\mathrm{S}}) = \begin{bmatrix} 1 & 0 & -7 & 7 & 0 & 0 \\ 0 & 3 & 0 & -7 & 0 & 0 \\ 0 & 0 & 9 & 0 & -21 & 0 \\ 0 & 0 & 0 & -42 & 63 & 0 \\ 0 & 0 & 0 & 0 & 196 & -294 \\ 0 & 0 & 0 & 0 & 0 & -49 \end{bmatrix} *$$

Помеченная $*$ строка означает, что при выборе ведущего элемента поменялись местами третья и четвертая строки, а это значит, что остатки Штурма суть $42x - 63$ и 49. Читателю следует *вручную* выполнить соответствующую последовательность операций и заметить, что до выбора ведущего элемента мы получили в третьей строке элементы -14 и 21. Это ключевое наблюдение для алгоритмической реализации метода матричной триангуляризации субрезультантных PRS.

Замечание. Если мы рассмотрим строки обеих матриц этого примера, то увидим, что каждая из них соответствует данному полиному, степень которого на единицу меньше числа элементов, *заключенных* между левыми и/или правыми нулями. Небольшая сложность, которая может возникнуть при такой реализации, состоит в том, что один или несколько коэффициентов при младших степенях x могут быть нулевыми. В этом случае степень полинома легко определить, рассматривая все строки верхней треугольной матрицы. Однако эта проблема не возникает, если мы изменим представление, используя, например, связные списки.

Все, что мы до сих пор сказали, относится к полным последовательностям полиномиальных остатков. Для того чтобы иметь дело с неполными PRS, нам понадобится следующая теорема, являющаяся нашим обобщением теоремы 5.3.5.

Теорема 5.3.6 (Akritas, 1986). Пусть $p_1(x)$ и $p_2(x)$ — два полинома степеней n и m соответственно, $n \geq m$. Пользуясь алгоритмом Доджсона, приведем матрицу $\mathbf{M}_{\mathrm{S}}$, соответствующую $\mathrm{res}_{\mathrm{S}}[p_1(x), p_2(x)]$, к верхней треугольной форме $T(\mathbf{M}_{\mathrm{S}})$. Пусть n_i — степень полинома, соответствующего i-й строке матрицы $T(\mathbf{M}_{\mathrm{S}})$, $i = 1, 2, \ldots, 2n$, и пусть $p_k(x)$, $k \geq 2$, — k-й член (*полной или неполной*) последовательности полиномиальных остатков для $p_1(x)$ и $p_2(x)$. Тогда если $p_k(x)$ находится в i-й строке матрицы $T(\mathbf{M}_{\mathrm{S}})$, то коэффициенты полинома $p_{k+1}(x)$ получаются из $i+j$-й строки матрицы $T(\mathbf{M}_{\mathrm{S}})$, где j — наименьшее целое, такое, что $n_{i+j} < n_i$. [Если $n = m$, то и $p_1(x)$, и $p_2(x)$ ассоциируются с первой строкой матрицы $T(\mathbf{M}_{\mathrm{S}})$.]

Доказательство. Доказательство основано на том, что элементы любых двух последовательных строк в $\mathbf{M}_{\mathrm{S}}$ те же самые, что в предыдущих строках, и что выполняется выбор ведущего элемента методом пузырька. Предположим, что в ходе триангуляризации мы получили k-й член PRS, $p_k(x)$, из строки i, так что $\deg[p_k(x)] = n_i$, и что мы довели триангуляризацию до этой строки. На следующем шаге процесса строка i не меняется, но предположим, что $i+1$-я строка изменилась таким образом, что $n_{i+1} = n_i - d$, $d > 1$, и диагональный элемент равен 0; пусть $p_{k+1}(x)$ — полином степени $n_i - d$, соответствующий этой строке. Этот полином является $(k+1)$-м членом PRS, но его положение в $T(\mathbf{M}_{\mathrm{S}})$ меняется, поскольку выполняется выбор ведущего элемента методом пузырька, после чего $n_{i+1} = n_i$ (обоснуйте это). Этот процесс повторяется до тех пор, пока старший коэффициент полинома $p_{k+1}(x)$ степени $n_i - d$ не появится в точности на диагонали и нам не потребуется больше выбирать ведущий элемент. Таким образом, в $T(\mathbf{M}_{\mathrm{S}})$ за несколькими строками, соответствующими полиномам степени n_i, будет следовать несколько строк, соответствующих полиномам степени $n_i - d$. Первый из таких полиномов — это $p_{k+1}(x)$, коэффициенты которого (к этому времени) значительно выросли в результате большого количества преобразований. Поэтому если мы хотим, чтобы коэффициенты полинома $p_{k+1}(x)$ были наименьшими, мы должны *запомнить* его, когда мы впервые встречаем его в строке $i+1$ (где впервые появляется разность степеней), и использовать его в запомненном варианте, а не в том, в котором он появляется в $T(\mathbf{M}_{\mathrm{S}})$. Читателю в качестве упражнения предлагается закончить доказательство, т.е. определить, из какой строки будут получены коэффициенты полинома $p_{k+2}(x)$. □

Заметим, что в качестве частного случая теоремы 5.3.6 мы получаем теорему 5.3.5 для полных PRS. Мы видим, таким образом,

что на базе теоремы 5.3.6 мы имеем другой метод субрезультантных PRS для вычисления последовательности полиномиальных остатков и наибольшего общего делителя двух полиномов. Этот метод, в котором *не* выполняется явное деление, с единой позиции рассматривает как полные, так и неполные PRS и дает наименьшие коэффициенты, которые можно ожидать без вычисления наибольших общих делителей коэффициентов. Как мы уже указывали в разд. 5.1.1, в некоторых случаях коэффициенты *меньше* тех, которые получаются при использовании метода Сильвестра–Габихта псевдоделения субрезультантных PRS.

Мы имеем следующий алгоритм (примеры были даны в разд. 5.1.1).

ASPRS. Матричная триангуляризация субрезультантных PRS (Matrix-Triangularization Subresultant **PRS**)

Вход: Два полинома в $\mathbb{Z}[x]$, $p_1(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0$ и $p_2(x) = d_m x^m + d_{m-1}x^{m-1} + \cdots + d_0$, $c_n \neq 0$, $d_m \neq 0$, $n \geq m$.

Выход: Субрезультантная PRS полиномов $p_1(x)$ и $p_2(x)$.

1. [Инициализация] Сформировать матрицу $\mathbf{M}_{\mathrm{S}}$, соответствующую результанту $\mathrm{res}_{\mathrm{S}}[p_1(x), p_2(x)]$.

2. [Основной шаг] Используя алгоритм Доджсона (D), описанный выше, привести $\mathbf{M}_{\mathrm{S}}$ к верхней треугольной форме $T(\mathbf{M}_{\mathrm{S}})$. Тогда члены PRS получаются из строк матрицы $T(\mathbf{M}_{\mathrm{S}})$ согласно теореме 5.3.6.

Анализ времени работы алгоритма ***ASPRS***. Мы знаем, что если n – наибольшая из степеней двух полиномов $p_1(x)$ и $p_2(x)$, то матрица Сильвестра $\mathbf{M}_{\mathrm{S}}$ будет иметь размер $2n$ на $2n$. Не пользуясь преимуществами ленточной формы матрицы, мы видим, что для приведения матрицы $\mathbf{M}_{\mathrm{S}}$ к верхней треугольной форме (используя метод Доджсона) нам нужно выполнить следующие операции. На i-м шаге для *каждой* из оставшихся $2n - i$ строк мы выполняем $2(2n - i)$ целочисленных умножения и $(2n - i)$ вычитаний и делений для определения получающихся элементов, которые не обязательно равны нулю. Таким образом, общее число операций в процессе триангуляризации равно

$$O\left[\sum_{1 \leq i \leq 2n-1} (2n - i)(2n - i + 1)\right].$$

Пользуясь формулами

$$\sum_{1\le i\le k} i = \frac{k(k+1)}{2}, \quad \sum_{1\le i\le k} i^2 = \frac{k(k+1)(k+2)}{6},$$

мы видим, что общее число целочисленных умножений равно $O(n^3)$. (Мы считаем только умножения, поскольку это наиболее дорогие операции.)

Поскольку мы имеем дело с точной целочисленной арифметикой, мы предполагаем, что в худшем случае каждое целочисленное умножение выполняется над наибольшими целыми числами, которые могут возникнуть. Эти наибольшие числа могут достигать значения $\mathrm{res}_S[p_1(x), p_2(x)]$. Пользуясь неравенством Адамара, получаем

$$|\,\mathrm{res}_S[p_1(x), p_2(x)]| \le \prod_{1\le i\le 2n} \left(\sum_{1\le i\le 2n} r_{ij}^2 \right)^{1/2} \le (2n)^n [|p(x)|_\infty]^{2n},$$

где $|p(x)|_\infty = \max(|p_1(x)|_\infty, |p_2(x)|_\infty)$. Таким образом, время умножения $(2n)^n \cdot (|p(x)|_\infty)^{2n}$ на себя равно

$$L^2\{(2n)^n[|p(x)|_\infty]^{2n}\} = \{nL(2n) + 2nL(|p(x)|_\infty)]\}^2 \sim n^2 L^2[|p(x)|_\infty],$$

а поскольку имеется $O(n^3)$ таких умножений, мы получаем

$$t_{\mathbf{ASPRS}}[p_1(x), p_2(x)] = O\{n^5 L^2[|p(x)|_\infty]\}.$$

Таким образом, мы видим, что теоретически время работы метода матричной триангуляризации субрезультантных PRS больше времени работы метода Сильвестра–Габихта псевдоделения субрезультантных PRS. В следующем разделе мы дадим некоторые эмпирические результаты и сравнения.

5.4

Эмпирическое сравнение двух методов субрезультантных PRS

В этом разделе мы рассматриваем десять примеров (Anderson, 1986). Каждый раз мы начинаем с краткого рассмотрения существенных аспектов данного примера, а затем следуют последовательности полиномиальных остатков, порожденные каждым методом субрезультантых PRS. Для метода Сильвестра–Габихта псевдоделения

мы даем две (иногда различные) последовательности, получающиеся из двух различных алгоритмов, обсуждавшихся в разд. 5.2. Число в скобках в правом конце каждой строки — это степень полинома; для метода матричной триангуляризации этот полином соответствует заданной строке. Выписываются только коэффициенты при различных степенях x, первым появляется коэффициент, соответствующий наивысшей степени x.

В конце каждого метода приводится наибольшее (по абсолютной величине) целое число, получающееся при псевдоделении, в скобках дается количество цифр в нем. Это наибольшее значение, порождаемое соответствующим методом для соответствующего примера. В алгоритмах PRS Сильвестра (S′) и Габихта (H) это значение должно размещаться в памяти до того, как его разделят. В методе матричной триангуляризации это значение делится до того, как размещается в памяти. Это целое число приводится, потому что коэффициенты членов PRS могут быть достаточно малыми, хотя в действительности для их получения использовались целые числа гораздо большего размера.

Результаты, полученные с помощью алгоритмов PRS Сильвестра и Габихта, просто отражают порождаемые последовательности полиномиальных остатков. Метод матричной триангуляризации, однако, требует приведения $\mathbf{M}_{\mathrm{S}}$, матрицы, соответствующей результанту в форме Сильвестра, к верхней треугольной форме $T(\mathbf{M}_{\mathrm{S}})$. Дается только $T(\mathbf{M}_{\mathrm{S}})$, в самом левом столбце указываются номера строк. Как мы уже отмечали, каждой строке соответствует полином, степень которого на единицу меньше числа элементов, *заключенных* между старшими и/или конечными нулями. Символ «#», предшествующий номеру строки, указывает, что для этой строки осуществлялся выбор ведущего элемента. Для определения членов PRS мы просматриваем сверху вниз самый правый столбец (степени) и из данного множества строк «степени d» выбираем полином, соответствующий первой из них, в качестве следующего члена PRS. Знак «>» после номера строки (в самом левом столбце) означает, что эта строка может быть членом PRS *при условии, что не осуществляется выбор ведущего элемента*. Если выполняются преобразования, связанные с выбором ведущего элемента, то соответствующая строка запоминается и распечатывается под матрицей $T(\mathbf{M}_{\mathrm{S}})$, а также отмечается знаком «>» после номера строки. В этом случае мы сравниваем коэффициенты запомненной строки с коэффициентами первой строки в $T(\mathbf{M}_{\mathrm{S}})$, имеющей ту же степень, и выбираем ту, коэффициенты которой меньше, в качестве нового члена PRS. (См. также доказательство теоремы

5.3.6 и историческое замечание 4.)

Пример 1. Это классический пример неполной PRS, используемый в литературе Кнутом (Knuth, 1981) и Брауном (Brown, 1971).

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$\begin{aligned} p(1) &= 1\ 0\ 1\ 0\ -3\ -3\ 8\ 2\ -5 && (8)\\ p(2) &= 3\ 0\ 5\ 0\ -4\ -9\ 21 && (6)\\ p(3) &= -15\ 0\ 3\ 0\ -9 && (4)\\ p(4) &= 585\ 1125\ -2205 && (2)\\ p(5) &= -18885150\ 24907500 && (1)\\ p(6) &= 527933700 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 786410803602112500 (18 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$\begin{aligned} p(1) &= 1\ 0\ 1\ 0\ -3\ -3\ 8\ 2\ -5 && (8)\\ p(2) &= 3\ 0\ 5\ 0\ -4\ -9\ 21 && (6)\\ p(3) &= -15\ 0\ 3\ 0\ -9 && (4)\\ p(4) &= -65\ -125\ 245 && (2)\\ p(5) &= -9326\ 12300 && (1)\\ p(6) &= 260708 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 21308697620 (11 цифр).

Метод матричной триангуляризации субрезультантных PRS

$$
\begin{array}{rl}
1> & 1\ 0\ 1\ 0\ -3\ -3\ 8\ 2\ -5\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \quad (8)\\
\#2) & 0\ 1\ 0\ 1\ 0\ -3\ -3\ 8\ 2\ -5\ 0\ 0\ 0\ 0\ 0\ 0 \quad (8)\\
3> & 0\ 0\ 3\ 0\ 5\ 0\ -4\ -9\ 21\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \quad (6)\\
4) & 0\ 0\ 0\ 9\ 0\ 15\ 0\ -12\ -27\ 63\ 0\ 0\ 0\ 0\ 0\ 0 \quad (6)\\
5) & 0\ 0\ 0\ 0\ -6\ 0\ -15\ 0\ 9\ 18\ -45\ 0\ 0\ 0\ 0\ 0 \quad (6)\\
\#6) & 0\ 0\ 0\ 0\ 0\ 4\ 0\ 10\ 0\ -6\ -12\ 30\ 0\ 0\ 0\ 0 \quad (6)\\
7> & 0\ 0\ 0\ 0\ 0\ 0\ 0\ -10\ 0\ 2\ 0\ -6\ 0\ 0\ 0\ 0\ 0 \quad (4)\\
8) & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 25\ 0\ -5\ 0\ 15\ 0\ 0\ 0\ 0 \quad (4)\\
9) & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -45\ 0\ 35\ 50\ -125\ 0\ 0\ 0 \quad (4)\\
\#10) & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 81\ 0\ -63\ -90\ 225\ 0\ 0 \quad (4)\\
11> & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -117\ -225\ 441\ 0\ 0\ 0 \quad (2)\\
12) & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 169\ 325\ -637\ 0\ 0 \quad (2)\\
13) & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -2035\ 2543\ -845\ 0 \quad (2)\\
14> & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 9326\ -12300\ 0 \quad (1)\\
15) & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -7778\ -46630 \quad (1)\\
16> & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -260708 \quad (0)
\end{array}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 2431362808 (10 цифр). При преобразовании 2, выбирая ведущий элемент, изменяем строку 2. Запоминаем строку

$$2>\ 0\ 0\ 3\ 0\ 5\ 0\ -4\ -9\ 21\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \quad (6)$$

При преобразовании 6, выбирая ведущий элемент, изменяем строку 6. Запоминаем строку

$$6>\ 0\ 0\ 0\ 0\ 0\ 0\ 15\ 0\ -3\ 0\ 9\ 0\ 0\ 0\ 0\ 0 \quad (4)$$

При преобразовании 10, выбирая ведущий элемент, изменяем строку 10. Запоминаем строку

$$10>\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 65\ 125\ -225\ 0\ 0\ 0 \quad (2)$$

Читателю следует отметить, что в данном примере член второй степени PRS получается не из матрицы $T(\mathbf{M}_S)$, а из запомненной строки, напечатанной под ней. Этот факт в следующих примерах не подчеркивается.

Пример 2. Это — небольшой пример неполной PRS.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$\begin{aligned} p(1) &= 2\ 5\ 5\ -2\ 1 && (4)\\ p(2) &= 3\ 3\ 3\ -4 && (3)\\ p(3) &= -21\ 45 && (1)\\ p(4) &= -47049 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 37044 (5 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$\begin{aligned} p(1) &= 2\ 5\ 5\ -2\ 1 && (4)\\ p(2) &= 3\ 3\ 3\ -4 && (3)\\ p(3) &= -21\ 45 && (1)\\ p(4) &= 15683 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 37044 (5 цифр).

Метод матричной триангуляризации субрезультантных PRS

$$\begin{aligned} 1 &> 2\ 5\ 5\ -2\ 1\ 0\ 0\ 0 && (4)\\ 2 &> 0\ 3\ 3\ 3\ -4\ 0\ 0\ 0 && (3)\\ 3) &\ 0\ 0\ 9\ 9\ 2\ 3\ 0\ 0 && (3)\\ \#4) &\ 0\ 0\ 0\ 27\ 41\ -24\ 9\ 0 && (3)\\ 5 &> 0\ 0\ 0\ 0\ 63\ -135\ 0\ 0 && (1)\\ 6) &\ 0\ 0\ 0\ 0\ 0\ 147\ -315\ 0 && (1)\\ 7) &\ 0\ 0\ 0\ 0\ 0\ 0\ 3068\ 147 && (1)\\ 8 &> 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -15683 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 2305401 (7 цифр). При преобразовании 4, выбирая ведущий элемент, изменяем строку 4. Запоминаем строку

$$4 >\ 0\ 0\ 0\ 0\ 21\ -45\ 0\ 0 \quad (1)$$

Пример 3. Пример полной PRS, где $\deg[p_1(x)] = \deg[p_2(x)]$; заметим, что свободный член полинома $p_3(x)$ равен нулю.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$
\begin{aligned}
p(1) &= 2\ 1\ 4\ 3\ 5\ -2\ 2 && (6)\\
p(2) &= 1\ -1\ -1\ 4\ -2\ 1\ 1 && (6)\\
p(3) &= 3\ 6\ -5\ 9\ -4\ 0 && (5)\\
p(4) &= 60\ -36\ 75\ -27\ 9 && (4)\\
p(5) &= -1628\ 240\ -376\ -468 && (3)\\
p(6) &= 42067\ -27959\ 12373 && (2)\\
p(7) &= -304996\ -147119 && (1)\\
p(8) &= 1873839 && (0)
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 3913180032193072 (16 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$
\begin{aligned}
p(1) &= 2\ 1\ 4\ 3\ 5\ -2\ 2 && (6)\\
p(2) &= 1\ -1\ -1\ 4\ -2\ 1\ 1 && (6)\\
p(3) &= 3\ 6\ -5\ 9\ -4\ 0 && (5)\\
p(4) &= 60\ -36\ 75\ -27\ 9 && (4)\\
p(5) &= -1628\ 240\ -376\ -468 && (3)\\
p(6) &= 42067\ -27959\ 12373 && (2)\\
p(7) &= -304996\ -147119 && (1)\\
p(8) &= 1873839 && (0)
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 3913180032193072 (16 цифр).

Метод матричной триангуляризации субрезультантных PRS

$$
\begin{aligned}
&1 > 2\ 1\ 4\ 3\ 5\ -2\ 2\ 0\ 0\ 0\ 0\ 0 && (6)\\
&2 > 0\ -3\ -6\ 5\ -9\ 4\ 0\ 0\ 0\ 0\ 0\ 0 && (5)\\
&\ \ 3)\ 0\ 0\ 9\ -22\ 9\ -23\ 6\ -6\ 0\ 0\ 0\ 0 && (5)
\end{aligned}
$$

$$
\begin{aligned}
&4)\ 0\ 0\ 0\ -60\ 36\ -75\ 27\ -9\ 0\ 0\ 0\ 0 && (4)\\
&5)\ 0\ 0\ 0\ 0\ -332\ -45\ -379\ 93\ -120\ 0\ 0\ 0 && (4)\\
&6>\ 0\ 0\ 0\ 0\ 0\ 0\ -1628\ 240\ -376\ -468\ 0\ 0\ 0 && (3)\\
&7)\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -2549\ -8203\ 5113\ -3256\ 0\ 0 && (3)\\
&8>\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 42067\ -27959\ 12373\ 0\ 0 && (2)\\
&9)\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 255739\ -151491\ 84134\ 0 && (2)\\
&10>\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 304996\ 147119\ 0 && (1)\\
&11)\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -1992731\ 609992 && (1)\\
&12>\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -1873839 && (0)
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 571513399644 (12 цифр). Заметим, что полином $p_2(x)$ не появляется в $T(\mathbf{M}_S)$.

Пример 4. Пример полной PRS, использованный Ван Влеком.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$
\begin{aligned}
p(1) &= 1\ 1\ -1\ -1\ 1\ -1\ 1 && (6)\\
p(2) &= 6\ 5\ -4\ -3\ 2\ -1 && (5)\\
p(3) &= -17\ -14\ 27\ -32\ 37 && (4)\\
p(4) &= 44\ -114\ 120\ -7 && (3)\\
p(5) &= -516\ 828\ 186 && (2)\\
p(6) &= 9108\ -3114 && (1)\\
p(7) &= 127359 && (0)
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 68687289792 (11 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$
\begin{aligned}
p(1) &= 1\ 1\ -1\ -1\ 1\ -1\ 1 && (6)\\
p(2) &= 6\ 5\ -4\ -3\ 2\ -1 && (5)\\
p(3) &= -17\ -14\ 27\ -32\ 37 && (4)\\
p(4) &= 44\ -114\ 120\ -7 && (3)\\
p(5) &= -516\ 828\ 186 && (2)\\
p(6) &= 9108\ -3114 && (1)\\
p(7) &= 127359 && (0)
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 68687289792 (11 цифр).

Метод матричной триангуляризации субрезультантных PRS

$$
\begin{aligned}
1 &> 1\ 1\ -1\ -1\ 1\ -1\ 1\ 0\ 0\ 0\ 0\ 0 && (6)\\
2 &> 0\ 6\ 5\ -4\ -3\ 2\ -1\ 0\ 0\ 0\ 0\ 0 && (5)\\
3) &\ 0\ 0\ 1\ -2\ -3\ 4\ -5\ 6\ 0\ 0\ 0\ 0 && (5)\\
4 &> 0\ 0\ 0\ 17\ 14\ -27\ 32\ -37\ 0\ 0\ 0\ 0 && (4)\\
5) &\ 0\ 0\ 0\ 0\ -8\ -4\ 6\ -8\ 17\ 0\ 0\ 0 && (4)\\
6 &> 0\ 0\ 0\ 0\ 0\ -44\ 114\ -120\ 7\ 0\ 0\ 0 && (3)\\
7) &\ 0\ 0\ 0\ 0\ 0\ 0\ 64\ -72\ 24\ -44\ 0\ 0 && (3)\\
8) &\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -516\ 828\ 186\ 0\ 0 && (2)\\
9) &\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 360\ 552\ -516\ 0 && (2)\\
10 &> 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 9108\ -3114\ 0 && (1)\\
11) &\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -11916\ 9108 && (1)\\
12 &> 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -127359 && (0)
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 1159985772 (10 цифр).

Пример 5. В этом примере неполной PRS имеется уменьшение степени ее членов на 3 (с 7 до 4), и это единственный представленный пример, требующий выбора ведущего элемента методом пузырька в методе матричной триангуляризации, т.е. в выборе ведущего элемента участвуют строки, не являющиеся соседними.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$
\begin{aligned}
p(1) &= 3\ 5\ 7\ -3\ -5\ -7\ 3\ 5\ 7\ -2 && (9)\\
p(2) &= 1\ 0\ 0\ -1\ 0\ 0\ -1\ -1\ -1 && (8)\\
p(3) &= 7\ 0\ 0\ -7\ 6\ 13\ 15\ 3 && (7)\\
p(4) &= -42\ -91\ -154\ -70\ -49 && (4)\\
p(5) &= -3146045\ -3785054\ -1840832\ -1391747 && (3)\\
p(6) &= -256801797\ -88236211\ -98913311 && (2)\\
p(7) &= 1999255468\ -2333396170 && (1)\\
p(8) &= -33438950636 && (0)
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 10264425417303045805977461 28 (28 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$p(1) = 3\ 5\ 7\ -3\ -5\ -7\ 3\ 5\ 7\ -2 \qquad (9)$$
$$p(2) = 1\ 0\ 0\ -1\ 0\ 0\ -1\ -1\ -1 \qquad (8)$$
$$p(3) = 7\ 0\ 0\ -7\ 6\ 13\ 15\ 3 \qquad (7)$$
$$p(4) = -42\ -91\ -154\ -70\ -49 \qquad (4)$$
$$p(5) = -64205\ -77246\ -37568\ -28403 \qquad (3)$$
$$p(6) = -5240853\ -1800739\ -2018639 \qquad (2)$$
$$p(7) = 40801132\ -47620330 \qquad (1)$$
$$p(8) = -682427564 \qquad (0)$$

Наибольшее целое число, порождаемое алгоритмом, равно 8724617648516388414672 (22 цифры).

Метод матричной триангуляризации субрезультантных PRS

1 > 3 5 7 −3 −5 −7 3 5 7 −2 0 0 0 0 0 0 0 0 (9)
2 > 0 1 0 0 −1 0 0 −1 −1 −1 0 0 0 0 0 0 0 0 0 (8)
3) 0 0 5 7 0 −5 −7 6 8 10 −2 0 0 0 0 0 0 0 0 (8)
4 > 0 0 0 −7 0 0 7 −6 −13 −15 −3 0 0 0 0 0 0 0 0 (7)
5) 0 0 0 0 −49 0 0 79 23 19 −55 14 0 0 0 0 0 0 0 (7)
#6) 0 0 0 0 0 −343 0 −24 501 73 93 −413 98 0 0 0 0 0 (7)
#7) 0 0 0 0 0 0 −2401 −510 −1273 1637 −339 56 −2891 686 0 0 0 0 (7)
8 > 0 0 0 0 0 0 0 2058 4459 7546 3430 2401 0 0 0 0 0 0 0 (4)
9) 0 0 0 0 0 0 0 0 0 −1764 −3822 −6468 −2940 −2058 0 0 0 0 0 (4)
10) 0 0 0 0 0 0 0 0 0 0 1512 3276 5544 2520 1764 0 0 0 0 (4)
11) 0 0 0 0 0 0 0 0 0 0 0 25811 −18982 4520 −811 −3024 0 0 0 (4)
12 > 0 0 0 0 0 0 0 0 0 0 0 0 −64205 −77246 −37568 −28403 0 0 0 (3)
13) 0 0 0 0 0 0 0 0 0 0 0 0 0 2124693 449379 519299 128410 0 0 (3)
14 > 0 0 0 0 0 0 0 0 0 0 0 0 0 0 −5240853 −1800739 −2018639 0 0 (2)
15) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 −22909248 −24412716 10481706 0 (2)
16 > 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 −40801132 47620330 0 (1)
17) 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 −398219984 81602264 (1)
18 > 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 682427564 (0)

Наибольшее целое число, порождаемое алгоритмом, равно 27843817119202448 (17 цифр). При преобразовании 6, выбирая ведущий элемент, изменяем строку 6. Запоминаем строку

$$6 > \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 42\ 91\ 154\ 70\ 49\ 0\ 0\ 0\ 0\ 0\ 0 \quad (4)$$

При преобразовании 7, выбирая ведущий элемент, изменяем строку 7. Запоминаем строку

$$7)\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 294\ 637\ 1078\ 490\ 343\ 0\ 0\ 0\ 0\ 0\ 0 \quad (4)$$

Заметим, что, поскольку степень запоминаемой строки 7 совпадает со степенью запоминаемой строки 6, последняя не является членом PRS.

Пример 6. Это — пример PRS, в которой $\deg[p_1(x)]$ и $\deg[p_2(x)]$ отличаются более, чем на 1.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$\begin{aligned}
p(1) &= 3\ \ 4\ \ 0\ \ -7\ \ 0\ \ -9\ \ 1\ \ -1\ \ 2 && (8)\\
p(2) &= 3\ \ 12\ \ 0\ \ -7\ \ 0\ \ 1 && (5)\\
p(3) &= 39960\ \ 5238\ \ -23895\ \ -945\ \ 3618 && (4)\\
p(4) &= 5371812\ \ 226170\ \ 56430\ \ -1003068 && (3)\\
p(5) &= -442106115\ \ 117089757\ \ 77380134 && (2)\\
p(6) &= 9708643482\ \ -4839461901 && (1)\\
p(7) &= 12487585839 && (0)
\end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 41671931313194660832971041260 (29 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$\begin{aligned}
p(1) &= 3\ \ 4\ \ 0\ \ -7\ \ 0\ \ -9\ \ 1\ \ -1\ \ 2 && (8)\\
p(2) &= 3\ \ 12\ \ 0\ \ -7\ \ 0\ \ 1 && (5)\\
p(3) &= 39960\ \ 5238\ \ -23895\ \ -945\ \ 3618 && (4)\\
p(4) &= 5371812\ \ 226170\ \ 56430\ \ -1003068 && (3)\\
p(5) &= -442106115\ \ 117089757\ \ 77380134 && (2)\\
p(6) &= 9708643482\ \ -4839461901 && (1)\\
p(7) &= 12487585839 && (0)
\end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 4167193131319466083297104126O (29 цифр).

Метод матричной триангуляризации субрезультантных PRS

1 >3 4 0 −7 0 −9 1 −1 2 0 0 0 0 0 0 0 (8)

#2)0 3 4 0 −7 0 −9 1 −1 2 0 0 0 0 0 0 (8)

#3)0 0 3 4 0 −7 0 −9 1 −1 2 0 0 0 0 0 (8)

4 >0 0 0 3 12 0 −7 0 1 0 0 0 0 0 0 0 (5)

5)0 0 0 0 9 36 0 −21 0 3 0 0 0 0 0 0 (5)

6)0 0 0 0 0 27 108 0 −63 0 9 0 0 0 0 0 (5)

7)0 0 0 0 0 0 −3456 −504 1746 99 −315 54 0 0 0 0 (5)

8 >0 0 0 0 0 0 0 −39960 −5238 23895 945 −3618 0 0 0 0 (4)

9)0 0 0 0 0 0 0 0 75456 474480 −25560 3096 −79920 0 0 0 (4)

10 >0 0 0 0 0 0 0 0 0 −5371812 −226170 −56430 1003068 0 0 0 (3)

11)0 0 0 0 0 0 0 0 0 0 63357144 −3542580 2310276 −10743624 0 0 (3)

12 >0 0 0 0 0 0 0 0 0 0 0 −442106115 117089757 77380134 0 0 (2)

13)0 0 0 0 0 0 0 0 0 0 0 0 1089441759 1102788303 −884212230 0 (2)

14 >0 0 0 0 0 0 0 0 0 0 0 0 0 9708643482 −4839461901 0 (1)

15)0 0 0 0 0 0 0 0 0 0 0 0 0 0 −36142658547 19417286964 (1)

16 >0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 −12487585839 (0)

Наибольшее целое число, порождаемое алгоритмом, равно 1212375188617228513 98 (21 цифра). При преобразовании 2, выбирая ведущий элемент, изменяем строку 2. Запоминаем строку

$$2 > \ 0\ 0\ 0\ 3\ 12\ 0\ -7\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \quad (5)$$

При преобразовании 3, выбирая ведущий элемент, изменяем строку 3. Запоминаем строку

$$3)\ 0\ 0\ 0\ 3\ 12\ 0\ -7\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \quad (5)$$

Пример 7. Еще один пример небольшой неполной PRS.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$\begin{aligned} p(1) &= 1\ 5\ 10\ 5\ 5\ 2 && (5)\\ p(2) &= 1\ 4\ 6\ 2\ 1 && (4)\\ p(3) &= -3\ 2\ 1 && (2)\\ p(4) &= -266\ -112 && (1)\\ p(5) &= 980 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 212268 (6 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$\begin{aligned} p(1) &= 1\ 5\ 10\ 5\ 5\ 2 && (5)\\ p(2) &= 1\ 4\ 6\ 2\ 1 && (4)\\ p(3) &= -3\ 2\ 1 && (2)\\ p(4) &= -266\ -112 && (1)\\ p(5) &= 980 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 212268 (6 цифр).

Метод матричной триангуляризации субрезультантных PRS

$$\begin{aligned} 1> &\ 1\ 5\ 10\ 5\ 5\ 2\ 0\ 0\ 0\ 0 && (5)\\ 2> &\ 0\ 1\ 4\ 6\ 2\ 1\ 0\ 0\ 0\ 0 && (4)\\ 3) &\ 0\ 0\ 1\ 4\ 3\ 4\ 2\ 0\ 0\ 0 && (4)\\ \#4) &\ 0\ 0\ 0\ 1\ 7\ 1\ 3\ 2\ 0\ 0 && (4)\\ 5> &\ 0\ 0\ 0\ 0\ 3\ -2\ -1\ 0\ 0\ 0 && (2)\\ 6) &\ 0\ 0\ 0\ 0\ 0\ 9\ -6\ -3\ 0\ 0 && (2)\\ 7) &\ 0\ 0\ 0\ 0\ 0\ 0\ 58\ 50\ 18\ 0 && (2)\\ 8> &\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -266\ -112\ 0 && (1)\\ 9) &\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -756\ -532 && (1)\\ 10> &\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -980 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 260680 (6 цифр).

При преобразовании 4, выбирая ведущий элемент, изменяем строку 4. Запоминаем строку

$$4 > \ 0\ 0\ 0\ 0\ 3\ \ -2\ \ -1\ 0\ 0\ 0 \qquad (2)$$

Пример 8. В этом примере неполной PRS нуль появляется в качестве свободного члена полинома $p_3(x)$.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$\begin{aligned} p(1) &= 2\ 3\ 5\ \ -2\ \ -1 && (4)\\ p(2) &= 1\ 1\ 2\ \ -1 && (3)\\ p(3) &= -2\ 0 && (1)\\ p(4) &= 8 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 16 (2 цифры).

b. Алгоритм Габихта субрезультантных PRS

$$\begin{aligned} p(1) &= 2\ 3\ 5\ \ -2\ \ -1 && (4)\\ p(2) &= 1\ 1\ 2\ \ -1 && (3)\\ p(3) &= -2\ 0 && (1)\\ p(4) &= 8 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 16 (2 цифры).

Метод матричной триангуляризации субрезультантных PRS

$$\begin{aligned} 1 &> 2\ 3\ 5\ \ -2\ \ -1\ 0\ 0\ 0 && (4)\\ 2 &> 0\ 1\ 1\ 2\ \ -1\ 0\ 0\ 0 && (3)\\ 3) &\ 0\ 0\ 1\ 1\ 0\ \ -1\ 0\ 0 && (3)\\ \#4) &\ 0\ 0\ 0\ 1\ 5\ 0\ \ -1\ 0 && (3)\\ 5 &> 0\ 0\ 0\ 0\ 2\ 0\ 0\ 0 && (1)\\ 6) &\ 0\ 0\ 0\ 0\ 0\ 4\ 0\ 0 && (1)\\ \#7) &\ 0\ 0\ 0\ 0\ 0\ 0\ 8\ \ -4 && (1)\\ 8 &> 0\ 0\ 0\ 0\ 0\ 0\ 0\ \ -8 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 32 (2 цифры).

При преобразовании 4, выбирая ведущий элемент, изменяем строку 4. Запоминаем строку

$$4 > \ 0\ 0\ 0\ 0\ 2\ 0\ 0\ 0 \qquad (1)$$

При преобразовании 7, выбирая ведущий элемент, изменяем строку 7. Запоминаем строку

$$7 > \ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -4 \qquad (0)$$

Пример 9. Здесь полиномы не являются взаимно простыми.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$\begin{aligned}
p(1) &= 6\ \ 7\ \ -3\ \ 16\ \ 20\ \ 8\ \ 17\ \ 6\ \ 18 && (8)\\
p(2) &= 2\ \ 1\ \ 3\ \ 1\ \ -2\ \ 9\ \ -7\ \ 3 && (7)\\
p(3) &= -56\ \ 28\ \ 96\ \ -60\ \ 80\ \ 44\ \ 48 && (6)\\
p(4) &= 5824\ \ 1792\ \ -1008\ \ 10528\ \ -2912\ \ 3696 && (5)\\
p(5) &= 782976\ \ 530624\ \ -321920\ \ 1104896\ \ 208704 && (4)\\
p(6) &= 51455488\ \ 25727744\ \ -25727744\ \ 77183232 && (3)\\
p(7) &= 0
\end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 2925396948683036033024 (22 цифры).

b. Алгоритм Габихта субрезультантных PRS

$$\begin{aligned}
p(1) &= 6\ \ 7\ \ -3\ \ 16\ \ 20\ \ 8\ \ 17\ \ 6\ \ 18 && (8)\\
p(2) &= 2\ \ 1\ \ 3\ \ 1\ \ -2\ \ 9\ \ -7\ \ 3 && (7)\\
p(3) &= -56\ \ 28\ \ 96\ \ -60\ \ 80\ \ 44\ \ 48 && (6)\\
p(4) &= 5824\ \ 1792\ \ -1008\ \ 10528\ \ -2912\ \ 3696 && (5)\\
p(5) &= 782976\ \ 530624\ \ -321920\ \ 1104896\ \ 208704 && (4)\\
p(6) &= 51455488\ \ 25727744\ \ -25727744\ \ 77183232 && (3)\\
p(7) &= 0
\end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 29253969486830360333024 (22 цифры).

Метод матричной триангуляризации субрезультантных PRS

```
 1 >  6  7  − 3  16  20  8  17  6  18  0  0  0  0  0  0  0  0  (8)
 2 >  0  2  1  3  1  − 2  9  − 7  3  0  0  0  0  0  0  0  0  (7)
 3)  0  0  8  − 24  26  52  − 38  76  − 6  36  0  0  0  0  0  0  (7)
 4 >  0  0  0  56  − 28  − 96  60  − 80  − 44  − 48  0  0  0  0  0  0  (6)
 5)  0  0  0  0  − 560  1112  1216  − 744  2304  24  1008  0  0  0  0  0  (6)
 6 >  0  0  0  0  0  − 5824  − 1792  1008  − 10528  2912  − 3696  0  0  0  0  0  (5)
 7)  0  0  0  0  0  0  −133568  −116384  −27904  −210496  −39456  −104832  0  0  0  0  (5)
 8 >  0  0  0  0  0  0  0  782976  530624  − 321920  1104896  208704  0  0  0  0  (4)
 9)  0  0  0  0  0  0  0  0  3477248  11134336  2959232  518016  14093568  0  0  0  (4)
10 >  0  0  0  0  0  0  0  0  0  51455488  25727744  − 25727744  77183232  0  0  0  (3)
11)  0  0  0  0  0  0  0  0  0  0  617465856  308732928  − 308732928  926198784  0  0  (3)
#12 >  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  (0)
13)  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  (0)
14)  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  (0)
15)  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  (0)
16)  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  (0)
```

Наибольшее целое число, порождаемое алгоритмом, равно 571896124988719104 (18 цифр).

Пример 10. Пример поменьше, где полиномы снова не взаимно просты.

Метод Сильвестра–Габихта псевдоделения субрезультантных PRS

a. Алгоритм Сильвестра редуцированных (субрезультантных) PRS

$$
\begin{aligned}
p(1) &= 2\ -1\ -14\ 17\ -5 && (4)\\
p(2) &= 6\ -7\ 4\ -1 && (3)\\
p(3) &= -496\ 592\ -172 && (2)\\
p(4) &= 14432\ -7216 && (1)\\
p(5) &= 0
\end{aligned}
$$

Наибольшее целое число, порождаемое алгоритмом, равно 123303313408 (12 цифр).

b. Алгоритм Габихта субрезультантных PRS

$$\begin{aligned} p(1) &= 2\ -1\ -14\ 17\ -5 \quad (4)\\ p(2) &= 6\ -7\ 4\ -1 \quad (3)\\ p(3) &= -496\ 592\ -172 \quad (2)\\ p(4) &= 14432\ -7216 \quad (1)\\ p(5) &= 0 \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 123303313408 (12 цифр).

Метод матричной триангуляризации субрезультантных PRS

$$\begin{aligned} &1 > 2\ -1\ -14\ 17\ -5\ 0\ 0\ 0 && (4)\\ &2 > 0\ 6\ -7\ 4\ -1\ 0\ 0\ 0 && (3)\\ &\quad 3)\ 0\ 0\ 8\ -92\ 104\ -30\ 0\ 0 && (3)\\ &4 > 0\ 0\ 0\ 496\ -592\ 172\ 0\ 0 && (2)\\ &\quad 5)\ 0\ 0\ 0\ 0\ -6816\ 8368\ -2480\ 0 && (2)\\ &6 > 0\ 0\ 0\ 0\ 0\ -14432\ 7216\ 0 && (1)\\ &\quad 7)\ 0\ 0\ 0\ 0\ 0\ 0\ -144320\ 72160 && (1)\\ &8 > 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 && (0) \end{aligned}$$

Наибольшее целое число, порождаемое алгоритмом, равно 983685120 (9 цифр).

Мы дадим сейчас табл. 5.4.1, суммирующую информацию о наибольших целых числах, участвующих в операциях деления, в каждом из рассмотренных примеров при обоих методах. Заметим, что метод матричной триангуляризации порождает наименьшие «наибольшие целые».

Упражнения

Раздел 5.1.1

Таблица 5.4.1

Число цифр в наибольшем из участвующих в делении чисел

Пример	Метод PRS Сильвестра–Габихта		Метод матричной триангуляризации PRS
	Алгоритм PRS Сильвестра	Алгоритм PRS Габихта	
1	18	11	10
2	5	5	7
3	16	16	12
4	11	11	10
5	28	22	17
6	29	29	22
7	6	6	6
8	2	2	2
9	22	22	18
10	12	12	9

1. Вычислите псевдочастное $q(x)$ и псевдоостаток $r(x)$, если $p_1(x) = x^3 - 3x + 1$ и $p_2(x) = 3x^2 - 3$. Чему равен их наибольший общий делитель? Сделайте то же самое для полиномов $p_1(x) = x^3 - 6x^2 + 8x + 40$ и $p_2(x) = 3x^2 - 12x + 8$.

2. Пусть $r(x)$ — псевдоостаток от псевдоделения полинома $p_1(x)$ на $p_2(x)$. Докажите, что если $\deg[p_1(x)] \geq \deg[p_2(x)] + 2$ и $\deg[p_2(x)] \geq \deg[r(x)]+2$, то $r(x)$ — кратное полинома $\mathrm{lc}[p_2(x)]$.

Раздел 5.1.2

1. Опишите своими словами различие между **M** и **M1**.

2. Используя один из модулярных алгоритмов вычисления gcd, описанных в тексте, вычислите наибольший общий делитель полиномов:

a. $p_1(x) = 6x^4 - 24x^3 + 42x^2 - 32x + 11$ и $p_2(x) = 24x^3 - 72x^2 + 84x - 32$.

b. $p_1(x) = 16x^4 - 32x^3 + 88x^2 - 8x + 17$ и $p_2(x) = 64x^3 - 96x^2 + 176x - 8$.

c. $p_1(x) = x^4 + 2x^3 - 4x + 10$ и $p_2(x) = 4x^3 + 6x^2 - 4$.

Раздел 5.2.1

1. Пользуясь теоремой 5.2.1 и следующими примерами этого раздела, вычислите последовательность полиномиальных остатков для полиномов:
   a. $p_1(x) = 2x^3 - 3x^2 + 4x - 1$ и $p_2(x) = 6x^2 - 6x + 4$.
   b. $p_1(x) = x^4 - x^3 + 3x^2 - 2x + 1$ и $p_2(x) = 4x^3 - 3x^2 + 6x - 2$.
   c. $p_1(x) = x^5 + x^4 - 2x^3 - 2x^2 + 4$ и $p_2(x) = 5x^4 + 4x^3 - 6x^2 - 4x$.
   d. $p_1(x) = x^7 - x^5 - 8x^2 + 3$ и $p_2(x) = 7x^6 - 5x^4 - 16x$.

Раздел 5.2.2

Упр. 1 *чрезвычайно* важно для досконального понимания следующего раздела; однако оно слишком длинное и трудоемкое (для проверки арифметических вычислений пользуйтесь системой компьютерной алгебры).

1. Используя $\mathbf{M}_{\mathrm{B}}$, матрицу, соответствующую результанту $\mathrm{res}_{\mathrm{B}}[p_1(x), p_2(x)]$, где $p_1(x) = 3x^9 + 5x^8 + 7x^7 - 3x^6 - 5x^5 - 7x^4 + 3x^3 + 5x^2 + 7x - 2$ и $p_2(x) = x^8 - x^5 - x^2 - x - 1$, и метод, кратко описанный в тексте, вычислите члены (неполной) последовательности полиномиальных остатков. Какой вывод вы можете сделать? (*Указание.* Следуйте примеру, предшествующему теореме 5.2.4.) Сравните PRS, полученную таким образом, с PRS, полученной методом Сильвестра.
2. Из теоремы 5.2.4 видно, что результант может быть вычислен с помощью того же самого процесса, который используется для нахождения наибольшего общего делителя двух полиномов. Найдите результант полиномов $p_1(x) = x^4 - x^3 + 2x^2 - 3x + 1$ и $p_2(x) = x^3 - 2x^2 + 3x - 1$.
3. Докажите теорему 5.2.9.
4. Вычислите дискриминант полинома $p(x) = x^2 + ax + b$.

Дополнительные формы результанта (известные также как **элиминанты***)* (см. с. 31–42 в книге Salmon G. *Modern Higher Algebra.* Hodges, Smith and Co., Dublin, 1859).

5. Докажите *основное предложение теории исключения* (см. также приложение): Дана система n однородных линейных уравнений от n неизвестных, $\mathbf{A}\mathbf{X}^T = \mathbf{0}^T$. Эта система имеет нетривиальное решение (т.е. отличное от $x_1 = \cdots = x_n = 0$) тогда и только тогда, когда определитель матрицы коэффициентов $\mathbf{A}$ обращается в нуль (т.е. $|\mathbf{A}| = 0$).
6. (Результант в форме Эйлера.)
   a. Пусть $p_1(x) = 0$ и $p_2(x) = 0$ — два полиномиальных уравнения степеней m и n соответственно. Если полино-

мы $p_1(x)$ и p_2 имеют общий сомножитель, произведение $p_1(x)$ на оставшиеся сомножители полинома $p_2(x)$ равно произведению $p_2(x)$ на оставшиеся сомножители полинома $p_1(x)$. Используйте этот факт и упр. 5, чтобы получить детерминантную форму результанта полиномов $p_1(x)$ и $p_2(x)$. (*Указание.* См. обсуждение, следующее за теоремой 5.2.4.)

b. Пусть $p_1(x) = ax^2 + bx + c$ и $p_2(x) = a'x^2 + b'x + c'$ — два полиномиальных уравнения, оба степени 2. Используя (a), вычислите детерминантную форму результанта.

7. (Диалитическое разложение Сильвестра; результаты совпадают с результатами Эйлера. *Историческое замечание: в тексте получающийся определитель мы называем по имени итальяно-французского математика Ди Брюно.*) Основная идея здесь — рассматривать все индивидуальные мономы (термы) полинома как *независимые* переменные (например, x^2 и x берутся как независимые переменные). Порождается некоторое число дополнительных уравнений простым умножением исходных уравнений на хорошо подобранные мономы, так что общее число уравнений равняется общему числу мономов, а затем применяется упр. 5.

a. Пусть $p_1(x) = 0$ и $p_2(x) = 0$ — два полиномиальных уравнения степеней m и n соответственно. Домножим $p_1(x)$ (степени m) на x^{n-1}, x^{n-2}, ..., а $p_2(x)$ (степени n) на x^{m-1}, x^{m-2}, ...; таким образом, мы получим $m + n$ уравнений от $m + n$ неизвестных (x^{m+n-1}, x^{m+n-2},...) или, эквивалентно, уравнение $\mathbf{A}\mathbf{x}^T = \mathbf{0}^T$, где $\mathbf{x} = (x^{m+n-1}, x^{m+n-2}, \ldots)$, а $\mathbf{A}$ — матрица с $m + n$ строками и столбцами. Примените упр. 5, чтобы получить детерминантную форму результанта полиномов $p_1(x)$ и $p_2(x)$.

b. Пусть $p_1(x) = ax^2 + bx + c$ и $p_2(x) = a'x^2 + b'x + c'$ — два полиномиальных уравнения, оба степени 2. Используя п. a, вычислите детерминантную форму результанта; чем она отличается от формы 6b?

8. (Результант в форме Безу.) Здесь мы применяем те же основные идеи, что и в упр. 7, т.е. из исходных двух полиномов мы формируем множество s уравнений от s неизвестных. Определитель теперь имеет меньше элементов, чем раньше, *но* каждый элемент получается более сложным способом. Этот подход лучше объяснить на примере.

a. Пусть $p_1(x) = ax^3 + bx^2 + cx + d$ и $p_2(x) = a'x^3 + b'x^2 + c'x + d'$ — два полинома степени 3. Наша цель — получить три уравнения от независимых переменных x^2, x и 1. После умножения $p_1(x)$ на a', $p_2(x)$ на a и вычитания первые слагаемые в каждом полиноме взаимно уничтожатся. Если мы воспользуемся обозначением $(i, j') = ij' - i'j$, то первое уравнение —

$$ap_2(x) - a'p_1(x) = (a, b')x^2 + (a, c')x + (a, d') = 0.$$

Затем, после умножения $p_1(x)$ на $(a'x+b)$, $p_2(x)$ на $(ax+b)$ и вычитания первые два слагаемых в каждом полиноме взаимно уничтожатся, и мы получаем уравнение

$$\begin{aligned}(ax + b)p_2(x) - (a'x + b')p_1(x) \\ &= (a, c')x^2 + [(a, d') + (b, c')]x + (b, d') \\ &= 0.\end{aligned}$$

Подобным же образом мы получаем последнее уравнение

$$\begin{aligned}(ax^2 + bx + c)p_2(x) - (a'x^2 + b'x + c')p_1(x) \\ &= (a, d')x^2 + (b, d')x + (c, d') \\ &= 0.\end{aligned}$$

Ясно, что мы получили уравнение $\mathbf{A}\mathbf{x}^T = \mathbf{0}^T$, где $\mathbf{x} = (x^2, x, 1)$, а $\mathbf{A}$ — матрица с 3 строками и столбцами. Примените теперь упр. 5, чтобы получить детерминантную форму результанта полиномов $p_1(x)$ и $p_2(x)$.

b. Пусть $p_1(x) = ax^2 + bx + c$ и $p_2(x) = a'x^2 + b'x + c'$ — два полиномиальных уравнения, оба степени 2. Используя п. a, вычислите детерминантную форму результанта; что случится, если степени не совпадают?

9. (Вариант Кэли метода Безу.)

Все необходимые дополнительные полиномы порождаются теперь за один проход, без необходимости находить дополнительные множители. Как и ожидается, снова применяется упр. 5.

a. Пусть $p_1(x) = 0$ и $p_2(x) = 0$ — два полиномиальных уравнения степеней m и n соответственно. Заметим, что

если $p_1(x)$ и $p_2(x)$ имеют общий корень $x = x_0$, то он будет корнем уравнения

$$p_1(x)p_2(\alpha) - p_1(\alpha)p_2(x) = 0$$

для любого значения α. Это уравнение удовлетворяется при $x = \alpha$ (даже если нет общего корня), поэтому выписанное выражение должно содержать $(x - \alpha)$ в качестве сомножителя. Разделив это выражение на $(x - \alpha)$, мы получим полином с мономами от 1, α, α^2, ..., где коэффициент при каждом мономе является полиномом от x. В общем корне $x = x_0$ полученное выражение должно обращаться в нуль при любом значении α, *поэтому коэффициент-полином от x должен обращаться в нуль при $x = x_0$*. Имеется n коэффициентов от n переменных, и это — в точности полиномы, необходимые для построения результанта в форме Безу, разумеется, используя упр. 5.

Пример. Пусть $p_1(x) = ax^2 + bx + c$ и $p_2(x) = a'x^2 + b'x + c'$ — два полиномиальных уравнения, оба степени 2. Снова используя обозначение $(i, j') = ij' - i'j$, мы получаем

$$[p_1(x)p_2(\alpha) - p_1(\alpha)p_2(x)]/(x - \alpha) = [(a, b')x + (a, c')]\alpha + [(a, c')x + (b, c')].$$

Для того чтобы существовал общий корень, все коэффициенты полинома от α должны быть равны нулю, т.е. мы имеем уравнение $\mathbf{A}\mathbf{x}^T = \mathbf{0}^T$, где $\mathbf{x} = (x, 1)$, а элементы матрицы $\mathbf{A}$ суть (a, b'), (a, c') (первая строка) и (a, c'), (b, c') (вторая строка), и, применяя упр. 5, мы получаем результант.

b. Пользуясь подходом Кэли, вычислите детерминантную форму результанта полиномов $p_1(x) = ax^3 + bx^2 + cx + d$ и $p_2(x) = a'x^3 + b'x^2 + c'x + d'$.

10. Докажите, что если каждый элемент в какой-либо (или в каком-либо столбце) представить в виде суммы двух других, то определитель разлагается в сумму двух других; т.е.

$$\det \begin{bmatrix} a_1 & a_2 & a_3 + a_4 \\ b_1 & b_2 & b_3 + b_4 \\ c_1 & c_2 & c_3 + c_4 \end{bmatrix} = \det \begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{bmatrix} + \det \begin{bmatrix} a_1 & a_2 & a_4 \\ b_1 & b_2 & b_4 \\ c_1 & c_2 & c_4 \end{bmatrix}.$$

11. В этой задаче мы увидим, как из n уравнений с m членами получить уравнение с $m-n+1$ членами. Рассмотрим четыре уравнения степени 4

$$\begin{aligned} p_1(x) &= a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 = 0, \\ p_2(x) &= b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = 0, \\ p_3(x) &= c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0 = 0, \\ p_1(x) &= d_4x^4 + d_3x^3 + d_2x^2 + d_1x + d_0 = 0, \end{aligned}$$

где x^4, x^3, x^2, x и 1 — неизвестные. Из этих четырех уравнений можно ислючить любые три неизвестные, оставив одно уравнение от оставшихся двух неизвестных. Предположим, что мы хотим получить из $p_1(x)$, $p_2(x)$, $p_3(x)$, $p_4(x)$ единственное уравнение, которое выполняется, когда выполняются исходные четыре уравнения, но которое не содержит мономов x^4, x^3, x^2. Пользуясь упр. 10, докажите, что требуемое уравнение получается из

$$\det \begin{bmatrix} a_4 & a_3 & a_2 & p_1(x) \\ b_4 & b_3 & b_2 & p_2(x) \\ c_4 & c_3 & c_2 & p_3(x) \\ d_4 & d_3 & d_2 & p_4(x) \end{bmatrix} = 0.$$

Раздел 5.2.3

1. Докажите формулы (H2) этого раздела.
2. Примените расширенный алгоритм Евклида для полиномов над целыми числами к полиномам x^3-7x+7 и $3x^2-7$ и сравните ваш ответ с ответом, полученным в примере, следующем за теоремой 5.2.11. Что вы можете сказать о последнем члене PRS?
3. Завершите вычисления в последнем примере этого раздела.
4. Используйте формулы в **H-1**, **H-2** и **H-3**, чтобы вывести выражение (H) из разд. 5.1.1.

Раздел 5.3.1

1. Определите наибольший общий делитель полиномов x^3-7x+7 и $3x^2-7$, пользуясь процессом гауссова исключения вместо псевдоделения.
2. Сделайте то же самое для полиномов $x^4+6x^3+5x^2-4x-2$ и $4x^3+18x^2+10x-4$.

Раздел 5.3.3

1. Используя метод, кратко описанный в тексте, завершите первый пример этого раздела, т.е. определите, чему равен полином $p_4(x)$?
2. Следуя первому примеру этого раздела, вычислите последовательность полиномиальных остатков для следующих пар полиномов:
   a. $p_1(x) = 2x^4+5x^3+5x^2-2x+1$ и $p_2(x) = 3x^3+3x^2+3x-4$.
   b. $p_1(x) = x^5+5x^4+10x^3+5x^2+5x+2$ и $p_2(x) = x^4+4x^3+6x^2+2x+1$.
   c. $p_1(x) = 2x^4+3x^3+5x^2-2x-1$ и $p_2(x) = x^3+x^2+2x-1$.
   d. $p_1(x) = 2x^4-x^3-14x^2+17x-5$ и $p_2(x) = 6x^3-7x^2+4x-1$.

Упражнения по программированию

Раздел 5.1.1

1. Напишите процедуру для вычисления псевдочастного и псевдоостатка от псевдоделения полинома $p_1(x)$ на $p_2(x)$.
2. Напишите процедуру для нахождения матрицы, соответствующей результанту в форме Сильвестра полиномов $p_1(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0$ и $p_2(x) = d_m x^m + d_{m-1}x^{m-1} + \cdots + d_0$, $c_n \neq 0$, $d_m \neq 0$, $n \geq m$.

Раздел 5.2.1

1. Напишите процедуру для реализации алгоритма **SRSPRS** и примените ее к полиномам упр. 1 к этому разделу.

Раздел 5.2.3

1. Напишите процедуру для реализации алгоритма **HSPRS** и примените ее к полиномам:
   a. $p_1(x) = 2x^4+5x^3+5x^2-2x+1$ и $p_2(x) = 3x^3+3x^2+3x-4$.
   b. $p_1(x) = x^5+5x^4+10x^3+5x^2+5x+2$ и $p_2(x) = x^4+4x^3+6x^2+2x+1$.
   c. $p_1(x) = 2x^4+3x^3+5x^2-2x-1$ и $p_2(x) = x^3+x^2+2x-1$.
   d. $p_1(x) = 2x^4-x^3-14x^2+17x-5$ и $p_2(x) = 6x^3-7x^2+4x-1$.

Раздел 5.3.2

1. Напишите процедуру, которая для данных двух полиномов с целыми коэффициентами будет распечатывать соответствующие матрицы $\mathbf{M}_{\mathrm{B}}$ и $\mathbf{M}_{\mathrm{T}}$. Примените ее к вашим любимым полиномам.
2. Напишите процедуру для реализации алгоритма Доджсона (D), обсуждаемого в разд. 5.3.1, и для данных матриц $\mathbf{M}_{\mathrm{B}}$ и $\mathbf{M}_{\mathrm{T}}$ воспользуйтесь ею, чтобы получить верхние треугольные матрицы $T(\mathbf{M}_{\mathrm{B}})$ и $T(\mathbf{M}_{\mathrm{T}})$.

Раздел 5.3.3

1. Напишите процедуру для реализации алгоритма **ASPRS** и примените ее к полиномам упр. 2 к этому разделу. Выпишите соответствующие последовательности полиномиальных остатков и сравните ответы.

Исторические замечания и литература

1. Браун в обеих своих статьях (1971, p. 485, 1978, p. 238) и Кнут (1981, p. 410) приписывают Коллинзу (1967) открытие метода Сильвестра–Габихта субрезультантных PRS. Как мы показали, это открытие было начато Сильвестром в 1853 г. и завершено Габихтом в 1948 г. В этой книге впервые должное было отдано тому, кому оно принадлежит. [См. также в этом контексте (Fryer, 1959).]

2. Статьи Сильвестра (1853) и Ван Влека (1899) не упоминаются в обзорной статье Лооса (1982).

3. Мы отдаем должное Доджсону (Dodgson, 1866), а не Барейсу (Bareiss, 1968) за сохраняющие целочисленность преобразования. См. также работу (Waugh, Dwyer, 1945), в которой используется тот же метод, что и у Барейса, но на 23 года раньше. Авторы последней работы указывают на Доджсона как на своего предшественника; их подход отличается только тем, что они не используют подвижный ведущий элемент (Waugh, Dwyer, 1945, p. 266). Доджсон — это человек, известный главным образом своими литературными произведениями под псевдонимом Льюис Кэрролл.

4. Дальнейшее исследование и развитие метода матричной триангуляризации субрезультантных PRS см. в работе (Akritas, Akritas,

Malaschonok, 1992). Получены теоретические результаты, не зависящие от теоремы Ван Влека (которая не всегда верна), где вместо преобразования матрицы порядка $2\max(m,n)$ преобразуется матрица порядка $m+n$.

Малашонок Г.И. Решение системы линейных уравнений над областью целостности. *Журнал вычислительной математики и математической физики* **23**, 1497–1500, 1983.

Малашонок Г.И. Система линейных уравнений над коммутативным кольцом. — Львов: ФМИ АН УССР, 1986 (препринт № 114).

Akritas A.G. *A new method for computing polynomial greatest common divisors.* University of Kansas, TR-86-9, Lawrence, Kansas, 1986.

Akritas A.G. A simple proof of the validity of the reduced prs algorithm. *Computing* **38**, 369–372, 1987.

Akritas A.G. A new subresultant prs method. Proceedings ot the 12th IMACS World Congress on Scientific Computation (July 1988, Paris, France), **4**, 654–655, 1988.

Akritas A.G. A new method for computing polynomial greatest common divisors and polynomial remainder sequences. *Numerische Mathematik*, **52**, 119–127, 1988.

Akritas A.G., Akritas E.K., Malaschonok G.I. Computation of polynomial remainder sequences in integral domains. Submitted for publication, 1992.

Anderson G. An examination of polynomial remainder sequences. M.S. Thesis, University of Kansas, Department of Computer Science, Lawrence, KS, 1986.

Bareiss E.H. Sylvester's identity and multistep integer-preserving Gaussian elimination. *Mathematics of Computation* **22**, 565–578, 1968.

Barnett S. *Polynomials and Linear Control Systems.* Marcel Dekker Inc., New York & Basel, 1983.

Barnett S. A new look at classical algorithms for polynomial resultant and gcd calculation. *SIAM Review* **16**, 193–206, 1974.

Berlekamp E.R. *Algebraic coding theory.* McGraw-Hill, New York, 1968. [Имеется перевод: Берлекэмп Э. Алгебраическая теория кодирования. — М.: Мир, 1974.]

Bocher M. *Introduction to higher algebra.* MacMillan, New York, 1907.

Brown W.S. On Euclid's algorithm and the computation of polynomial greatest common divisors. *Journal of Association for Computing Machinery* **18**, 476–504, 1971.

Brown W.S. The subresultant prs algorithm. *ACM Transactions on Mathematical Software* **4**, 237–249, 1978.

Brown W.S., Traub J.F. On Euclid's algorithm and the theory of subresultants. *Journal of Association for Computing Machinery* **18**, 505–514, 1971.

Collins G.E. Polynomial remainder sequences and determinants. *American Mathematical Monthly* **73**, 708–712, 1966.

Collins G.E. Subresultants and reduced polynomial remainder sequences. *Journal of Association for Computing Machinery* **14**, 128–142, 1967.

Collins G.E. The calculation of multivariate polynomial resultants. *Journal of Association for Computing Machinery* **19**, 515–532, 1971.

Dodgson C.L. Condensation of determinants. *Proceedings of the Royal Society of London* **15**, 150–155, 1866.

Fryer W.D. Applications of Routh's algorithm to network theory problems. *IEEE Transactions on Circuit Theory* **CT-6**, 144–149, 1959.

Gonzalez L., Lombardi H., Recio T., Roy M.-F. Specialization de la suite de Sturm et sous-resultants. Preprint Num. 8-1990, Department of Mathematics Statistics and Computation, University of Cantabria, 39071 Santander, Spain.

Habicht W. Eine Verallgemeinerung des Sturmschen Wurzelzaehlverfahrens. *Commentarii Mathematici Helvetici* **21**, 99–116, 1948.

Householder A.S. *The numerical treatment of a single nonlinear equation.* McGraw-Hill, New York, 1970.

Knuth D.E. The art of computer programming. Vol. 2, 2nd ed. *Seminumerical Algorithms.* Addison-Wesley, Reading, MA 1981. [Имеется перевод первого издания: Кнут Д. *Искусство программирования для ЭВМ.* Т. 2. — М.: Мир, 1977.]

Laidacker M.A. Another theorem relating Sylvester's matrix and the greatest common divisor. *Mathematics Magazine* **42**, 126–128, 1969.

Lipson J.D. *Elements of algebra and algebraic computing.* Addison-Wesley. Reading, MA, 1981.

Loos R. Generalized polynomial remainder sequences. In *Computer Algebra Symbolic and Algebraic Computations.* B. Buchberger, G.E. Collins, and R. Loos eds. Springer Verlag, New York, 1982, *Computing Supplement* **4**, 115–137. [Имеется перевод: Лоос Р. Обобщенные последовательности полиномиальных остатков. — В кн.: Компьютерная алгебра. Символьные и алгебраические вычисления. — М.: Мир, 1986, с. 151–171.]

Miola A., Yun D.Y.Y. Computational aspects of Hensel-type univariate polynomial greatest common divisor algorithms. *Proceedings of EUROSAM 1974*, pp. 46–54 (also *ACM SIGSAM Bulletin 31*).

Moses J., Yun D.Y.Y. The EZGCD algorithm. *Proceedings of the ACM Annual Conference* (August 1973, Atlanta), pp. 159–166.

Muir T. *A treatise on the theory of determinants.* Macmillan, London, 1882.

Muir T. *The theory of determinants.* Vol III. Macmillan, London, 1920.

Netto E. *Vorlesungen ueber Algebra. Erster Band.* Teubner, Leipzig, 1896.

Sylvester J.J. On a theory of the syzygetic relations of two rational integral functions, comprising an application to the theory of Sturm's functions, and that of the greatest algebraical common measure. *Philosophical Transactions* **143**, 407–548, 1853.

Van Vleck, E. B. On the determination of a series of Sturm's functions by the calculation of a single determinant. *Annals of Mathematics, Second Series* **1**, 1–13, 1899–1900.

Waugh F.V., Dwyer P.S. Compact computation of the inverse of a matrix. *Annals of Mathematical Statistics* **16**, 259–271, 1945.

Yun D.Y.Y. *The Hensel lemma in algebraic manipulation.* Ph.D. Thesis, Department of Mathematics, MIT, 1974.

Глава 6

Разложение на множители полиномов над целыми числами

6.1

Введение и обоснования

Ясно, что самый простой способ нахождения множителей данного полинома $p(x)$, $\deg[p(x)] = k$, — это алгебраически найти его корни ρ_i, и тогда $p(x) = (x - \rho_1)(x - \rho_2)\ldots(x - \rho_k)$. (Читателю следует сейчас снова просмотреть разд. 3.2.3.) Однако, как мы убедимся в гл. 7, мы можем алгебраически найти корни полинома, *только* если его степень ≤ 4. Для полиномов, степень которых больше 4, мы должны применить другой подход.

Помимо чисто теоретического интереса разложение полиномов на множители имеет большое практическое значение (например, в теории кодирования). Ньютон в своей *Arithmetica Universalis* (1707 г.) формулирует метод нахождения линейных и квадратичных множителей полиномов с целыми коэффициентами. В 1793 г. метод Ньютона был обобщен астрономом Фридрихом фон Шубертом, показавшим, как находить все множители степени n за конечное число шагов (Cantor, 1908; p. 136–137). Метод Шуберта примерно через 90 лет был заново открыт Л. Кронекером. Однако, как мы вскоре убедимся, метод Кронекера–Шуберта имеет экспоненциальную вычислительную сложность по времени и, следовательно, мало пригоден для практического применения.

Гораздо лучшие результаты получаются при использовании методов разложения «по модулю p» вместе с техникой *подъема* разложения «по модулю p» до разложения на множители над целыми числами. Этот подход проиллюстрирован на рис. 6.1.1.

Однако, несмотря на то что подход, показанный на рис. 6.1.1, работает почти во всех практических случаях очень хорошо, в худшем случае он также имеет экспоненциальную границу времени вычислений. Совсем недавно А. Ленстра, Х. Ленстра мл. и Л. Ловас (или просто L^3) применили теорию решеток к задаче разложения на множители и получили метод разложения с полиномиальным временем

вычисления (Lenstra A.K., Lenstra H.W., Lovasz L., 1982); изложение L^3-метода выходит, однако, за рамки этой книги; см. (Lenstra, 1981, 1982a; Панкратьев, 1988).

6.1.1. Метод Кронекера–Шуберта разложения на множители над целыми числами

Этот метод работает следующим образом. Пусть $p(x)$ — данный полином с целыми коэффициентами степени n, который мы хотим разложить на множители. Если $p(x)$ в действительности разлагается на множители, то один из его множителей должен иметь степень не выше $\frac{n}{2}$. Пусть $m = \lfloor \frac{n}{2} \rfloor$ — наибольшее целое число $\leq \frac{n}{2}$; тогда мы должны выяснить, имеет ли $p(x)$ множитель $q(x)$ степени $\leq m$. Для $m+1$ различных целых значений $a_0, a_1, \ldots, a_m$ $(0, \pm 1, \pm 2, \ldots)$ вычисляем $p(a_0), p(a_1), \ldots, p(a_m)$. Если $q(x)$ — делитель полинома $p(x)$, то число $q(a_0)$ должно быть делителем числа $p(a_0)$, $q(a_1)$ — числа $p(a_1)$ и т.д. Обозначим через f_i конечное множество всех различных целых делителей числа $p(a_i)$. Тогда для $k = 2, \ldots, m+1$ выполняем следующие операции: выбираем k элементов b_i из различных f_i и вычисляем интерполяционный полином Лагранжа $q(x)$ степени $k-1$, такой, что $q(a_i) = b_i$ для всех i. Если $q(x)$ делит $p(x)$, то мы нашли множитель полинома $p(x)$ и можем рекурсивно применить этот метод к $p(x)/q(x)$. В противном случае выбираем другое множество k элементов b_i из f_i (не совпадающее ни с одним из выбранных ранее множеств), интерполируем и снова проверяем на делимость. Когда мы испытаем все возможные комбинации из $m+1$ или меньшего количества целых значений из f_i, мы придем к заключению, что полином $p(x)$ неприводим.

Рис. 6.1.1.

Окольный способ разложения на множители произвольного полинома $p(x)$ в $\mathbb{Z}[x]$.

Пример. Попробуем разложить на множители полином $p(x) = x^4 + 4$. Заметим, что $n = 4$ и $m = \lfloor \frac{n}{2} \rfloor = 2$. Тогда выберем $m + 1 = 3$

точки $a_0 = 0,\ a_1 = 1,\ a_2 = -1$ и сформируем множества f_i:

$$\begin{aligned} p(a_0) = p(0) = 4, &\qquad f_0 = \{1, -1, 2, -2, 4, -4\}, \\ p(a_1) = p(1) = 5, &\qquad f_1 = \{1, -1, 5, -5\}, \\ p(a_2) = p(-1) = 5, &\qquad f_2 = \{1, -1, 5, -5\}. \end{aligned}$$

Легко видеть, что имеется громадное количество возможных комбинаций; мы пропускаем случай $k = 2$, так как он не даст нам линейных сомножителей полинома $p(x)$, и попробуем $k = 3$. Выберем сначала $b_0 = 1,\ b_1 = -1,\ b_2 = 5$ и, интерполируя пары

$$\{a_0 = 0, b_0 = -1\},\ \{a_1 = 1, b_1 = -1\},\ \{a_2 = -1, b_2 = 5\},$$

получаем

$$\begin{aligned} q(x) = &-1 \left\{ \frac{(x-1)(x+1)}{(0-1)(0+1)} \right\} + (-1) \left\{ \frac{(x-0)(x+1)}{(1-0)(1+1)} \right\} \\ &+ 5 \left\{ \frac{(x-0)(x-1)}{(-1-0)(-1-1)} \right\} \\ = &x^2 - 3x + 1. \end{aligned}$$

Затем делим $p(x)$ на $q(x)$ и видим, что $q(x)$ не является сомножителем $p(x)$, так как

$$x^4 + 4 = (x^2 - 3x + 1)(x^2 + 3x + 8) + (21x - 4).$$

Тогда выбираем $b_0 = -2,\ b_1 = 5,\ b_2 = 1$ и пары для интерполяции

$$\{a_0 = 0, b_0 = -2\},\ \{a_1 = 1, b_1 = 5\},\ \{a_2 = -1, b_2 = 1\};$$

получаем

$$\begin{aligned} q(x) = &-2 \left\{ \frac{(x-1)(x+1)}{(0-1)(0+1)} \right\} + 5 \left\{ \frac{(x-0)(x+1)}{(1-0)(1+1)} \right\} \\ &+ 1 \left\{ \frac{(x-0)(x-1)}{(-1-0)(-1-1)} \right\} \\ = &x^2 + 2x + 2. \end{aligned}$$

Видим, что на этот раз $q(x)$ является сомножителем полинома $p(x)$, так как

$$x^4 + 4 = (x^2 + 2x + 2)(x^2 - 2x + 2),$$

и это — полное разложение на множители полинома $p(x)$, поскольку по критерию Эйзенштейна (теорема 3.2.15) эти два сомножителя неприводимы.

Очевидно, что время вычислений рассмотренного метода экспоненциально, и для $n \geq 5$ метод работает очень неэффективно.

6.1.2. Общая схема «окольного» метода разложения над кольцом целых чисел

В этом разделе мы дадим краткое описание различных шагов, включенных в «окольный» метод, показанный на рис. 6.1.1; детальное описание некоторых шагов приводится в следующих разделах; см. также (Lazard, 1982; Moenck, 1977) (для случая многих переменных см. (Wang, 1978)).

Предположим, что дан полином $p(x) \in \mathbb{Z}[x]$, т.е.

$$p(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_0, \quad c_i \in \mathbb{Z},$$

который следует разложить на множители. Мы хотим найти неприводимые сомножители этого полинома. Прежде всего заметим, что без потери общности можно считать, что $c_n = 1$, поскольку в противном случае можно положить $x := x/c_n$ и умножить получающийся полином на c_n^{n-1}. Таким образом мы получаем нормированный полином $p_{\text{н}}(x) = c_n^{n-1} p(x/c_n)$ с целыми коэффициентами, задача разложения которого эквивалентна задаче разложения на множители полинома $p(x)$, т.е. если $p_{\text{н}}(x) = p_1(x) \ldots p_r(x)$, то разложение полинома $p(x)$ имеет вид $p(x) = (1/c_n^{n-1}) p_{\text{н}}(c_n x) = (1/c_n^{n-1}) p_1(c_n x) \ldots p_r(c_n x)$. Рациональное число $1/c_n^{n-1}$ исчезнет после того, как мы разделим на c_n как можно больше сомножителей и/или выделим содержание сомножителей. В качестве первого примера рассмотрим полином $p(x) = 2x^2 + 3x + 1$. Тогда

$$p_{\text{н}}(x) = 2p\left(\frac{x}{2}\right) = 2\left[2\left(\frac{x}{2}\right)^2 + 3\left(\frac{x}{2}\right) + 1\right] = x^2 + 3x + 2 = (x+1)(x+2).$$

Разложение на множители полинома $p(x)$ получается следующим образом:

$$p(x) = \left(\frac{1}{2}\right) p_{\text{н}}(2x) = \left(\frac{1}{2}\right)(2x+1)(2x+2) = (2x+1)(x+1),$$

где рациональное число $1/2$ в этом случае домножается на 2 (содержание второго сомножителя).

В качестве второго примера того же процесса рассмотрим полином $p(x) = 112x^4 + 58x^3 - 31x^2 + 107x - 66$. Тогда

$$\begin{aligned} p_{\text{н}}(x) &= 112^3 p\left(\frac{x}{112}\right) = x^4 + 58x^3 - 3472x^2 + 1342208x - 92725248 \\ &= (x^2 + 98x - 9408)(x^2 - 40x + 9856). \end{aligned}$$

Разложение на множители полинома $p(x)$ получается следующим образом:

$$\begin{aligned} p(x) &= \left(\frac{1}{112^3}\right) p_{\text{н}}(112x) \\ &= \left(\frac{1}{112^3}\right) [(112x)^2 + 98 \cdot 112x - 9408][(112x)^2 - 40 \cdot 112x + 9856) \\ &= (8x^2 + 7x - 6)(14x^2 - 5x + 11), \end{aligned}$$

где в этом случае рациональное число $1/112^3$ умножается на 112^3, что соответствует делению на 112 обоих сомножителей u выделению в обоих сомножителях содержания.

Таким образом мы предполагаем, что полином $p(x)$ нормирован. (Это ограничение снимается в разд. 6.3.) Ниже излагаются основные шаги «окольного» метода разложения на множители над целыми числами (Musser, 1971, 1975 и 1978):

1. Прежде всего исключаем из $p(x)$ нетривиальные сомножители нулевой степени и кратные сомножители, что достигается вычислением наибольших общих делителей в $\mathbb{Z}$ и $\mathbb{Z}[x]$. (Другими словами, мы вычисляем примитивный свободный от квадратов полином; читателю следует по этому поводу освежить в памяти разд. 3.2.4.) В качестве результата мы получаем нормированный свободный от квадратов полином $p_{\text{sf}}(x) = \prod_{1 \le i \le q} p_i(x)$, где все $p_i(x)$ — различные неприводимые полиномы. Наша цель состоит в вычислении этих q сомножителей $p_i(x)$.

2. Выбираем простое число p, такое, что разложение на множители в $\mathbb{Z}_p[x]$ возможно. Первое требование к выбору числа p заключается в том, чтобы полином $p_{\text{sf}}(x)$, полученный на шаге 1, имел *ту же самую* степень и *оставался* свободным от квадратов по модулю p. Поскольку полином $p_{\text{sf}}(x)$ нормирован, p не делит его старший коэффициент, и, следовательно, степень остается той же самой по модулю p; см. также теорему 3.2.16. Кроме того, мы можем эффективно проверить, является ли полином $p_{\text{sf}}(x) \pmod p$ свободным от квадратов, проверив равенство $\gcd[p_{\text{sf}}(x), p'_{\text{sf}}(x)] = 1$ в $\mathbb{Z}_p[x]$ ($p'_{\text{sf}}(x)$ обозначает производную полинома $p_{\text{sf}}(x)$). Справедливость этого утверждения следует из того, что если $p(x) = [p_1(x)]^2 p_2(x)$, то $p'(x)$ является кратным полинома $p_1(x)$. Поэтому если $\gcd[p(x), p'(x)] = 1$, то мы знаем, что $p(x)$ свободен от квадратов, в то время как если $\gcd[p(x), p'(x)] \neq 1$ и $\gcd[p(x), p'(x)] \neq p(x)$, то нам нужно разложить на множители $\gcd[p(x), p'(x)]$ и $p(x)/\gcd[p(x), p'(x)]$; наконец, если $\gcd[p(x), p'(x)] = p(x)$, то $p(x) = q(x^p) = [q(x)]^p$, и нам нужно

разложить на множители $q(x)$. Дальнейшие детали см. в разд. 6.2.1. Размер используемых простых чисел также является существенным фактором при выборе алгоритма разложения в $\mathbb{Z}_p[x]$. Применение больших простых чисел сократило бы количество шагов гензелева подъема (описанных в разд. 6.3), необходимых для подъема разложения $p_{\rm sf}(x)$ из $\mathbb{Z}_p[x]$ до $\mathbb{Z}[x]$. Однако эффективные алгоритмы разложения на множители в $\mathbb{Z}_p[x]$ имеются только для малых p (Berlekamp, 1970). Третье главное требование к выбору числа p — это число r сомножителей в полном разложении по модулю p. Если $p_{\rm sf}(x)$ — неприводимый полином, но по модулю p расщепляется на $r > 1$ неприводимых сомножителей (для объяснения этого явления см. шаг 3), то в конце подъема мы получаем r сомножителей и в итоге 2^{r-1} подмножеств сомножителей должно рассматриваться на шаге 5, описанном ниже, для определения истинных сомножителей полинома $p_{\rm sf}(x)$ над целыми числами. (Именно из-за этого максимальное время работы алгоритма оказывается экспоненциальным.) Некоторое упрощение задачи получается, если разложить на множители $p_{\rm sf}(x)$ (шаг 3) по модулю нескольких простых чисел p_i, для которых полином остается свободным от квадратов, и выбрать из них в качестве p число, дающее наименьшее число неприводимых сомножителей.

3. Для простого числа p, выбранного выше, мы получили разложение на множители в $\mathbb{Z}_p[x]$

$$p_{\rm sf} f(x) \equiv \prod_{1 \le k \le r} g_k \pmod{p}, \quad g_k \in \mathbb{Z}_p[x].$$

В разд. 6.2 мы рассматриваем способы получения такого разложения. Заметим, что число q истинных сомножителей полинома $p_{\rm sf}(x)$ над кольцом целых чисел (шаг 1) *не обязательно равно* числу r сомножителей полинома $p_{\rm sf}(x)$ по модулю p; например, $x^2 + 2 \equiv (x-2)(x-1) \pmod 3$. Интересно. что существуют полиномы, неприводимые над целыми числами, которые могут быть разложены на множители по модулю p для любого простого p. Например, для любых целых чисел a, b и любого простого числа p полином $p(x) = x^4 + ax^2 + b^2$ разлагается на множители в $\mathbb{Z}_p[x]$. Чтобы убедиться, что полином $p(x) = x^4 + ax^2 + b^2$ разлагается на множители в $\mathbb{Z}_p[x]$ для любого простого p, рассмотрим прежде всего случай $p = 2$. Тогда для любых целых чисел a, b имеются следующие четыре возможности: $p(x) \equiv x^4 \pmod 2$, или $p(x) \equiv x^4 + x^2 = x^2(x^2+1) \pmod 2$, или $p(x) \equiv x^4 + 1 \equiv (x+1)^4 \pmod 2$, или $p(x) \equiv x^4 + x^2 + 1 = (x^2 + x + 1)^2 \pmod 2$. Очевидно, что каждый из них приводим по модулю 2. В общем случае, когда p — нечетное простое число, выберем c, такое, что $a \equiv 2c$

$(\mathrm{mod}\ p)$. Тогда $p(x) \equiv x^4 + 2cx^2 + b^2 \pmod p$, и мы имеем три следующие возможности:

$$p(x) \equiv (x^2 + c)^2 - (c^2 - b^2) \pmod p,$$
$$p(x) \equiv (x^2 + b)^2 - (2b - 2c)x^2 \pmod p,$$
$$p(x) \equiv (x^2 - b)^2 - (-2b - 2c)x^2 \pmod p.$$

Ясно, что для доказательства разложимости $p(x)$ по модулю p достаточно показать, что хотя бы одно из чисел (c^2-b^2), $(2b-2c)$, $(-2b-2c)$ является квадратом $\bmod p$ [$p(x)$ будет тогда разностью двух квадратов]. Пусть d — примитивный корень по модулю p (см. теорему 2.3.20). Если $2b - 2c$ и $-2b - 2c$ не являются квадратами $\bmod p$, то $2b - 2c \equiv d^e$ и $-2b - 2c \equiv d^f \pmod p$, где e и f оба нечетны. Взяв произведение $(2b-2c)(-2b-2c) = 4(b^2-c^2) \equiv d^{e+f}$, где теперь $e+f = 2g$ — четное число, видим, что $4(c^2 - b^2)$ — квадрат $\bmod p$. Пусть $2h \equiv 1 \pmod p$; тогда $c^2-b^2 \equiv 4(c^2-b^2)h^2 \equiv d^{2g}h^2 = (d^g h)^2$, и мы видим, что c^2-b^2 — квадрат $\bmod p$. Таким образом, $p(x) = x^4+ax^2+b^2$ разлагается на множители в $\mathbb{Z}_p[x]$ по модулю любого простого p. Пользуясь критерием Эйзенштейна (теорема 3.2.15), мы легко можем построить полином вида $p(x) = x^4 + ax^2 + b^2$, неприводимый над целыми числами, который разлагается на множители $\bmod p$ для каждого простого p. В качестве примера рассмотрим $x^4 + 1$. В качестве упражнения читателю остается доказательство того, что при $0 < b < a$ полином $p(x) = x^4 + 2ax^2 + b^2$ неприводим над целыми числами.

4. Используя построения гензелева типа (рассматриваемые в разд. 6.3), мы поднимаем полиномы $g_k(x)$, полученные на шаге 3, до соответствующих полиномов $h_k(x) \pmod{p^j}$, таких, что

$$p_{\mathrm{sf}}(x) \equiv \prod_{1 \le k \le r} h_k(x) \pmod{p^j}$$

для достаточно большого положительного целого числа j (это будет разобрано в разд. 6.3). Теперь каждый истинный сомножитель $p_i(x)$ полинома $p_{\mathrm{sf}}(x)$ соответствует или отдельному полиному $h_i(x)$, или произведению $\pmod{p^j}$ некоторых из них. Это соответствие выявляется на шаге 5.

5. Здесь мы разбиваем множество полиномов $h_k(x)$ на подмножества h_i, $i = 1, 2, \ldots, f$, такие, что

$$p_i(x) \equiv \prod_{h(x) \in h_i} h(x) \pmod p^j.$$

Истинные сомножители полинома $p_{\rm sf}(x)$ над целыми числами определяются тогда пробным делением. [То есть получив на предыдущем шаге сомножители $h_1(x), h_2(x), \ldots, h_r(x)$ полинома $p_{\rm sf}(x) \mod p^j$, мы должны рассмотреть каждое сочетание этих сомножителей, проверяя делением, является ли их произведение по $\mod p^j$ истинным сомножителем; если найден истинный сомножитель, то полагаем $p_{\rm sf}(x) := p_{\rm sf}(x)/p_i(x)$ и удаляем соответствующие значения $h_i(x)$ из списка. Необходимо рассмотреть только те сочетания, для которых степень $\le \lfloor \deg[p(x)]/2 \rfloor$.]

Анализ времени работы ***«окольного» метода разложения***. Выше мы уже видели, что в худшем случае окольный метод разложения имеет экспоненциальное время работы, поскольку r может быть так же велико, как и n, и потребуется выполнить большое число пробных делений (а именно 2^n). Однако, как мы сейчас убедимся, в среднем $r = \ln(n)$, и поскольку среднее значение величины 2^r приблизительно равно n, *среднее время вычислений* окольного алгоритма полиномиально. (Среднее значение выражения 2^r получается с помощью производящих функций и здесь не обсуждается.)

Чтобы убедиться, что $r = \ln(n)$, рассмотрим эквивалентную задачу определения среднего числа циклов в n-перестановках (тот же самый результат может быть получен с помощью производящих функций). Эквивалентность этих задач была установлена в 1880 г. Г. Фробениусом (Knuth, 1981, p. 632).

Таким образом имеет место

Теорема 6.1.1. Среднее число j-циклов в n-перестановках равно $1/j$ (не зависит от n).

Доказательство. Пусть c — произвольный фиксированный j-цикл. Пусть b — любая n-перестановка, под действием которой элементы цикла c остаются неподвижными. Пусть $f_c(b) = b \cdot c$; очевидно, что c является j-циклом перестановки $b \cdot c$.

Перестановка $f_c(b)$ является взаимно однозначным отображением из множества всех $(n-j)$-перестановок на множество всех n-перестановок, содержащих цикл c. Поэтому существует $(n-j)!$ n-перестановок, содержащих c в качестве j-цикла. Мы можем $\binom{n}{j}$ способами выбрать j элементов j-цикла c, затем можем $(j-1)!$ способами сформировать из этих j элементов j-цикл. Следовательно, число j-циклов c равно $\binom{n}{j}(j-1)!$.

В сумме общее число вхождений j-циклов во все n-перестановки

равно

$$(n-j)!\binom{n}{j}(j-1)! = \frac{n!}{j},$$

и, разделив его на $n!$, общее число n-перестановок, получаем нужный результат.

Следствие 6.1.2. Среднее число циклов в n-перестановке равно

$$\sum_{1\le j\le n}\frac{1}{j} = H_n,$$

где H_n есть n-е гармоническое число, приблизительно равное $\ln(n)$.

Ниже, в разд. 6.2, мы рассмотрим разложение на множители полиномов над конечным полем, а в разд. 6.3 — процесс подъема. [Представляют интерес работы (Butler, 1954; Calmet et al., 1980, 1982; Camion, 1980; Lauer, 1982; Petr, 1936; Rabin, 1980).]

6.2

Разложение на множители полиномов над конечным полем

Как мы уже видели в предыдущем разделе, разложение полиномов в $\mathbb{Z}_p[x]$, где p — простое число, интересно не только само по себе, но также в связи с тем, что оно полезно при разложении на множители в $\mathbb{Z}[x]$. (Читателю следует заглянуть здесь в разд. 3.3.) Любой полином степени n в $\mathbb{Z}_p[x]$ может быть разложен на множители за конечное число шагов, поскольку существует p^n возможных полиномов степени $< n$, и мы можем просто проверить каждый из них, используя PDF. Однако этот метод проб и ошибок очень неэффективен.

В этом разделе мы подробно обсуждаем эффективный алгоритм Берлекэмпа разложения на множители в $\mathbb{Z}_p[x]$ для *малых* простых p. Этот метод был найден в 1967 г. (Berlekamp, 1967, 1968) и переводит задачу разложения на множители в задачу решения системы линейных уравнений с коэффициентами в $\mathbb{Z}_p[x]$ и нахождения наибольших общих делителей; последние две задачи могут быть решены очень эффективно; таким образом, это преобразование весьма желательно. (Берлекэмп в 1970 г. разработал также другой алгоритм для разложения на множители полиномов в $\mathbb{Z}_p[x]$ при *больших* простых p; этот вопрос кратко обсуждается в разд. 6.2.4.)

Начнем с рассмотрения близкого материала.

6.2.1. Разложение на свободные от квадратов множители над конечными полями

В этом разделе мы модифицируем **PSQFF**, алгоритм, разработанный в разд. 3.2.4, так, чтобы мы имели возможность разлагать полиномы над конечными полями.

Как вы помните, доказательство теоремы 3.2.17 зависело от того факта, что производная полинома, не являющегося константой, не может быть тождественно равной нулю, если характеристика области коэффициентов J равна нулю. Пусть теперь область J имеет простую характеристику p, и рассмотрим полином $p(x) = c_n x^n + \cdots + c_0$. Тогда производная $p'(x) = nc_n x^{n-1} + \cdots + c_1$ равна нулю тогда и только тогда, когда $ic_i = 0$ для каждого i, что имеет место тогда и только тогда, когда $p|i$ или $c_i = 0$ для каждого i. Поэтому $p'(x) = 0$ для некоторого полинома $p(x) \neq 0$ *тогда и только тогда, когда* $p(x)$ — *полином от* x^p. Справедливо следующее обобщение теоремы 3.2.17.

Теорема 6.2.1. Пусть J — область с однозначным разложением на множители произвольной характеристики и $p(x)$ — не являющийся константой примитивный полином над J. Пусть $p(x) = [p_1(x)]^{e_1}[p_2(x)]^{e_2}\ldots[p_n(x)]^{e_n}$ — однозначное разложение $p(x)$ на неприводимые множители, и пусть $\delta_i = 0$, если $e_i p_i'(x) = 0$, в противном случае пусть $\delta_i = 1$. [Условие $e_i p_i'(x) = 0$ имеет место тогда и только тогда, когда $p_i'(x) = 0$ или характеристика области J делит e_i.] Тогда

$$\gcd[p(x), p'(x)] = [p_1(x)]^{(e_1-\delta_1)}\ldots[p_n(x)]^{(e_n-\delta_n)}.$$

Доказательство. Доказательство получается небольшой модификацией доказательства теоремы 3.2.17. □

Предположим теперь, что характеристика области J равна p, и пусть $p(x) = [p_1(x)]^{e_1}[p_2(x)]^{e_2}\ldots[p_n(x)]^{e_n}$ в $J[x]$. Пусть $e = \max(e_1, \ldots, e_n)$, и для $1 \leq i \leq e$ положим

$$s_i(x) = \begin{cases} \prod p_j(x) : e_j = i \text{ и } p_j'(x) \neq 0, & \text{если } p \text{ не делит } i, \\ 1 & \text{в противном случае} \end{cases}$$

и

$$s(x) = \prod\{[p_j(x)]^{e_j} : p|e_j \text{ или } p_j'(x) = 0\}.$$

Тогда

$$p(x) = \{s_1(x)[s_2(x)]^2\ldots[s_e(x)]^e\}s(x),$$

и по теореме 6.2.1 мы имеем $\gcd(s_i(x), s_i'(x)) = 1$, причем следует отметить, что $s'(x) = 0$; кроме того, из теоремы 6.2.1 следует, что

$$\gcd[p(x), p'(x)] = \{s_2(x)[s_3(x)]^2 \dots [s_e(x)]^{e-1}\} s(x),$$

а потому

$$\frac{p(x)}{\gcd[p(x), p'(x)]} = s_1(x) s_2(x) \dots s_e(x).$$

Легко проверить, что алгоритм **PSQFF**, примененный к $p(x)$, даст разложение $s_1(x), \dots, s_e(x)$, и при условии, что $\deg(s(x)) = 0$, он даст на выходе $r(x)$; мы должны также дописать в конце шага 1 «если $t(x) = 1$, то $\{e := 0$; выход$\}$», чтобы учесть случай $p'(x) = 0$. Если $\deg(r(x)) = 0$, то мы получили разложение $p(x)$ на свободные от квадратов множители. Однако если $\deg(r(x)) > 0$, то для получения полного разложения на свободные от квадратов множители требуются дополнительные вычисления. Хотя совсем не очевидно, как это можно сделать для полиномов над произвольной областью J характеристики $p > 0$ (не прибегая к алгоритму полного разложения), мы приводим алгоритм для частного случая, когда J — область полиномов от одной переменной над конечным полем. (В этом месте читателю следует еще раз посмотреть теоремы 3.3.15–3.3.16.)

Две следующие теоремы потребуются для разложения на свободные от квадратов множители в кольце $\mathbb{Z}_p[x]$ и в следующих разделах.

Теорема 6.2.2. Пусть $p(x)$ — полином в кольце $\mathbb{Z}_p[x]$. Тогда

$$[p(x)]^p = p(x^p).$$

Доказательство. Докажем теорему следующим образом. Если $p_1(x)$ и $p_2(x)$ — два полинома по модулю p, то

$$\begin{aligned}[p_1(x) + p_2(x)]^p =& [p_1(x)]^p + \binom{p}{1}[p_1(x)]^{p-1} p_2(x) \\ &+ \dots + \binom{p}{p-1} p_1(x)[p_2(x)]^{p-1} + [p_2(x)]^p \\ =& [p_1(x)]^p + [p_2(x)]^p,\end{aligned}$$

поскольку все биномиальные коэффициенты $\binom{p}{1}, \dots, \binom{p}{p-1}$ делятся на p. Более того, по малой теореме Ферма $c^p \equiv c \pmod p$ для любого целого числа c. Поэтому если

$$p(x) = c_n x^n + c_{n-1} x^{n-1} + \dots + c_0,$$

то мы получаем

$$\begin{aligned}[p(x)]^p &= [c_n x^n]^p + [c_{n-1} x^{n-1}]^p + \dots + [c_0]^p \\ &= c_n x^{np} + c_{n-1} x^{(n-1)p} + \dots + c_0 = p(x^p).\end{aligned}$$

□

Теорема 6.2.3. Пусть $p(x)$ — полином в $\mathbb{Z}_p[x]$. Тогда $p'(x) = 0$ в том и только том случае, когда $p(x)$ есть p-я степень некоторого полинома $q(x)$ в $\mathbb{Z}_p[x]$.

Доказательство. Если $p(x) = [q(x)]^p$, то $p'(x) = p[q(x)]^{p-1}q'(x) = 0$. Обратно, если $p'(x) = 0$, то $p(x)$ может быть записан в виде $p(x) = c_0 + c_p x^p + c_{2p}x^{2p} + \cdots + c_{kp}x^{kp}$. Пусть $q(x) = (c_0)^{1/p} + (c_p)^{1/p}x + (c_{2p})^{1/p}x^2 + \cdots + (c_{kp})^{1/p}x_k$; тогда $q(x)$ лежит в $\mathbb{Z}_p[x]$, и $p(x) = [q(x)]^p$. $\square$

Мы получили следующий алгоритм разложения полиномов на свободные от квадратов множители над конечным полем.

PSQFFF. Разложение полиномов на свободные от квадратов множители над конечным полем (**P**olynomial **S**quare-free **F**actorization over a **F**inite **F**ield)

Вход: $p(x)$ — не являющийся константой нормированный полином в $\mathbb{Z}_p[x]$, $p > 0$ — простое число.

Выход: $s_1(x), \ldots, s_e(x)$ и e, такие, что $p(x) = \prod_{1 \le i \le e}[s_i(x)]^i$ — разложение $p(x)$ на свободные от квадратов множители.

1. [Инициализация] $k := 0$; $m := 1$; $e := 0$.

2. [Основной цикл] $j := 1$; $r(x) := \gcd(p(x), p'(x))$; $t(x) := p(x)/r(x)$; если $t(x) = 1$, то перейти к шагу 7.

3. [Итерация] $e' := jm$; если $e' > e$, то $\{s_{e+1}(x) := s_{e+2}(x) := \ldots s_{e'-1}(x) := 1; e := e'\}$.

4. [Вычисление $s_{e'}(x)$] $v(x) := \gcd(r(x), t(x))$; $s_{e'}(x) := t(x)/v(x)$.

5. [Итерация] Если $v(x) \neq 1$, то $\{r(x) := r(x)/v(x)$; $t(x) := v(x)$; $j := j + 1$; перейтп к шагу 3$\}$.

6. [Конец?] Если $r(x) = 1$, то выход из цикла (а также из алгоритма).

7. [$r'(x) = 0$] $p(x) := [r(x)]^{1/p}$; $k := k + 1$; $m := mp$; перейти к шагу 2.

Время вычислений этого алгоритма проанализировано в разд. 3.2.4, где приводится также пример. Во все время выполнения алгоритма **PSQFFF** выполняется соотношение $m = p^k$, и когда мы попадаем на шаг 6, значение e — это наибольший индекс i, такой, что p^k не делит i и $p(x)$ имеет не являющийся константой делитель порядка i. Мы предполагаем также, что полиномы $r(x)$ и $v(x)$, вычисленные на шагах 2 и 4 соответственно, нормированы.

На шаг 7 мы попадаем, только если $r'(x) = 0$; поэтому в силу теоремы 6.2.3 мы знаем вид полинома $r(x)$ и легко вычисляем $[r(x)]^{1/p}$, пользуясь тождеством $a^{1/p} = a^{p^{n-1}}$.

6.2.2. Вычисление числа неприводимых полиномов над конечными полями

Из теоремы 3.3.20 нам известно, что в $\mathbb{Z}_p[x]$ для любого n существует неприводимый полином степени n; более того, из теоремы 3.3.21 нам известно, что $x^{p^n} - x$ — произведение всех неприводимых полиномов в $\mathbb{Z}_p[x]$, степени которых делят n. Теорема 3.3.21 будет использована ниже для (частичного) разложения на множители разных степеней; в данном разделе мы представим некоторый результат о $i_p(n)$ — числе всех нормированных неприводимых полиномов степени n в $\mathbb{Z}_p[x]$. Заметим, что, согласно теореме 3.3.21, $x^{p^n} - x$, полином степени p^n, имеет в качестве сомножителей $i_p(d)$ неприводимых полиномов степени d для каждого d, делящего n; отсюда следует, что степень произведения всех нормированных неприводимых полиномов над $\mathbb{Z}_p[x]$, степени которых делят n, равна $\sum_{d|n} i_p(d)d = p^n$.

Чтобы получить выражение для $i_p(n)$, нам понадобятся некоторые дополнительные результаты, и прежде всего мы введем следующее (Bender et al., 1975; Berlekamp, 1968; Childs, 1979)

Определение 6.2.4. *Функция Мёбиуса* $\mu(n)$ определяется для $n \geq 1$ следующим образом:

$$\mu(n) = \begin{cases} 1, & \text{если } n = 1, \\ 0, & \text{если } p^e | n \text{ для некоторого простого } p \\ & \text{и некоторого } e > 1, \\ (-1)^r, & \text{если } n \text{ является произведением } r \\ & \text{различных простых чисел.} \end{cases}$$

Заметим, что $\mu(n) = 0$, если n делится на квадрат простого числа, и что $\mu(p) = -1$ для любого простого p. Кроме того, легко проверяется, что если $(m, n) = 1$, то $\mu(mn) = \mu(m)\mu(n)$, т.е. μ — *мультипликативная* функция. (Другим примером мультипликативной функции является функция Эйлера ϕ.)

Имеет место следующая теорема о мультипликативных функциях.

Теорема 6.2.5. Если f — мультипликативная функция и функция F определена соотношением $F(n) = \sum_{d|n} f(d)$, то F — также мультипликативная функция.

Доказательство. Пусть $(m,n)=1$. Тогда каждый делитель d числа mn может быть единственным способом представлен в виде $d=d_1d_2$, где $d_1|m$, $d_2|n$ и $(d_1,d_2)=1$. Поэтому

$$\begin{aligned}F(mn)&=\sum_{d|mn}f(d)=\sum_{\{d_1|m,d_2|n\}}f(d_1d_2)=\sum_{\{d_1|m,d_2|n\}}f(d_1)f(d_2)\\&=\sum_{d_1|m}f(d_1)\sum_{d_2|n}f(d_2)=F(m)F(n).\end{aligned}$$

□

Обратное для данного утверждения также имеет место, но нам оно не понадобится. Мы докажем сейчас теорему о функции μ.

Теорема 6.2.6. $\sum_{d|n}\mu(d)=0$, если $n\neq 1$.

Доказательство. Из теоремы 6.2.5 нам известно, что функция

$$M(n)=\sum_{d|n}\mu(d)$$

мультипликативна, а поскольку $M(p^e)=1$, если $e=0$, в то время как $M(p^e)=1-1+0+\cdots+0$, если $e\geq 1$, мы видим, что $M(n)=0$ если n делится на какое-либо простое число, т.е. $M(n)=0$, если $n>1$. □

Теорема 6.2.7 (формула обращения Мёбиуса). Для любой функции f, определенной на множестве натуральных чисел (не обязательно мультипликативной), если

$$F(n)=\sum_{d|n}f(d) \text{ для любого } n\geq 1,$$

то

$$f(n)=\sum_{d|n}\mu\left(\frac{n}{d}\right)F(d)=\sum_{e|n}\mu(e)F\left(\frac{n}{e}\right).$$

Доказательство. Положим $e=n/d$, $d=n/e$. Равенство двух сумм становится очевидным. Затем по определению F мы имеем

$$\sum_{e|n}\mu(e)F(n/e)=\sum_{e|n}\mu(e)\left[\sum_{d|(n/e)}f(d)\right]=\sum_{e|n}\left[\sum_{d|(n/e)}\mu(e)f(d)\right].$$

Меняя порядок суммирования и замечая, что если $d|(n/e)$, то $de|n$ и, таким образом, $e|(n/d)$, получаем

$$=\sum_{d|n}\left[\sum_{e|(n/d)}\mu(e)f(d)\right]=\sum_{d|n}\left[\sum_{e|(n/d)}\mu(e)\right]f(d).$$

По теореме 6.2.6 в последней сумме коэффициент при $f(d)$ равен 0, кроме случаев, когда $n/d = 1$ или $d = n$. Поэтому эта сумма сводится к единственному члену $f(n)$, и теорема доказана. □

Мы теперь в состоянии получить формулу для $i_p(n)$.

Теорема 6.2.8. Число всех нормированных неприводимых полиномов степени n над $\mathbb{Z}_p[x]$ задается формулой

$$i_p(n) = \left(\frac{1}{n}\right) \sum_{d|n} \mu\left(\frac{n}{d}\right) p^d.$$

Доказательство. Из теоремы 3.3.21 следует, что степень произведения всех нормированных неприводимых полиномов над $\mathbb{Z}_p[x]$, степени которых делят n, равна

$$\sum_{d|n} i_p(d)d = p^n.$$

Применяя теорему 6.2.7 при $F(n) = p^n$ и $f(d) = i_p(d)$, получаем желаемый результат. □

Интересно знать, насколько быстро растет $i_p(n)$. Приведенная ниже табл. 6.2.1 для $p = 7$ взята из (Simmons, 1970).

Таблица 6.2.1

Рост функции $i_p(n)$

n	$i_p(n)$	Общее число нормированных полиномов степени $n (= 7^n)$
1	7	7
2	21	49
3	112	343
4	588	2401
5	3360	16807
6	19544	117649
7	117648	823543

Тесты неприводимости в $\mathbb{Z}_p[x]$**.** Ниже предлагаются два теста для определения, является ли нормированный полином $p(x)$ степени $n > 1$ неприводимым в $\mathbb{Z}_p[x]$; третий тест будет представлен в разд. 6.2.4. Ни один из этих тестов не требует, чтобы полином $p(x)$ был свободным от квадратов, но создается впечатление, что в среднем применение сначала алгоритма разложения на свободные от квадратов множители позволяет экономить время.

Тест 1. Полином $p(x)$ степени $n > 1$ неприводим в $\mathbb{Z}_p[x]$ тогда и только тогда, когда

$$\gcd[p(x), x^{p^i} - x] = 1 \text{ для } i = 1, 2, \ldots, \lfloor \frac{n}{2} \rfloor.$$

Этот тест непосредственно следует из теоремы 3.3.21. Если полином $p(x)$ приводим, то данный тест работает очень быстро. Однако для неприводимых полиномов он становится довольно неэффективным из-за большого числа вычислений gcd в $\mathbb{Z}_p[x]$. Можно уменьшить число вычислений gcd, если воспользоваться следующим тестом.

Тест 2. Полином $p(x)$ степени $n > 1$ неприводим в $\mathbb{Z}_p[x]$ тогда и только тогда, когда (a) $p(x) | (x^{p^n} - x)$ и (b) $\gcd[p(x), x^{p^{n_i}} - x] = 1$ для всех $n_i = n/k_i$, где k_i — простой делитель числа n.

Доказательство. Заметим, что если $p(x)$ — неприводимый полином, то любой корень ξ уравнения $p(x) = 0$ лежит в поле $GF(p^n)$, а поскольку $\xi^{p^n} - \xi = 0$, то $(x - \xi) | (x^{p^n} - x)$, откуда немедленно следует условие (a). Кроме того, поскольку $p(x)$ — неприводимый полином степени n, у него нет корней ни в каком поле $GF(p^{n_i})$, $n_i < n$, откуда непосредственно следует условие (b).

Для доказательства обратного предположим, что выполнены условия (a) и (b). Из условия (a) следует, что все корни полинома $p(x)$ лежат в $GF(p^n)$. Предположим теперь, что у $p(x)$ имеется неприводимый сомножитель $p_1(x)$ степени $n' < n$. Тогда все корни полинома $p_1(x)$ лежат в поле $GF(p^{n'})$, порожденном над $\mathbb{Z}_p$ любым из этих корней. Поэтому $GF(p^{n'}) \subset GF(p^n)$ и $n' | n$. Значит, $n' | n_i$ для некоторого максимального делителя n_i числа n, и все корни полинома $p_1(x)$ лежат в $GF(p^{n_i})$. В этом случае, однако, $p_1(x) | \gcd[p(x), x^{p^{n_i}} - x]$, что противоречит (b). Следовательно, полином $p(x)$ неприводим. □

6.2.3. Разложение полиномов на множители разных степеней (частичное) над конечными полями

В этом разделе мы пользуемся теоремой 3.3.21, чтобы частично разложить на множители свободный от квадратов полином $p(x)$; а

именно мы воспользуемся тем, что неприводимый полином $p(x)$ степени d делит $x^{p^d} - x$, но не делит $x^{p^c} - x$ для $c < d$. Поэтому мы можем выделить неприводимые множители каждой степени отдельно, добавив в алгоритм соотношения

$$p_i(x) = \gcd\left[x^{p^i} - x, \frac{p(x)}{\prod_{1\le j\le i-1} p_j(x)}\right] \quad \text{для} \quad i = 1, 2, \ldots, n$$

и используя теорему 6.2.2.

DDF. Разложение на сомножители разных степеней (**D**istinct **D**egree **F**actorization)

Вход: $p(x)$ — свободный от квадратов полином степени n над $\mathbb{Z}_p[x]$.

Выход: Полиномы $p_d(x)$ над $\mathbb{Z}_p[x]$, $d = 1, 2, \ldots, \lfloor n/2 \rfloor$, такие, что $p(x) = \prod_{1\le d\le \lfloor n/2\rfloor} p_d(x)$; $p_d(x)$ — произведение всех нормированных неприводимых сомножителей степени d полинома $p(x)$.

1. [Инициализация] $q(x) := p(x)$; $r(x) := x$; $d := 0$.

2. [Конец?] {Отметим, что в этом месте $r(x) = x^{p^d} \pmod{q(x)}$; все неприводимые сомножители полинома $q(x)$ различны и их степени $> d$.} Если $d + 1 > (1/2)\deg(q(x))$, то выход [процедура заканчивается, поскольку либо $q(x) = 1$, либо $q(x)$ — неприводимый полином], иначе $\{d := d + 1;\ r(x) := [r(x)]^p \pmod{q(x)}\}$.

3. [Вычисление $p_d(x)$] $p_d(x) := \gcd[r(x) - x, q(x)]$. [Это — произведение всех неприводимых сомножителей полинома $p(x)$, степени которых равны d.] Если $p_d(x) \neq 1$, то $\{q(x) := q(x)/p_d(x)$; $r(x) := r(x) \pmod{q(x)}\}$; перейти к шагу 2. [Ниже мы обсуждаем, как получить сомножители, если степень полинома $p_d(x)$ больше d.]

Пример. Применяя **DDF** к $x^{15} - 1$ в $\mathbb{Z}_2[x]$ (этот полином был разложен на множители в разд. 3.3.1), получаем следующее.

При первом проходе на шаге 1 мы имеем $q(x) = x^{15} - 1$, $r(x) = x$ и $d = 0$; поскольку $d < 7$, на шаге 2 мы имеем $d = 1$ и $r(x) = x^2$. На шаге 3 мы получаем $\gcd[q(x), r(x) - x] = x - 1 = p_1(x)$ и обновляем $q(x)$ и $r(x)$, заменяя их на $x^{14} + x^{13} + \cdots + x + 1$ и x^2 соответственно.

При втором проходе на шаге 2 мы имеем $d = 2$ и $r(x) = x^4$; на шаге 3 тогда $\gcd[q(x), r(x) - x] = x^2 + x + 1 = p_2(x)$, и мы обновляем $q(x)$ и $r(x)$, заменяя их на $x^{12} + x^9 + x^6 + x^3 + 1$ и x^4 соответственно.

Последующие проходы шагов 2 и 3 не дают новых сомножителей, и, следовательно, мы получили всего три сомножителя, а именно $x-1$, x^2+x+1 и $x^{12}+x^9+x^6+x^3+1$, сомножитель 12-й степени, являющийся произведением трех полиномов степени 4. (Заметим, что в $\mathbb{Z}_2[x]$ мы имеем $+1=-1$. См. также историческое замечание 1.)

Мы видим, что в приведенном выше алгоритме требуется вычислять $x^p \pmod{p(x)}$ и

$$x^{p^i} \pmod{p(x)} = \{x^{p^{i-1}} \pmod{p(x)}\}^p \pmod{p(x)}$$

для $i = 2, \ldots, \lfloor n/2 \rfloor$. Поэтому вычисление (частичного) разложения на множители различных степеней может быть выполнено за $O(\lfloor n/2 \rfloor L(p))$ полиномиальных умножений по модулю $p(x)$. Мы можем уменьшить число полиномиальных умножений по модулю $p(x)$ следующим образом (Lenstra, 1982b).

Пусть в i-й строке $n \times n$-матрицы $\mathbf{Q}$ находятся коэффициенты полинома $x^{ip} \pmod{p(x)}$ для $i = 0, 1, \ldots, n-1$; ниже мы отождествляем полиномы степени $< n$ с вектор-строками, образованными их коэффициентами. Тогда имеет место

Теорема 6.2.9. Для любого полинома $v(x) = \sum_{0 \le i \le n-1} v_i x^i$ в $\mathbb{Z}_p[x]$, рассматриваемого как вектор $\mathbf{v}$ своих коэффициентов,

$$\mathbf{v} \cdot \mathbf{Q} = \mathbf{v}^p \pmod{p(x)}.$$

Доказательство. Пусть $\mathbf{Q} = (r_{ij})$, где $i, j = 0, 1, \ldots, n-1$. Тогда

$$\begin{aligned}
\{v(x)\}^p \pmod{p(x)} = v(x^p) \pmod{p(x)} &= \sum_{0 \le i \le n-1} v_i x^{ip} \pmod{p(x)} \\
&= \sum_{0 \le i \le n-1} v_i \sum_{0 \le k \le n-1} r_{ik} x^k \pmod{p(x)} \\
&= \sum_{0 \le k \le n-1} \Big(\sum_{0 \le i \le n-1} v_i r_{ik} \Big) x^k \pmod{p(x)} \\
&= \mathbf{v} \cdot \mathbf{Q}. \qquad \square
\end{aligned}$$

Из приведенной теоремы видно, что $\{x^{p^{i-1}} \pmod{p(x)}\}\mathbf{Q} = x^{p^i} \pmod{p(x)}$, так что $\lfloor n/2 \rfloor$ полиномов $(x^{p^i} - x) \pmod{p(x)}$ для $i = 1, \ldots, \lfloor n/2 \rfloor$ может быть вычислено за $O(n^3)$ операций в конечном поле, если известна матрица $\mathbf{Q}$. Вычисление этой матрицы может быть выполнено за $O(L(p) + n - 2)$ полиномиальных умножений по модулю $p(x)$. Ясно, что для больших p предпочтительней пользоваться матрицей $\mathbf{Q}$.

Другой способ нахождения (частичного) разложения на сомножители разных степеней полинома $p(x)$ предложен в работе (Berlekamp, 1970). Этот метод использует ядро матрицы $\mathbf{Q} - \mathbf{I}$, где $\mathbf{I}$ — единичная $n \times n$-матрица, и может быть расширен до полного разложения в $\mathbb{Z}_p[x]$. Обсуждение этого метода не входит в данную книгу.

Мы видели, что алгоритм **DDF** определяет произведение всех неприводимых сомножителей каждой степени d, и поэтому он сообщает нам, сколько имеется сомножителей каждой степени. Если мы хотим знать *только степени*, то можем действовать следующим образом [$p(x)$ не обязательно свободен от квадратов].

Обозначим через σ_i число различных неприводимых сомножителей степени i полинома $p(x)$, и пусть ν_i — ранг ядра $\mathbf{Q}^i - \mathbf{I}$ для $i = 1, \ldots, n$. Пусть $\mathbf{A} = (a_{ij})$ есть $n \times n$-матрица, где $a_{ij} = \gcd(i, j)$, $\sigma = (\sigma_1, \ldots, \sigma_n)$, $\nu = (\nu_1, \ldots, \nu_n)$. Интересный результат Смита (Smith, 1876) утверждает, что

$$\det(A) = \prod_{0 \le i \le n} \phi(i),$$

где ϕ — функция Эйлера, а в 1956 г. Шварцом [см. также (Schwarz, 1939)] было показано, что

$$\mathbf{A} \cdot \sigma^T = \nu^T.$$

Поэтому матрица $\mathbf{A}$ обратима, и вектор σ однозначно определяется вектором ν. Так например, если для данного полинома степени 8 мы получили $\sigma = (2, 0, 2, 0, 0, 0, 0, 0)$, то мы знаем, что у него два сомножителя степени 1 и два сомножителя степени 3. Дополнительные детали могут быть найдены в работе (Gunji, Arnon, 1981).

Расщепление сомножителей различных степеней. Таким образом мы редуцировали проблему разложения на множители к задаче разделения произведений полиномов одной и той же степени. Чтобы получить метод полного разложения в $\mathbb{Z}_p[x]$, нам нужно научиться расщеплять $p_d(x)$ на неприводимые сомножители, когда $\deg[p_d(x)] > d$. Для этого имеется несколько способов. Мы изложим простой вероятностный алгоритм, предложенный Кантором и Цассенхаузом (Cantor, Zassenhaus, 1981), основанный на следующем тождестве. Если p — произвольное *нечетное* простое число, то

$$\begin{aligned} p_d(x) = &\gcd[p_d(x), q(x)] \cdot \gcd\{p_d(x), [q(x)]^{(p^d-1)/2} + 1\} \\ &\cdot \gcd\{p_d(x), [q(x)]^{(p^d-1)/2} - 1\}, \end{aligned}$$

для всех полиномов $q(x)$, поскольку полином $[q(x)]^{p^d} - q(x)$ делится на все неприводимые полиномы степени d (см. лемму 6.2.10). Приведенное выше тождество основано на полиномиальном разложении, справедливом для *нечетного* p:

$$x^{p^d} - x = x[x^{(p^d-1)/2} + 1][x^{(p^d-1)/2} - 1].$$

Лемма 6.2.10. Пусть $p(x) = p_1(x)p_2(x)\ldots p_r(x)$, $r > 1$, — произведение двух или более различных неприводимых полиномов одной и той же степени d по модулю *нечетного* простого p. Тогда для любого полинома $q(x)$, случайно выбранного среди p^{rd} полиномов степени $< rd$ по модулю p, полином $\gcd\{p(x), [q(x)]^{(p^d-1)/2} - 1\}$ будет собственным сомножителем полинома $p(x)$ с вероятностью

$$1 - \left(\frac{1}{2^r}\right)\left[\left(1 + \frac{1}{p^d}\right)^r + \left(1 - \frac{1}{p^d}\right)^r\right],$$

которая не меньше, чем $4/9$.

Доказательство. Каждый элемент поля $GF(p^d)$ является корнем полинома

$$x^{p^d} - x = x[x^{p^d-1} - 1] = x[x^{(p^d-1)/2} + 1]\{x^{(p^d-1)/2} - 1\}.$$

Поэтому для $(p^d-1)/2$ элементов поля $GF(p^d)$ имеет место равенство $x^{(p^d-1)/2} = 1$, для $(p^d-1)/2$ элементов — равенство $x^{(p^d-1)/2} = -1$ и для одного элемента, $x = 0$, — равенство $x^{(p^d-1)/2} = 0$.

Из греко-китайской теоремы об остатках для полиномов (см. теорему 6.2.12) нам известно, что кольцо $\mathbb{Z}_p[x]/p(x)$ изоморфно прямой сумме $X \quad r$ экземпляров поля $GF(p^d)$. Если $x = (x_1, \ldots, x_r)$ — элемент кольца X, то

$$x^{(p^d-1)/2} = [x_1^{(p^d-1)/2}, \ldots, x_r^{(p^d-1)/2}]$$

имеет в качестве компонент только 0, 1, -1. Число таких x, что $x^{(p^d-1)/2}$ не имеет компонент, равных 1, равно

$$\left(\frac{p^d-1}{2} + 1\right)^r,$$

и, таким образом, число таких x, что $x^{(p^d-1)/2}$ имеет по крайней мере одну компоненту, равную 1, равно

$$p^{rd} - \left(\frac{p^d-1}{2} + 1\right)^r.$$

Из них число таких x, что все компоненты $x^{(p^d-1)/2}$ равны 1, равно

$$\left(\frac{p^d-1}{2}\right)^r.$$

Следовательно, число таких x, что $x^{(p^d-1)/2}$ имеет по крайней мере одну компоненту, равную 1, и по крайней мере одну компоненту, отличную от 1, равно

$$p^{rd} - \left(\frac{p^d-1}{2}+1\right)^r - \left(\frac{p^d-1}{2}\right)^r.$$

Для каждого такого x элемент $x-1$ является ненулевым обратимым элементом кольца X. Если $q(x)$ в $\mathbb{Z}_p[x]$ соответствует одному из таких x значений, то $[q(x)]^{(p^d-1)/2}-1$ не является взаимно простым с $p(x)$ и не делится на $p(x)$; следовательно,

$$\gcd\{p(x), [q(x)]^{(p^d-1)/2}-1\}$$

является собственным делителем полинома $p(x)$. Это происходит с вероятностью

$$\begin{aligned}
&\frac{p^{rd} - [(p^d-1)/2+1]^r - [(p^d-1)/2]^r}{p^{rd}} \\
&= 1 - \left[\frac{(p^d-1)/2+1)}{p^d}\right]^r - \left(\frac{p^d-1}{2p^d}\right)^r \\
&= 1 - \left(\frac{p^d+1}{2p^d}\right)^r - \left(\frac{p^d-1}{2p^d}\right)^r \\
&= 1 - \left(\frac{1}{2^r}\right)\left[\left(1+\frac{1}{p^d}\right)^r + \left(1-\frac{1}{p^d}\right)^r\right].
\end{aligned}$$

Так как $1-1/p^d \le 1$ и $1+1/p^d \le 4/3$ (поскольку $p^d \ge 3$), эта вероятность не меньше, чем

$$1 - \left(\frac{1}{2^r}\right)\left[\left(\frac{4}{3}\right)^r + 1\right] = 1 - \left(\frac{2}{3}\right)^r - \frac{1}{2^r}.$$

Видно, что при $r=2$ мы получаем

$$1 - \left(\frac{1}{4}\right)\left[2+\frac{2}{p^{2d}}\right] \ge 1 - \left(\frac{1}{4}\right)\left[2+\frac{2}{9}\right] = 4/9. \qquad \square$$

Чтобы вычислить $\gcd\{p(x), [q(x)]^{(p^d-1)/2} - 1\}$, мы сначала возведем $q(x)$ в степень $(p^d-1)/2$ по модулю $p(x)$, пользуясь бинарным алгоритмом возведения в степень **E**, описанным в разд. 2.3.2, а затем вычтем 1.

Лемма 6.2.10 основана на разложении $x^{p^d} - x = x[x^{(p^d-1)/2} + 1][x^{(p^d-1)/2} - 1]$, справедливом для всех нечетных простых чисел. Для $p = 2$ имеется другое разложение полинома $x^{p^d} - x$, которое может быть использовано для получения результата, аналогичного лемме 6.2.10.

В поле $GF(p^d)$ полином *след*, $\mathrm{tr}(x)$, определяется формулой $\mathrm{tr}(x) = x + x^p + \cdots + x^{p^{d-1}}$. В этом случае в поле $GF(2^d)$ имеет место разложение $x^{2^d} - x = \mathrm{tr}(x)[\mathrm{tr}(x) + 1]$. В этом легко убедиться, воспользовавшись тем, что $-1 = 1$ в $\mathbb{Z}_2$, и теоремой 3.3.16, согласно которой возведение в квадрат является линейным преобразованием; а именно $\mathrm{tr}(x)[\mathrm{tr}(x)+1] = \mathrm{tr}(x)+[\mathrm{tr}(x)]^2 = x+x^2+\cdots+x^{2^{d-1}}+x^2+x^4+\cdots+x^{2^d} = x + x^{2^d} = x^{2^d} - x$.

Заметим также, что если $p(x)$ — неприводимый полином степени d в $\mathbb{Z}_p[x]$ и $q(x)$ — произвольный полином, то значение полинома $\{q(x) + [q(x)]^p + [q(x)]^{p^2} + \cdots + [q(x)]^{p^{d-1}}\} \pmod{p(x)}$ является целым числом (т.е. полиномом степени ≤ 0). Чтобы в этом убедиться, положим $t(x) = \mathrm{tr}(q(x)) \pmod{p(x)}$. Поскольку $[q(x)]^{p^d} = q(x)$ в поле полиномиальных остатков по модулю $p(x)$, в этом поле имеет место равенство $[t(x)]^p = t(x)$. Поэтому $t(x)$ является одним из p корней уравнения $x^p - x = 0$, а следовательно, $t(x)$ — целое число.

Лемма 6.2.11. Пусть $p(x) = p_1(x)p_2(x)\ldots p_r(x)$, $r > 1$, — произведение двух или более различных неприводимых полиномов одной и той же степени d по модулю 2. Тогда для любого полинома $q(x)$, случайным образом выбранного среди 2^{rd} полиномов степени $< rd$ по модулю 2, полином $\gcd\{p(x), \mathrm{tr}[q(x)]\}$ является собственным делителем полинома $p(x)$ с вероятностью $1 - 1/2^{r-1}$.

Доказательство. Доказательство аналогично доказательству леммы 6.2.10 и остается в качестве упражнения. □

Пример. Попробуем расщепить полином 12-й степени $p(x) = x^{12} + x^9 + x^6 + x^3 + 1$, полученный в примере, следующем за алгоритмом **DDF**, который является произведением трех полиномов степени 4. Следуя лемме 6.2.11, выберем случайным образом полином степени 4, $q(x) = x^4 + x^3 + x^2 + x + 1$, вычислим его след $\mathrm{tr}[q(x)] = q(x) + [q(x)]^2 + [q(x)]^4 + [q(x)]^8$, а затем полином $\gcd\{p(x), \mathrm{tr}[q(x)]\} = x^4 + x^3 + x^2 + x + 1$, являющийся делителем полинома $p(x)$ (повезло!). Теперь нам осталось разложить полином $p(x)/\gcd\{p(x), \mathrm{tr}[q(x)]\} =$

$x^8 - x^7 + x^5 - x^4 + x^3 - x + 1$ на два сомножителя степени 4; читателю остается закончить пример (и испытать таким образом свое счастье).

Заметим, что для *любого* p имеет место разложение

$$x^{p^d} - x = \prod_{0 \le s < p} [\mathrm{tr}(x) - s],$$

и лемма 6.2.11 может быть использована для расщепления сомножителя в разложении полинома $p(x)$ на множители разных степеней так же, как и лемма 6.2.10, т.е. в этом случае мы используем соотношение $\prod_{0 \le s < p} \gcd\{p_d(x), \mathrm{tr}[q(x)] - s\} = p_d(x)$.

Из лемм 6.2.10 и 6.2.11 мы видим, что для нахождения всех сомножителей полинома $p(x)$ не требуется много времени. Однако в следующем разделе мы представим детерминистический метод решения этой задачи.

6.2.4. Алгоритм Берлекэмпа разложения на множители над конечными полями

В этом разделе мы рассмотрим разложение на множители нормированного свободного от квадратов полинома $p(x)$ в $\mathbb{Z}_p[x]$, т.е. для данного полинома $p(x)$ с коэффициентами из множества $\{0, 1, 2, \ldots, p-1\}$ мы хотим найти его неприводимые сомножители $p_i(x)$, $i = 1, 2, \ldots, r$, такие, что

$$p(x) = p_1(x)p_2(x) \ldots p_r(x).$$

Идея Берлекэмпа состоит в использовании греко-китайской теоремы об остатках, которая, как мы уже убедились в разд. 3.1.3, справедлива не только для целых чисел, но и для полиномов.

Теорема 6.2.12 (греко-китайская теорема об остатках для полиномов). Пусть $p_1(x), \ldots, p_r(x)$ — полиномы из кольца $\mathbb{Z}_p[x]$, причем $p_j(x)$ взаимно прост с $p_k(x)$ для всех $j \ne k$. [Эквивалентно, мы можем взять в качестве $p_i(x)$ различные неприводимые полиномы над этим полем], и пусть $s_1(x), \ldots, s_r(x)$ — *произвольные* полиномы в $\mathbb{Z}_p[x]$. Тогда существует единственный полином $t(x)$ в $\mathbb{Z}_p[x]$, такой, что

$$\deg[t(x)] < \deg(p_1(x)] + \cdots + \deg(p_r(x)]$$

[т.е. полином $t(x)$ определен по модулю $p_1(x)p_2(x) \ldots p_r(x)$] и

$$t(x) \equiv s_i(x) \pmod{p_i(x)}, \ 1 \le i \le r.$$

Другими словами, отображение, ставящее в соответствие каждому полиному $t(x)$ из $\mathbb{Z}_p[x]/p_1(x) \ldots p_r(x)$ r-вектор $[s_1(x), \ldots, s_r(x)]$, где $t(x) \equiv s_i(x) \pmod{p_i(x)}$, $1 \le i \le r$, является биекцией между $\mathbb{Z}_p[x]/p_1(x) \ldots p_r(x)$ и $\mathbb{Z}_p[x]/p_1(x) + \mathbb{Z}_p[x]/p_2(x) + \cdots + \mathbb{Z}_p[x]/p_r(x)$.

Доказательство. Используем расширенный алгоритм Евклида для определения полиномов $m_i(x)$, таких, что $m_i(x)\prod_{j\neq i}p_i(x)\equiv 1 \pmod{p_i(x)}$. Полагая $t(x):=\sum_i[s_i(x)m_i(x)\prod_{j\neq i}p_i(x)]$, имеем $t(x)\equiv s_i(x) \pmod{p_i(x)}$, $1\le i\le r$. Если бы $t_1(x)$ также был решением этой системы сравнений, то полином $t_1(x)-t(x)$ делился бы на каждый $p_i(x)$, а потому $t_1(x)\equiv t(x) \pmod{\prod_i p_i(x)}$. □

Отображение $t(x)\equiv s_i(x) \pmod{p_i(x)}$, используемое в теореме 6.2.12, эквивалентно утверждению «$t(x)\equiv s_i(x) \pmod{p_i(x)}$ и p]», поскольку мы рассматриваем полиномиальную арифметику по модулю p.

Теорема 6.2.12 утверждает, что для произвольного r-вектора $(s_1,\ldots,s_r)$ целых чисел по модулю p существует единственный полином $t(x)$, такой, что

$$t(x)\equiv s_1 \pmod{p_1(x)},\ldots,t(x)\equiv s_r \pmod{p_r(x)},$$
$$\deg[t(x)]<\deg[p_1(x)]+\cdots+\deg[p_r(x)]=\deg[p(x)], \qquad \text{(B1)}$$

где, как мы уже говорили, $p(x)=p_1(x)p_2(x)\ldots p_r(x)$. Заметим, что полином $t(x)$, определенный в (B1), позволяет получить информацию о сомножителях полинома $p(x)$, поскольку, если $r\ge 2$ и $s_1\neq s_2$, то $\gcd[p(x),t(x)-s_1]$ делится на $p_1(x)$ и не делится на $p_2(x)$. Проанализируем соотношения (B1) более тщательно. Заметим, что по теореме 6.2.2 полином $t(x)$ удовлетворяет условию $[t(x)]^p\equiv s_j^p=s_j\equiv t(x) \pmod{p_j(x)}$ для $1\le j\le r$ и потому

$$[t(x)]^p\equiv t(x) \pmod{p(x)}, \quad \deg[t(x)]<\deg[p(x)]. \qquad \text{(B2)}$$

Более того, справедлив следующий результат.

Теорема 6.2.13. В конечном поле $GF(p)$ имеет место разложение

$$x^p-x=\prod_{s\in GF(p)}(x-s). \qquad \text{(B3)}$$

Доказательство. Для любого элемента $s\in GF(p)$ по малой теореме Ферма мы имеем $s^p=s$ [равенство имеет место, поскольку вычисления производятся в $GF(p)$]; значит, s является корнем полинома x^p-x, или, эквивалентно, $(x-s)$ является делителем полинома x^p-x. Поскольку это верно для всех $s\in GF(p)$, произведение $\prod_{s\in GF(p)}(x-s)$ является делителем полинома x^p-x. Однако

$$\deg[\prod_{s\in GF(p)}(x-s)]=p=\deg(x^p-x),$$

а поскольку оба этих полинома нормированы,

$$x^p - x = \prod_{s \in GF(p)} (x - s).$$

□

Следствие 6.2.14. Для любого полинома $t(x)$ над $GF(p)$ имеет место равенство

$$[t(x)]^p - t(x) = \prod_{s \in GF(p)} [t(x) - s]. \tag{B4}$$

Если полином $t(x)$ удовлетворяет условию (B2), то $p(x)$ делит левую часть соотношения (B4), так что каждый неприводимый сомножитель полинома $p(x)$ должен делить один из p взаимно простых сомножителей в правой части соотношения (B4). Поэтому все решения сравнения (B2) должны иметь вид (B1) для некоторых $s_1, \ldots, s_r$; поскольку мы можем p^r способами выбрать элементы s_i, существует в точности p_r решений сравнения (B2). [Может показаться, что найти все решения сравнения (B2) очень сложно, но в действительности это не так, поскольку множество решений сравнения (B2) замкнуто относительно сложения.] Когда мы найдем все решения сравнения (B2), мы можем разложить на множители полином $p(x)$ в $\mathbb{Z}_p[x]$, как указывает следующая теорема.

Теорема 6.2.15. Пусть $p(x)$ и $t(x)$ — два нормированных полинома над $GF(p)$, такие, что

$$[t(x)]^p \equiv t(x) \pmod{p(x)}, \quad \deg[t(x)] < \deg[p(x)].$$

Тогда

$$p(x) = \prod_{s \in GF(p)} \gcd[p(x), t(x) - s]. \tag{B5}$$

Доказательство. По предположению

$$[t(x)]^p \equiv t(x) \pmod{p(x)},$$

а следовательно, $p(x)$ делит $[t(x)]^p - t(x)$. Поэтому

$$p(x) = \gcd\{p(x), [t(x)]^p - t(x)\} = \gcd\{p(x), \prod_{s \in GF(p)} [t(x) - s]\}$$

по следствию 6.2.14. Кроме того, $\gcd[t(x) - s, t(x) - t] = 1$ для $s \neq t$, а полиномы $\gcd[p(x), t(x) - s]$ и $\gcd[p(x), t(x) - t]$ также взаимно просты. Поэтому $\gcd\{p(x), \prod_{s \in GF(p)} [t(x) - s]\} = \prod_{s \in GF(p)} \gcd[p(x), t(x) - s]$ (см. также упр. 1 к этому разделу). □

Степень каждого сомножителя в правой части соотношения (B5) не более степени полинома $t(x)$, которая, в свою очередь, $< \deg[p(x)]$. Таким образом, в правой части соотношения (B5) должно быть не менее двух нетривиальных делителей полинома $p(x)$, и (B5) представляет собой *нетривиальное разложение* полинома $p(x)$. Кроме того, если $t(x)$ — скаляр (т.е. $\deg[t(x)] = 0$), то

$$p(x) = \gcd[p(x), 0] \cdot \prod_{s \neq 0,\, s \in GF(p)} \gcd[p(x), s] = p(x) \cdot \prod_{s \neq 0} 1.$$

Поэтому (B5) — формула полного разложения, и она является основной формулой, используемой в алгоритме Берлекэмпа.

Подведем итоги. Пусть $p(x)$ — свободный от квадратов полином степени n, который мы хотим разложить на множители над полем $GF(p)$, и предположим, что каким-то образом мы можем найти полиномы $t(x)$ в кольце $\mathbb{Z}_p[x]$, $1 \leq \deg[t(x)] < n$, такие, что $[t(x)]^p \equiv t(x) \pmod{(p(x))}$. [Заметим, что если степень полинома $t(x) \geq 1$, то $[t(x)]^p - t(x) \neq 0$, поскольку коэффициент при наивысшей степени x не равен нулю.] Из теоремы 6.2.13 мы знаем, что у полинома $x^p - x$ в поле $GF(p)$ имеется p корней, а именно $x = 0, 1, 2, \ldots, p-1$; следовательно, полином $x^p - x$ разлагается на множители по модулю p следующим образом: $x^p + x = x(x-1)(x-2)\ldots(x-p1)$. Заменяя x на $t(x)$, получаем разложение

$$[t(x)]^p - t(x) = t(x)[t(x) - 1][t(x) - 2]\ldots[t(x) - p + 1]$$

в кольце $\mathbb{Z}_p[x]$. Так как полином $p(x)$ делит $[t(x)]^p - t(x)$, то

$$p(x) = \gcd\{p(x), [t(x)]^p - t(x)\}.$$

Кроме того, поскольку полиномы $t(x) - s$ и $t(x) - t$ взаимно просты при $s \neq t$, мы имеем (по следствию 6.2.14, теореме 6.2.15 и упр. 1)

$$p(x) = \gcd\{p(x), [t(x)]^p - t(x)\} = \prod_{s \in GF(p)} \gcd[p(x), t(x) - s],$$

что является нетривиальным разложением на множители полинома $p(x)$ над $GF(p)$.

Таким образом, основная задача состоит в определении полиномов $t(x)$. Это делается путем решения *системы линейных уравнений*, и введенная в разд. 6.2.3 матрица $\mathbf{Q}$ оказывается очень полезной; в частности, очень важную роль играет нуль-пространство (определенное ниже) матрицы $\mathbf{Q} - \mathbf{I}$, где $\mathbf{I}$ — единичная $n \times n$-матрица.

Система линейных уравнений, необходимая для определения полиномов $t(x)$, таких, что $p(x)$ делит $[t(x)]^p - t(x)$, получается следующим образом. Пусть

$$t(x) = t_0 + t_1 x + \cdots + t_{n-1}x^{n-1},$$

где t_i, $i = 0, \ldots, n-1$, — коэффициенты, которые требуется найти. Чтобы проверить, делит ли $p(x)$ полином $[t(x)]^p - t(x)$, взглянем сначала на $[t(x)]^p$; по теореме 6.2.2

$$[t(x)]^p = t_0 + t_1 x^p + t_2 x^{2p} + \cdots + t_{n-1}x^{(n-1)p}. \tag{B6}$$

Деля x^{ip} на $p(x)$, получаем

$$x^{ip} = p(x)q_i(x) + r_i(x), \quad i = 0, 1, 2, \ldots, n-1, \tag{B7}$$

где $r_i(x) = r_{i,0} + r_{i,1}x + \cdots + r_{i,n-1}x^{n-1}$. [Заметим, что полиномы $q_i(x)$ и $r_i(x)$ легко вычислить, если мы знаем $p(x)$.]

Теперь, заменяя x^{ip} в (B6) соответствующими выражениями из (B7), получаем

$$[t(x)]^p = t_0 r_0(x) + t_1 r_1(x) + \cdots + t_{n-1}r_{n-1}(x) + [\text{кратное } p(x)].$$

Таким образом, $p(x)$ делит $[t(x)]^p - t(x)$ тогда и только тогда, когда $p(x)$ делит полином

$$\begin{aligned} &t_0 r_0(x) + t_1 r_1(x) + \cdots + t_{n-1}r_{n-1}(x) - (t_0 + t_1 x + \cdots + t_{n-1}x^{n-1}) \\ &= t_0[r_0(x) - 1] + t_1[r_1(x) - x] + \cdots + t_{n-1}[r_{n-1}(x) - x^{n-1}], \end{aligned} \tag{B8}$$

степень которого $\leq n-1$. Поэтому полином $p(x)$ степени n будет делить (B8) тогда и только тогда, когда последний равен нулю.

Полагая (B8) равным нулю и собирая коэффициенты при $1, x, x^2, \ldots, x^{n-1}$, мы получаем систему n линейных уравнений от n неизвестных $t_0, t_1, \ldots, t_{n-1}$; эти неизвестные суть коэффициенты полинома $t(x)$, такого, что $p(x)$ делит $[t(x)]^p - t(x)$.

Пусть

$$\mathbf{Q} = \begin{bmatrix} r_{0,0} & r_{0,1} & \ldots & r_{0,n-1} \\ r_{1,0} & r_{1,1} & \ldots & r_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n-1,0} & r_{n-1,1} & \ldots & r_{n-1,n-1} \end{bmatrix} \tag{B9}$$

— матрица, строки которой образуют коэффициенты полиномов-остатков $r_0(x), \ldots, r_{n-1}(x)$. (*Замечание.* Сначала выписываются коэффициенты меньших степеней x.) Тогда имеет место

Теорема 6.2.16. Полином $t(x) = t_0 + t_1 x + \cdots + t_{n-1}x^{n-1}$ является решением сравнения $[t(x)]^p \equiv t(x) \pmod{p(x)}$ тогда и только тогда, когда

$$(t_0, t_1, \ldots, t_{n-1})\mathbf{Q} = (t_0, t_1, \ldots, t_{n-1}), \quad \text{или}$$
$$(t_0, t_1, \ldots, t_{n-1})(\mathbf{Q} - \mathbf{I}) = (0, 0, \ldots, 0). \tag{B10}$$

Доказательство. Доказательство следует из того, что (B10) имеет место тогда и только тогда, когда

$$t(x) = \sum_j t_j x^j = \sum_j \sum_k t_k r_{k,j} x^j \equiv \sum_k t_k x^{p^k}$$
$$= t(x^p) \equiv [t(x)]^p \pmod{p(x)}.$$

(См. также теорему 6.2.9.) □

Пусть N — множество векторов $\mathbf{t} = (t_0, t_1, \ldots, t_{n-1})$, таких, что $\mathbf{t}(\mathbf{Q} - \mathbf{I}) = \mathbf{0}$, где $\mathbf{t}$ — вектор коэффициентов полинома $t(x)$, а $\mathbf{0}$ — n-мерный нулевой вектор. Тогда N называется *нуль-пространством* матрицы $\mathbf{Q} - \mathbf{I}$. Пусть $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r\}$ — множество векторов в N, таких, что каждый вектор $\mathbf{a}$ из N является линейной комбинацией векторов $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r$ [заметим, что для каждого вектора $\mathbf{b}_k$ имеется соответствующий полином $b_k(x)$], т.е. для любого вектора $\mathbf{a}$ из N в $\mathbb{Z}_p$ существуют числа $x_1, \ldots, x_r$, такие, что $\mathbf{a} = x_1\mathbf{b}_1 + \cdots + x_r\mathbf{b}_r$. Наименьшее r, для которого существует такое множество $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r\}$, называется *размерностью* пространства N, а само множество называется базисом нуль-пространства. (См. также приложение.)

Таким образом, в силу теоремы 6.2.16 нахождение подходящих полиномов $t(x)$ эквивалентно определению нуль-пространства матрицы $\mathbf{Q} - \mathbf{I}$ (см. также историческое замечание 2). Как мы убедимся ниже, мы легко можем вычислить нуль-пространство и, следовательно, можем применить теорему 6.2.15 и вычислить сомножители полинома $p(x)$. Однако как мы узнаем, что нашли полное разложение полинома $p(x)$? Ответ дается следующей теоремой.

Теорема 6.2.17. Число различных неприводимых сомножителей $p_i(x)$ полинома $p(x)$ в $\mathbb{Z}_p[x]$ равно размерности r нуль-пространства матрицы $\mathbf{Q} - \mathbf{I}$.

Доказательство. Полином $p(x)$ делит $\prod_{s \in GF(p)} \gcd[p(x), t(x) - s]$ тогда и только тогда, когда каждый полином $p_i(x)$ делит $t(x) - s_i$ для некоторого $s_i \in \mathbb{Z}_p$. Из теоремы 6.2.12 следует существование для

данных $s_1, \ldots, s_r \in \mathbb{Z}_p$ единственного полинома $t(x)$ $[\mathrm{mod}\ p(x)]$, такого, что $t(x) \equiv s_i\ [\mathrm{mod}\ p_i(x)]$. Мы можем p^r способами выбрать элементы s_i, и, как мы уже отмечали, существует в точности p^r решений сравнения $[t(x)]^p - t(x) \equiv 0\ [\mathrm{mod}\ p(x)]$. Из теоремы 6.2.16 нам известно, что $t(x)$ является решением (B2) тогда и только тогда, когда

$$(t_0, t_1, \ldots, t_{n-1})\mathbf{Q} = (t_0, t_1, \ldots, t_{n-1}), \quad \text{или}$$
$$(t_0, t_1, \ldots, t_{n-1})(\mathbf{Q} - \mathbf{I}) = (0, 0, \ldots, 0).$$

У этой системы имеется p^r решений. Таким образом, размерность нуль-пространства матрицы $\mathbf{Q} - \mathbf{I}$ равна r и равна числу различных нормированных неприводимых сомножителей полинома $p(x)$, а ранг матрицы $\mathbf{Q} - \mathbf{I}$ равен $n - r$. (См. также упр. 2 к этому разделу.)

Кроме того, теорема 6.2.17 дает нам третий тест неприводимости (два первых теста см. в разд. 6.2.2).

Тест 3. Полином $p(x)$ неприводим в $\mathbb{Z}_p[x]$ тогда и только тогда, когда нуль-пространство матрицы $\mathbf{Q}-\mathbf{I}$ одномерно и $\gcd[p(x), p'(x)] = 1$.

Доказательство. По теореме 6.2.17 нуль-пространство матрицы $\mathbf{Q}-\mathbf{I}$ одномерно тогда и только тогда, когда $p(x) = [p_1(x)]^k$, т.е. является степенью неприводимого полинома. Тогда $r = 1$ и $p(x)$ неприводим в том и только том случае, когда $\gcd[p(x), p'(x)] = 1$. □

Теорема 6.2.18. Пусть $p(x) = p_1(x) \ldots p_r(x)$ в $\mathbb{Z}_p[x]$ и $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r\}$ — базис нуль-пространства матрицы $\mathbf{Q}-\mathbf{I}$. Тогда для каждого $j \neq j'$, $1 \leq j < j' \leq r$, существуют k, $1 \leq k \leq r$, и $s \in \mathbb{Z}_p$, такие, что $p_j(x)$ делит, а $p_j'(x)$ не делит $\gcd[p(x), b_k(x) - s]$.

Доказательство. Прежде всего заметим, что в нуль-пространстве матрицы $\mathbf{Q} - \mathbf{I}$ существует вектор, j-я компонента которого отличается от его j'-й компоненты. Следовательно, существует k, $1 \leq k \leq r$, такое, что

$$b_k(x)\ [\mathrm{mod}\ p_j(x)] \neq b_k(x)\ [\mathrm{mod}\ p_{j'}(x)].$$

Это можно также заключить по противоречию, к которому мы придем в противном случае. Именно, если для всех k имеет место равенство, то, поскольку любое решение уравнения (B2) является линейной комбинацией $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r\}$ с коэффициентами из $\mathbb{Z}_p$, для любого такого решения $\mathbf{b}$ существует элемент $s \in \mathbb{Z}_p$, такой, что $b(x) \equiv s\ [\mathrm{mod}\ p_j(x)]$, $b(x) \equiv s\ [\mathrm{mod}\ p_{j'}(x)]$ [см. также (B1)]. Однако существует решение уравнения (B2), такое, что $b(x) \equiv 0\ [\mathrm{mod}\ p_j(x)]$

и $b(x) \equiv 1 \pmod{p_{j'}(x)}$, и это противоречие доказывает выписанное выше соотношение. Полагая $b_k(x) \pmod{p_j(x)} = s \in \mathbb{Z}_p$, получаем, что $p_j(x) | [b_k(x) - s]$ и $p_{j'}(x)$ не делит $[b_k(x) - s]$. □

Из предыдущего обсуждения ясно, как полином $p(x)$ над конечным полем разлагать на неприводимые множители.

BA. Алгоритм Берлекэмпа (**B**erlekamp's **A**lgorithm)

Вход: Нормированный свободный от квадратов полином $p(x)$ над $GF(p)$, $\deg[p(x)] = n$.

Выход: Неприводимые сомножители полинома $p(x)$ над $GF(p)$.

1. [Построение матрицы $\mathbf{Q}$] Построить $n \times n$-матрицу $\mathbf{Q}$ так, как описано в (B9). Как показано ниже, это можно сделать одним из двух способов в зависимости от того, насколько велико число p.

2. [Триангуляризация $\mathbf{Q} - \mathbf{I}$] Привести матрицу $\mathbf{Q} - \mathbf{I}$ к треугольному виду, вычислив ее ранг $n - r$ и найдя нуль-пространство матрицы $\mathbf{Q} - \mathbf{I}$, т.е. найти r линейно независимых векторов $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r$, таких, что $\mathbf{b}_j[\mathbf{Q} - \mathbf{I}] = \mathbf{0}$ для $1 \le j \le r$. [Первый вектор всегда может быть выбран в виде $(1, 0, \ldots, 0)$, что представляет тривиальное решение $b_1(x) = 1$ уравнения (B2). Приведение к треугольному виду может быть осуществлено так, как описано в разд. 5.3.3, или с использованием представленного ниже алгоритма **NS**.] *В этой точке r — это число неприводимых сомножителей полинома $p(x)$*, поскольку решениями уравнения (B2) являются p^r полиномов, соответствующих векторам $a_1\mathbf{b}_1 + a_2\mathbf{b}_2 + \cdots + a_r\mathbf{b}_r$ при любом выборе целых чисел $0 \le a_1, \ldots, a_r \le p$. Поэтому, если $r = 1$, то полином $p(x)$ неприводим, и алгоритм заканчивает работу.

3. [Вычисление сомножителей] Пусть $b_2(x)$ — полином, соответствующий вектору $\mathbf{b}_2$. Вычислим $\gcd[p(x), b_2(x) - s]$ для всех $s \in GF(p)$. В результате по теореме 6.2.15 получим нетривиальное разложение полинома $p(x)$. Если с использованием $b_2(x)$ получено менее r сомножителей, вычислим $\gcd[w(x), b_k(x) - s]$ для всех $s \in GF(p)$ и всех сомножителей $w(x)$, найденных к данному времени, для $k = 3, 4, \ldots, r$, пока не найдем r сомножителей. Теорема 6.2.18 гарантирует, что таким образом мы найдем все сомножители полинома $p(x)$. Если p мало, то вычисления на данном шаге весьма эффективны. Однако для больших p (например, $p > 25$) может быть предложен лучший способ, разбираемый ниже.

Анализ времени работы алгоритма ***BA***. Число вычислений, выполняемых на шаге 2 алгоритма Берлекэмпа, равно $O(n^3)$. (Это было показано при доказательстве анализа времени вычислений алгоритма **ASPRS** в разд. 5.3.3.) Из разд. 3.2.2 нам известно, что $O(n^3)$ мажорирует время вычисления gcd; поскольку требуется не более p вычислений gcd для каждого вектора $\mathbf{b}$ из базиса и не более r из этих вычислений gcd будут нетривиальны, получаем

$$t_{\mathbf{BA}}[p(x)] = O(prn^3).$$

Читателю следует отметить зависимость этого алгоритма от p. Для больших p алгоритм неэффективен. Кроме того, из следствия 6.1.2 нам известно, что среднее число r сомножителей приблизительно равно $\ln(n)$.

Пример. Разложим на множители над $GF(13)$ свободный от квадратов полином $p(x) = 8x^4 + 6x^3 + 8x^2 + 3x + 12$ или эквивалентный ему нормированный полином $x^4 + 4x^3 + x^2 + 2x + 8$ (полученный домножением полинома $p(x)$ на $8^{-1} = 5$). Это достаточно длинный пример, но читателю следует основательно его изучить.

Вычислим сначала обратный каждого ненулевого элемента поля $GF(13) = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$, поскольку мы будем ими пользоваться. Обратные элементы вычисляются с использованием методов, приведенных в гл. 2. Эти элементы суть $\{1, 7, 9, 10, 8, 11, 2, 5, 3, 4, 6, 12\}$.

На первом шаге алгоритма **BA** вычисляется матрица $\mathbf{Q}$, размер которой в этом случае равен 4×4. Первая строка матрицы $\mathbf{Q}$ всегда имеет вид $(1, 0, 0, 0)$, представляя полином $x^0 \pmod{p(x)} = 1$. Вторая строка представляет $x^{13} \pmod{p(x)}$, третья представляет $x^{26} \pmod{p(x)}$ и, наконец, последняя представляет $x^{39} \pmod{p(x)}$. Очевидно, что нам нужно научиться быстро вычислять $x^{k+1} \pmod{p(x)}$, если известно $x^k \pmod{p(x)}$; ниже мы предлагаем такую процедуру общего вида.

Пусть $p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0$, и предположим, что

$$x^k \equiv r_{k,n-1}x^{n-1} + \cdots + r_{k,1}x + r_{k,0} \pmod{p(x)};$$

тогда

$$\begin{aligned} x^{k+1} \equiv & r_{k,n-1}x^n + \cdots + r_{k,1}x^2 + r_{k,0}x \\ \equiv & r_{k,n-1}(-c_{n-1}x^{n-1} - \cdots - c_1 x - c_0) + r_{k,n-2}x^{n-1} + \\ & \cdots + r_{k,1}x^2 + r_{k,0}x \\ \equiv & r_{k+1,n-1}x^{n-1} + \cdots + r_{k+1,1}x + r_{k+1,0} \pmod{p(x)}, \end{aligned}$$

где

$$r_{k+1,j} = r_{k,j-1} - r_{k,n-1}c_j, \quad r_{k,-1} = 0. \tag{B11}$$

Эта простая рекуррентная формула позволяет без особого труда вычислять $x, x^2, x^3, \ldots, x^k \pmod{p(x)}$. Вычисления могут быть выполнены с использованием одномерного массива $(r_{n-1}, \ldots, r_1, r_0)$, если положить в цикле $t := r_{n-1}$, $r_{n-1} := (r_{n-2} - tc_{n-1}) \pmod{p}, \ldots,$ $r_1 := (r_0 - tc_1) \pmod{p}$ и $r_0 := (-tc_0) \pmod{p}$. Таким образом, используя арифметику по модулю 13 для нашего примера, мы получаем следующую таблицу:

k	$r_{k,3}$	$r_{k,2}$	$r_{k,1}$	$r_{k,0}$
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0
4	9	12	11	5
5	2	2	0	6
6	7	11	2	10
7	9	8	9	9
8	11	0	4	6
9	8	6	10	3
10	0	2	0	1
11	2	0	1	0
12	5	12	9	10
13	5	4	0	12

Таким образом, вторая строка $\mathbf{Q}$ равна $(12, 0, 4, 5)$. Аналогично мы определяем $x^{26} \pmod{p(x)}$, $x^{39} \pmod{p(x)}$ и, наконец,

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 12 & 0 & 4 & 5 \\ 12 & 5 & 6 & 12 \\ 1 & 4 & 3 & 6 \end{bmatrix}, \quad \mathbf{Q} - \mathbf{I} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 12 & 12 & 4 & 5 \\ 12 & 5 & 5 & 12 \\ 1 & 4 & 3 & 5 \end{bmatrix}.$$

Если p — большое число, то мы можем получить полиномы $x^k \pmod{p(x)}$ более эффективным способом, чем описанный в (B11); а именно, эти значения могут быть получены за $O[\log(p)]$ операций возведения в квадрат $\bmod p(x)$, т.е. перехода от $x^k \pmod{p(x)}$ к $x^{2k} \pmod{p(x)}$. Операцию возведения в квадрат сравнительно легко

выполнить, если сначала составить дополнительную таблицу значений $x^i \pmod{p(x)}$ для $i = n, n+1, \ldots, 2n-2$. Итак, если

$$x^k \pmod{p(x)} = c_{n-1}x^{n-1} + \cdots + c_1 x + c_0,$$

то

$$x^{2k} \pmod{p(x)} = [c_{n-1}^2 x^{2n-2} + \cdots + (c_1 c_0 + c_1 c_0)x + c_0^2] \pmod{p(x)},$$

где степени $x^{2n-2}, \ldots, x^n$ могут быть заменены полиномами из дополнительной таблицы. Таким образом мы вычисляем $x^p \pmod{p(x)}$, вторую строку матрицы $\mathbf{Q}$. Чтобы получить остальные строки матрицы $\mathbf{Q}$, мы можем вычислить $x^{2p} \pmod{p(x)}$, $x^{3p} \pmod{p(x)}, \ldots$, просто последовательно умножая на $x^p \pmod{p(x)}$ аналогично тому, как мы возводили в квадрат $\bmod p(x)$.

Этим завершается шаг 1 алгоритма Берлекэмпа.

Следующий шаг алгоритма **BA** требует нахождения нуль-пространства матрицы $\mathbf{Q} - \mathbf{I}$. В общем случае пусть $\mathbf{M}$ — матрица размера $n \times n$ над полем, ранг $n - r$ которой нам нужно определить. Кроме того, предположим, что мы хотим найти линейно независимые векторы $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r\}$, такие, что $\mathbf{b}_i\mathbf{M} = \mathbf{0}$, $i = 1, 2, \ldots, r$. Алгоритм этих вычислений основан на наблюдении, что любой столбец матрицы $\mathbf{M}$ можно домножить на ненулевую величину и любое кратное одного из ее столбцов может быть прибавлено к другому столбцу без изменения ранга или векторов $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r\}$. {Заметим, что полиномы $b_i(x)$, соответствующие базисным векторам $\mathbf{b}_i$ нуль-пространства матрицы $\mathbf{Q} - \mathbf{I}$, суть все решения сравнения $[t(x)]^p \equiv t(x) \pmod{p(x)}$.} Таким образом, мы имеем следующий алгоритм приведения к треугольному виду (Knuth, 1981; Lidl et al., 1984).

NS. Алгоритм нуль-пространства (**N**ull-**S**pace algorithm)

Вход: Матрица $\mathbf{M} = (m_{ij})$, $0 \le i, j \le n-1$ размера $n \times n$, элементы которой принадлежат полю.

Выход: Линейно независимые векторы $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r$, такие, что $\mathbf{b}_i\mathbf{M} = \mathbf{0}$, $i = 1, 2, \ldots, r$, где $n - r$ — ранг матрицы $\mathbf{M}$.

1. [Инициализация флагов столбцов] Положить $r := 0$ и $c_0 := c_1 := c_2 := \cdots := c_{n-1} := -1$.

2. [Главный цикл] Для $h = 0, 1, \ldots, n-1$ выполнять {если существует столбец j, такой, что $m_{hj} \neq 0$ и $c_j < 0$, $j = 0, 1, \ldots, n-1$, то выполнять следующее [умножить столбец j матрицы $\mathbf{M}$ на

$-1/m_{hj}$, так что m_{hj} станет равным -1; затем прибавить умноженный на m_{hi} столбец j к столбцу i для всех $i \neq j$; наконец, положить $c_j := h$]. Если не существует столбца j, такого, что $m_{hj} \neq 0$ и $c_j < 0$, $j = 0, 1, 2, \ldots, n-1$, то положить $r := r+1$ и выдать вектор $\mathbf{b}_r = (b_{r,0}, b_{r,1}, \ldots, b_{r,n-1})$, где для $j = 0, 1, \ldots, n-1$

$$b_{r,j} = \begin{cases} m_{hk}, & \text{если } c_k = j > 0, \text{ (если } c_k > 0 \text{ для более чем} \\ & \text{одного } k, \text{ взять } \textit{любой} \text{ из них)}, \\ 1, & \text{если } j = h, \\ 0 & \text{в противном случае}\}. \end{cases}$$

Применяя описанную выше процедуру к вычисленной выше матрице $\mathbf{M} := \mathbf{Q} - \mathbf{I}$, элементы которой принадлежат полю $GF(13)$, мы приходим к следующему (следует помнить, что в алгоритме **NS** строки и столбцы матрицы нумеруются с 0, а *не* с 1).

При $h = 0$ мы получаем на выходе вектор $\mathbf{b}_1 = (1, 0, 0, 0)$, соответствующий полиному-константе 1. При $h = 1$ мы можем взять j равным $0, 1, 2$ или 3, поскольку $c_i = -1$ для $i = 0, 1, 2, 3$; выбор полностью произволен, хотя он влияет на получаемые на выходе векторы. Мы выбираем $j = 0$ и, пользуясь таблицей обратных величин, вычисленной в начале этого примера, получаем $-1/m_{10} = -(1/12) = -12 \equiv 1 \pmod{13}$; операции над столбцами ($\mathrm{col}_0 := 1 \cdot \mathrm{col}_0$, $\mathrm{col}_1 := \mathrm{col}_1 + 12 \cdot \mathrm{col}_0$, $\mathrm{col}_2 := \mathrm{col}_2 + 4 \cdot \mathrm{col}_0$, $\mathrm{col}_3 := \mathrm{col}_3 + 5 \cdot \mathrm{col}_0$), описанные в приведенном выше алгоритме, меняют матрицу $\mathbf{M}$ на матрицу

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ \mathbf{12} & 0 & 0 & 0 \\ 12 & 6 & 1 & 7 \\ 1 & 3 & 7 & 10 \end{bmatrix}.$$

Выделенный жирным шрифтом элемент **12** в строке 1 и столбце 0 означает, что $c_0 = 1$. При $h = 2$ мы можем выбрать $j = 1$ и, действуя аналогичным образом [т.е. вычисляя сначала $-(1/6) = -11 \equiv 2 \pmod{13}$, а затем полагая $\mathrm{col}_1 := 2 \cdot \mathrm{col}_1$, $\mathrm{col}_0 := \mathrm{col}_0 + 12 \cdot \mathrm{col}_1$, $\mathrm{col}_2 := \mathrm{col}_2 + 1 \cdot \mathrm{col}_1$, $\mathrm{col}_3 := \mathrm{col}_3 + 7 \cdot \mathrm{col}_1$], получим матрицу

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ \mathbf{12} & 0 & 0 & 0 \\ 0 & \mathbf{12} & 0 & 0 \\ 8 & 6 & 0 & 0 \end{bmatrix}.$$

Как и выше, выделенный жирным шрифтом элемент **12** в строке 2 и столбце 1 означает, что $c_1 = 2$. Теперь каждый столбец, не содержащий выделенного жирным шрифтом элемента, состоит полностью из нулей, поэтому при $h = 3$ алгоритм дает на выходе вектор $\mathbf{b}_2 = (0, 8, 6, 1)$, соответствующий полиному $x^3 + 6x^2 + 8x$.

Из вида матрицы $\mathbf{M}(:= \mathbf{Q} - \mathbf{I})$ при $h = 3$ ясно, что векторы $\mathbf{b}_1$ и $\mathbf{b}_2$ удовлетворяют уравнению $\mathbf{b}_i\mathbf{M} = \mathbf{0}$. Поскольку выполненные выше вычисления дали два линейно независимых вектора, полином $p(x)$ должен иметь в точности два неприводимых сомножителя над $GF(13)$, которые получаются на шаге 3 алгоритма Берлекэмпа.

На этом завершающем шаге 3 алгоритма **BA** нам требуется выполнить необходимые вычисления наибольших общих делителей, для того чтобы найти два сомножителя полинома $p(x)$, т.е. нам нужно вычислить $\gcd[p(x), b_2(x) - s]$ для *всех* $s \in GF(13)$, где $p(x) = x^4 + 4x^3 + x^2 + 2x + 8$ и $b_2(x) = x^3 + 6x^2 + 8x$. Выполнив вычисления, видим, что два сомножителя получаются при $s = 1$ и $s = 9$, т.е.

$$s = 1: \ \gcd[p(x), b_2(x) - 1] = x^2 + 9x + 9,$$
$$s = 9: \ \gcd[p(x), b_2(x) - 9] = x^2 + 8x + 11.$$

Проверка: $(x^2+9x+9)(x^2+8x+11) = x^4+4x^3+x^2+2x+8 \pmod{13}$.

Ясно, что когда p большое и мы хотим вычислить наибольшие общие делители для всех $s \in GF(p)$, мы должны выполнить огромный объем работы. Кантор и Цассенхауз (Cantor, Zassenhaus, 1981) предложили лучший способ вычислений. Если $t(x)$ — произвольное решение уравнения (B2) и p *нечетно*, то $p(x)|\{[t(x)]^p - t(x)\} = t(x)\{[t(x)]^{(p-1)/2} + 1\}\{[t(x)]^{(p-1)/2} - 1\}$. Это предполагает, что мы будем вычислять

$$\gcd\{p(x), [t(x)]^{(p-1)/2} - 1\}. \tag{B12}$$

При этом с некоторой вероятностью (B12) является нетривиальным делителем полинома $p(x)$. Для определения этой вероятности заметим, что из $\gcd\{p(x), [t(x)]^{(p-1)/2} + / - 1\} = p(x)$ следует, что $\gcd\{p(x), [t(x)]^{(p-1)/2} - / + 1\} = 1$, а значит, вероятность того, что полином $\gcd\{p(x), [t(x)]^{(p-1)/2} - 1\}$ или $\gcd\{p(x), [t(x)]^{(p-1)/2} + 1\}$ будет нетривиальным делителем, равна

$$1 - \mathrm{prob}(\gcd\{p(x), [t(x)]^{(p-1)/2} - 1\} = p(x))$$
$$- \mathrm{prob}(\gcd\{p(x), [t(x)]^{(p-1)/2} + 1\} = p(x)).$$

[Заметим, что не может одновременно быть $\gcd\{p(x), [t(x)]^{(p-1)/2} - 1\} = 1$ и $\gcd\{p(x), [t(x)]^{(p-1)/2} + 1\} = 1$, поскольку $\deg[t(x)] < n$.] Равенство $\gcd\{p(x), [t(x)]^{(p-1)/2} - 1\} = p(x)$ выполняется тогда и только

тогда, когда $p_j(x) | \{[t(x)]^{(p-1)/2} - 1\}$, $1 < j < r$, что имеет место тогда и только тогда, когда $s_j^{(p-1)/2} \equiv 1 \pmod p$, $1 < j < r$ [см. (B1)]. Мы знаем, что в точности $(p-1)/2$ целых чисел s в интервале $0 \le s < p$ удовлетворяют сравнению $s^{(p-1)/2} \equiv 1 \pmod p$. Если $t(x)$ — выбранное случайным образом решение уравнения (B2), все p^r решений которого равновероятны, то вероятность того, что gcd в (B12) совпадает с полиномом $p(x)$, равна $[(p-1)/2p]^r$. Те же самые рассуждения справедливы для $\text{prob}(\gcd\{p(x), [t(x)]^{(p-1)/2} + 1\} = p(x))$, и, таким образом, вероятность того, что нетривиальный делитель полинома $p(x)$ будет получен путем вычисления (B12), равна $1 - 2[(p-1)/2p]^r > 1 - 2(1/2)^r \ge 1/2$ для всех $r \ge 2$ и $p \ge 3$.

Поэтому, если только p не слишком мало, удачная идея — заменить последний шаг алгоритма Берлекэмпа следующей процедурой. Положим $t(x) := a_1 b_1(x) + a_2 b_2(x) + \cdots + a_r b_r(x)$, где коэффициенты a_i выбираются случайным образом в интервале $0 \le a_i < p$. Пусть $p(x) = w_1(x) \ldots w_h(x)$ — текущее частичное разложение полинома $p(x)$, где первоначально h равно 1. Вычисляем

$$g_i(x) = \gcd\{w_i(x), [t(x)]^{(p-1)/2} - 1\}$$

для всех i, таких, что $\deg[w_i(x)] > 1$, после этого заменяем $w_i(x)$ на $g_i(x)[w_i(x)/g_i(x)]$ и увеличиваем значение h до тех пор, пока не будет найден нетривиальный gcd. Этот процесс мы повторяем при различном выборе полиномов $t(x)$ до тех пор пока не достигнем $h = r$. Детали оставляются читателю.

6.3

Подъем (mod p)-разложения до разложения над целыми числами

В этом разделе мы рассмотрим, как поднять разложение $\bmod p$ до разложения над целыми числами, и в этом процессе мы откажемся от ограничения, что полином $p(x)$ в $Z[x]$ нормирован. Читателю следует в этом месте еще раз просмотреть разд. 6.1.2.

Прежде всего давайте соберем вместе различные соображения о разложении полинома $p(x)$ в общем случае. Предположим, что полином $p(x)$ разлагается на множители над целыми числами и что $p(x) = p_1(x)p_2(x)$; тогда по теореме 3.2.16 $p(x) \equiv p_1(x)p_2(x) \pmod p$

для всех простых p, так что за исключением случая $p|\,\mathrm{lc}[p(x)]$ существует нетривиальное разложение по модулю p. Поэтому в попытке восстановить возможное разложение полинома $p(x)$ над целыми числами можно использовать эффективный алгоритм полиномиального разложения в $GF(p)$ [рассматривавшийся в разд. 6.2]. Если, например, для данного полинома $p(x)$ степени 8 выполняются соотношения $p(x) \equiv p_1(x)p_2(x) \pmod{p_1}, \deg[p_1(x)] = 6$ и $\deg[p_2(x)] = 2$, а для другого простого числа p_2 — соотношения $p(x) \equiv p_1(x)p_2(x)p_3(x) \pmod{p_2}$, $\deg[p_1(x)] = 4$, $\deg[p_2(x)] = 3$ и $\deg[p3(x)] = 1$, то, поскольку в разложении $\mod p_2$ нет сомножителей степени 2, полином $p(x)$ должен быть неприводимым над целыми числами.

Приведенный выше пример, может быть, слишком прост и прямолинеен, и следует отметить, что неприводимость далеко не всегда устанавливается так легко. Следует помнить, что для всех $k \geq 2$ Свиннертон-Дайер (Swinnerton-Dyer, 1970) построил полиномы степени 2^k, неприводимые над целыми числами, но полностью разлагающиеся на линейные и квадратичные множители по модулю любого простого числа.

Если мы попытаемся найти сомножители полинома $p(x)$, рассматривая его поведение по модулю разных простых чисел, то в общем случае результаты не так легко свести воедино. Поэтому желательно ограничиться одним простым числом и посмотреть, какие «дивиденды» могут быть из него извлечены при условии, что по нашему убеждению сомножители по модулю этого простого числа имеют нужные степени. Мы в деталях рассмотрим два подхода: выбор одного *большого* простого числа и выбор одного маленького простого числа в комбинации с «подъемом».

Мы начнем с первой идеи, а именно разлагать полином $p(x)$ по модулю *большого* простого p, где p — некоторое число, большее удвоенного значения абсолютной величины коэффициентов всех возможных делителей полинома $p(x)$, т.е. коэффициенты любого истинного разложения $p(x) = p_1(x)p_2(x)$ над целыми числами должны в действительности лежать между $-p/2$ и $p/2$. [Ниже мы вычисляем границы коэффициентов делителей полинома $p(x)$, не вычисляя самих этих делителей.] Тогда все возможные делители над целыми числами могут быть «вычитаны» из делителей по модулю p, которые мы умеем вычислять. За этим подходом скрываются следующие рассуждения. Если мы выберем маленькое простое p и $p(x) \equiv p_1(x)p_2(x) \pmod{p}$, то в общем случае может существовать много способов согласовать это разложение с разложением на множители полинома $p(x)$ над целыми

числами. Например, рассмотрим полином

$$\begin{aligned} p(x) &= 112x^4 + 58x^3 - 31x^2 + 107x - 66 \\ &= (14x^2 - 5x + 11)(8x^2 + 7x - 6) \text{ в } \mathbb{Z}[x]. \end{aligned}$$

По модулю 13 полином $p(x)$ разлагается следующими способами:

$$\begin{aligned} p(x) &\equiv 8(x^2 + 8x + 11)(x^2 + 9x + 9) \pmod{13} \\ &\equiv (8x^2 + 12x + 10)(x^2 + 9x + 9) \pmod{13} \\ &\equiv (x^2 + 8x + 11)(8x^2 + 7x + 7) \pmod{13}. \end{aligned}$$

(См. также пример в разд. 6.2.4.) Заметим, что существует много полиномов, сравнимых с каким-либо сомножителем mod 13, которые могут быть сомножителями полинома $p(x)$ над целыми числами, и для того чтобы обнаружить истинные сомножители в $\mathbb{Z}[x]$, потребуются определенные усилия. [Этой проблемы *не* существует, если мы работаем с нормированным полиномом $p(x)$.]

Итак, предположим теперь, что $p(x) \equiv p_1(x)p_2(x) \pmod{p}$, где p *достаточно велико*, так что коэффициенты любого возможного делителя полинома $p(x)$, степень которого равна степени полинома $p_1(x)$, лежат между $-p/2$ и $p/2$; более того, предположим, что мы выбрали $p_1(x)$ так, что все его коэффициенты лежат в этом интервале. Тогда если $p(x) = P_1(x)P_2(x)$ в $\mathbb{Z}[x]$, где $P_1(x) \equiv p_1(x) \pmod{p}$ и $P_2(x) \equiv p_2(x) \pmod{p}$, то должно выполняться равенство $P_1(x) = p_1(x)$. [В противном случае существует коэффициент полинома $P_1(x)$, отличающийся от соответствующего коэффициента полинома $p_1(x)$ ненулевым кратным числа p. Но тогда этот коэффициент полинома $P_1(x)$ должен быть по абсолютной величине $\geq p/2$, т.е. слишком большим по абсолютной величине, чтобы определять $P_1(x)$ как возможный делитель полинома $p(x)$.] Поэтому если $p(x) \equiv p_1(x)p_2(x) \pmod{p}$, то либо $p_1(x)|p(x)$ в $\mathbb{Z}[x]$, либо разложение $p(x) \equiv p_1(x)p_2(x) \pmod{p}$ не соответствует никакому разложению над целыми числами.

Рассмотрим теперь, как найти подходящее большое p. (В этом месте читателю следует еще раз посмотреть определения различных норм полиномов, введенные в разд. 1.2.)

Ограничения на коэффициенты делителей полинома. Мы начнем с хорошо известной теоремы, которую докажем в общем случае для комплексных коэффициентов.

Теорема 6.3.1. Пусть $p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_0$ — нормированный полином с комплексными коэффициентами, и предположим, что все его комплексные корни по абсолютной величине меньше некоторого положительного вещественного числа b. Если $f(x)$ — нормированный делитель полинома $p(x)$, $\deg[f(x)] = r$, то все коэффициенты полинома $f(x)$ по абсолютной величине $\leq \max_{1\leq k\leq r}\{\binom{r}{k}b^k\}$.

Доказательство. В $\mathbb{C}[x]$ пусть $f(x) = \prod_{1\leq i\leq r}(x+s_i) = x^r + f_{r-1}x^{r-1} + \cdots + f_0$, где $-s_i$, $1 \leq i \leq r$, — комплексные корни полинома $f(x)$. Перемножая разложение $f(x) = \prod_{1\leq i\leq r}(x+s_i)$ и приравнивая коэффициенты, получаем

$$\begin{aligned}
f_{r-1} &= s_1 + s_2 + \cdots + s_r\,,\\
f_{r-2} &= s_1s_2 + s_1s_3 + s_1s_4 + \cdots = \sum_{i<j} s_is_j\,,\\
f_{r-3} &= \sum s_is_js_k\,,\\
&\cdots\\
f_0 &= s_1s_2\ldots s_r\,.
\end{aligned}$$

Поскольку каждый $-s_i$, $1 \leq i \leq r$, является корнем полинома $p(x)$, имеет место неравенство $|-s_i| \leq b$. Поэтому, пользуясь неравенством треугольника, для каждого i мы получаем, что $|f_i|$ не превосходит суммы, полученной заменой всех s_i на b. Например, при $r = 4$

$$\begin{aligned}
|f_3| &= |s_1 + s_2 + s_3 + s_4| \leq |s_1| + |s_2| + |s_3| + |s_4| \leq b + b + b + b = 4b,\\
|f_2| &= |s_1s_2 + s_1s_3 + s_1s_4 + s_2s_3 + s_2s_4 + s_3s_4|\\
&\qquad \leq |s_1s_2| + |s_1s_3| + |s_1s_4| + |s_2s_3| + |s_2s_4| + |s_3s_4| \leq 6b^2,\\
|f_1| &= |s_1s_2s_3| + |s_1s_2s_4| + |s_1s_3s_4| + |s_2s_3s_4| \leq 4b^3,\\
|f_0| &= |s_1s_2s_3s_4| \leq b^4.
\end{aligned}$$

Заменяя каждый s_i на b, получаем полином $f_1(x) = (x+b)^r = \sum_{0\leq k\leq r}\binom{r}{k}b^kx^{r-k}$. Таким образом,

$$\begin{aligned}
|f_{r-1}| &\leq \binom{r}{1}b = rb,\\
|f_{r-2}| &\leq \binom{r}{2}b^2 = \left[\frac{r(r-1)}{2}\right]b^2,\\
&\cdots\\
|f_0| &= \binom{r}{r}b^r = b^r,
\end{aligned}$$

чем заканчивается доказательство теоремы. □

Из теоремы 6.3.1. видно, что, для того чтобы найти хорошую границу для коэффициентов делителей полинома $p(x)$, нам нужно найти хорошую границу для корней $p(x)$. Имеет место следующая

Теорема 6.3.2 (Specht, 1949). Пусть $p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_0$ — нормированный полином с комплексными коэффициентами. Кроме того, пусть $z_1, z_2, \ldots, z_k$ — корни полинома $p(x)$ (с соответствующими кратностями), такие, что $1 \le |z_1| \le |z_2| \le \cdots \le |z_k|$. Тогда

$$\prod_{1 \le i \le k} |z_i| \le |p(x)|_2.$$

Доказательство. Доказательство достаточно громоздко, и мы его опускаем. Оно имеется в работе Шпехта (Specht, 1949) на немецком языке и (другое) в статье Миньотта (Mignotte, 1974) на английском. (См. также историческое замечание 3.) □

Из хорошо известного выражения для коэффициентов полинома замечаем, что для выписанного выше полинома $p(x)$

$$|c_i| \le \binom{n}{i} |z_1 \ldots z_k|$$

и

$$\sum_{0 \le i \le n} |c_i| \le 2^n |z_1 \ldots z_k|.$$

Теперь мы готовы получить границу для коэффициентов делителей полинома $p(x)$.

Теорема 6.3.3 (Mignotte, 1974). Пусть $p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_0$ — нормированный полином в $\mathbb{Z}[x]$, и предположим, что $p(x) = p_1(x)p_2(x)\ldots p_m(x)$, где $p_i(x)$, $1 \le i \le m$, — нормированные полиномы в $\mathbb{Z}[x]$. Тогда

$$\prod_{1 \le i \le m} |p_i(x)|_1 \le 2^d |p(x)|_2, \tag{M1}$$

где $d = \sum_{1 \le i \le m} \deg[pi(x)]$; кроме того, если $p_h(x)$ — один из этих сомножителей, где $p_h(x) = x^h + \cdots + c_{h,1}x + c_{h,0}$, то

$$|c_{h,j}| \le \binom{h}{j} |p(x)|_2, \ j = 0, 1, \ldots, h-1 \tag{M2}$$

Доказательство. Оно непосредственно следует из теорем 6.3.1, 6.3.2 и хорошо известной формулы для коэффициентов полинома, выписанной выше. □

Из приведенных выше рассуждений видно, что большое простое число p должно быть выбрано так, что $p \geq 2^d |p(x)|_2$, так что если $p(x) \equiv p_1(x)p_2(x) \pmod{p}$, то либо $p_1(x)|p(x)$ в $\mathbb{Z}[x]$, либо разложение $p(x) \equiv p_1(x)p_2(x) \pmod{p}$ не соответствует никакому разложению над целыми числами.

Рассмотрим теперь второй подход. Вместо использования больших простых чисел p, которые должны быть чрезвычайно велики, если у полинома $p(x)$ высокая степень или большие коэффициенты (или и то, и другое), мы можем также использовать маленькие простые числа при условии, что полином $p(x) \pmod{p}$ свободен от квадратов. Один из лучших методов разложения полинома $p(x)$ над целыми числами, известных к настоящему времени, состоит в том, чтобы разложить его по модулю p для маленького простого p, а затем расширить (или «поднять») это разложение единственным способом до разложения по модулю p^k для подходящего k, определенного условием (M2).

6.3.1. Линейный и квадратичный подъем

В этом разделе мы предполагаем, что разложили на множители полином $p(x)$ в $\mathbb{Z}_p[x]$ для некоторого маленького простого p и хотим поднять это разложение до разложения над целыми числами.

Пусть $p(x)$ — полином степени n в $\mathbb{Z}_{p^k}[x]$, такой, что $\mathrm{lc}[p(x)] \neq 0$ в $GF(p)$. Кроме того, пусть $g_1(x)$ и $h_1(x)$ — полиномы в $GF(p)[x]$, такие, что $p(x) = g_1(x)h_1(x)$ над $GF(p)$ и $\gcd[g_1(x), h_1(x)] = 1$. Ниже мы предлагаем два метода подъема этого разложения над $GF(p)$ до единственного разложения над $\mathbb{Z}_{p^k} = \mathbb{Z}_q$, т.е. нахождения $g_k(x), h_k(x)$ в $\mathbb{Z}_{p^k}[x]$, таких, что $p(x) = g_k(x)h_k(x)$ над $\mathbb{Z}_{p^k}$ и $g_k(x) \equiv g_1(x)$, $h_k(x) \equiv h_1(x) \pmod{p}$. Для достижения этого нам понадобятся следующие два алгоритма. Первый алгоритм — это хорошо известный расширенный алгоритм Евклида в $GF(p)[x]$, а второй алгоритм решает полиномиальное уравнение $a'(x)g_j(x) + b'(x)h_j(x) = c(x)$ для данных $g_j(x), h_j(x)$ и $c(x)$ в $\mathbb{Z}_{p^j}[x]$.

Алгоритм 6.3.1 (расширенный алгоритм Евклида в $GF(p)[x]$). Для данных $g(x)$ и $h(x)$ в $GF(p)[x]$ этот алгоритм вычисляет единственные (с точностью до обратимых множителей) полиномы $a(x), b(x)$ и $c(x)$ в $GF(p)[x]$, такие, что $g(x)a(x) + h(x)b(x) = c(x) = \gcd[g(x), h(x)]$ над $GF(p)$ при условии, что $\deg[a(x)] < \deg[h(x)] - \deg[c(x)]$ и $\deg[b(x)] < \deg[g(x)] - \deg[c(x)]$. Описание этого алгоритма уже было представлено в разд. 3.2.2 (**ХЕА-Р**).

Алгоритм 6.3.2 (решение полиномиального уравнения в $\mathbb{Z}_{p^j}[x]$). Для данных полиномов $a(x), b(x), c(x), g_j(x), h_j(x)$ в $\mathbb{Z}_{p^j}[x]$, таких, что $g_j(x)a(x)+h_j(x)b(x) = 1$ над $\mathbb{Z}_{p^j}$ и старшие коэффициенты полиномов $a(x), b(x), g_j(x), h_j(x)$ обратимы в $\mathbb{Z}_{p^j}$, этот алгоритм вычисляет полиномы $a'(x)$ и $b'(x)$ в $\mathbb{Z}_{p^j}[x]$, такие, что $g_j(x)a'(x)+h_j(x)b'(x) = c(x)$ над $\mathbb{Z}_{p^j}$, при условии, что $\deg[a'(x)] < \deg[h_j(x)]$, следующим образом[1)].

1. Хотя коэффициенты полиномов принадлежат не полю, а кольцу $\mathbb{Z}_{p^j}$, предположение об обратимости старших коэффициентов позволяет использовать **PDF**, алгоритм деления полиномов над полем, чтобы вычислить в $\mathbb{Z}_{p^j}[x]$ полиномы $q(x)$ и $r(x)$, такие, что $a(x)c(x) = h_j(x)q(x) + r(x)$ и $\deg[r(x)] < \deg[h_j(x)]$.

2. Полагаем $a'(x) := r(x),\ b'(x) := b(x)c(x) + g_j(x)q(x)$.

Тогда

$$\begin{aligned} g_j(x)a'(x) + h_j(x)b'(x) =& g_j(x)[a(x)c(x) - h_j(x)q(x)] \\ &+ h_j(x)[b(x)c(x) + g_j(x)q(x)] \\ =& [g_j(x)a(x) + h_j(x)b(x)]c(x) \\ =& c(x) \text{ в } \mathbb{Z}_{p^j}. \end{aligned}$$

Используя это, мы можем получить следующий алгоритм линейного подъема, вытекающий из доказательства следующей леммы.

Лемма 6.3.4 (Гензель). Пусть $p(x)$ — полином степени n в $\mathbb{Z}[x]$, такой, что $\mathrm{lc}[p(x)] \neq 0$ в $GF(p)$, p — простое число. Пусть $g_1(x)$ и $h_1(x)$ — полиномы в $GF(p)[x]$, такие, что $p(x) = g_1(x)h_1(x)$ над $GF(p)$ и $\gcd[g_1(x), h_1(x)] = 1$ над $GF(p)$. Тогда в $\mathbb{Z}_{p^k}[x]$ существуют единственные полиномы $g_k(x), h_k(x)$, такие, что $p(x) = g_k(x)h_k(x)$ над $\mathbb{Z}_{p^k}$, $\deg[h_k(x)] = \deg[h_1(x)]$ и $g_k(x) \equiv g_1(x)$, $h_k(x) \equiv h_1(x)$ над $GF(p)$.

Доказательство. Пользуясь алгоритмом 6.3.1, найдем полиномы $a(x), b(x)$, такие, что $g_1(x)a(x)+h_1(x)b(x) = 1$. Достаточно показать, как построить $g_j(x), h_j(x)$ в $\mathbb{Z}_{p^j}[x]$ для $j = 2, \ldots, k$, такие, что $p(x) = g_j(x)h_j(x)$ над $\mathbb{Z}_{p^j}$, $\deg[h_j(x)] = \deg[h_1(x)]$ и $g_j(x) \equiv g_1(x)$, $h_j(x) \equiv h_1(x)$ над $GF(p)$. Предположим, что такие $g_j(x), h_j(x)$ существуют для некоторого $j \geq 1$, и пусть $c(x)$ — такой полином в $GF(p)[x]$, что

[1)] Далее этот алгоритм применяется только в случае, когда $\deg[c(x)] < \deg[g_j(x) \cdot h_j(x)]$. Тогда в качестве $a'(x)$ и $b'(x)$ можно взять остатки от деления полиномов $a(x)c(x)$ и $b(x)c(x)$ на $h_j(x)$ и $g_j(x)$ соответственно. — *Прим. перев.*

$p(x) - g_j(x)h_j(x) = p^j c(x)$ над $\mathbb{Z}_{p^{j+1}}$. Пользуясь алгоритмом 6.3.2, вычислим $a'(x)$, $b'(x)$ в $GF(p)[x]$, такие, что $g_1(x)a'(x) + h_1(x)b'(x) = c(x)$ над $GF(p)$. Определим

$$g_{j+1}(x) = g_j(x) + p^j \cdot b'(x),$$
$$h_{j+1}(x) = h_j(x) + p^j \cdot a'(x),$$

где следует отметить, что $\deg[h_{j+1}(x)] = \deg[h_j(x)]$ и $\mathrm{lc}[h_{j+1}(x)] = \mathrm{lc}[h_j(x)]$. Очевидно, что $g_{j+1}(x)$, $h_{j+1}(x)$ — полиномы в $\mathbb{Z}_{p^{j+1}}[x]$, $g_{j+1}(x) \equiv g_1(x)$, $h_{j+1}(x) \equiv h_1(x)$ над $GF(p)$ и

$$\begin{aligned} g_{j+1}(x)h_{j+1}(x) &= [g_j(x) + p^j \cdot b'(x)][h_j(x) + p^j \cdot a'(x)] \\ &= g_j(x)h_j(x) + p^j \cdot [g_j(x)a'(x) + h_j(x)b'(x)] \\ &= g_j(x)h_j(x) + p^j \cdot c(x) \\ &= p(x) \text{ над } \mathbb{Z}_{p^{j+1}}. \end{aligned}$$

□

Очевидно, что доказанная выше лемма дает нам пошаговый метод подъема разложения, т.е. от $GF(p)$ к $\mathbb{Z}_{p^2}, \mathbb{Z}_{p^3}, \ldots, \mathbb{Z}_{p^k}$. Поэтому он называется *алгоритмом линейного подъема*. Заметим, что на практике часто требуется поднимать более двух сомножителей. В этом случае подъем осуществляется повторным применением леммы 6.3.4 к парам сомножителей, один из которых — полином $p_i(x)$, а другой — произведение $p_{i+1}(x)p_{i+2}(x)\ldots p_r(x)$. А именно, если

$$p(x) \equiv p_1(x)p_2(x)\ldots p_r(x) \pmod{p},$$

то мы вначале берем $g_1(x) = p_1(x)$ и $h_1(x) = p_2(x)\ldots p_r(x)$ и поднимаем до $\mathbb{Z}_{p^k}[x]$; затем полагаем $p(x) := p(x)/g_1(x)$ в $\mathbb{Z}_{p^k}$, $g_1(x) := p_2(x)$, $h_1(x) := p_3(x)\ldots p_r(x)$ и поднимаем до $\mathbb{Z}_{p^k}[x]$ и т.д. Если полином $p(x)$ нормирован, то алгоритм может быть модифицирован так, что одновременно может подниматься произвольное число взаимно простых сомножителей.

Читателю следует тщательно изучить два следующих примера; в первом мы используем множество неотрицательных вычетов целых чисел, а во втором — симметричное множество вычетов.

HLLA. Алгоритм линейного подъема Гензеля (**H**ensel **L**inear-**L**ifting **A**lgorithm)

Вход: Простое число p, натуральное число k, а также полиномы $p(x) \in \mathbb{Z}[x]$, $g_1(x)$ и $h_1(x)$, оба $\in GF(p)[x]$, такие, что $p(x) \equiv g_1(x)h_1(x) \pmod{p}$ и $\gcd[g_1(x), h_1(x)] = 1$.

Выход: Полиномы $g_k(x)$ и $h_k(x)$, оба $\in (\mathbb{Z}/p^k)[x]$, такие, что $p(x) \equiv g_k(x)h_k(x) \pmod{p}^k$, а также $g_k(x) \equiv g_1(x) \pmod{p}$ и $h_k(x) \equiv h_1(x) \pmod{p}$.

1. [Инициализация] Положить $g(x) := g_1(x)$, $h(x) := h_1(x)$, а затем применить расширенный алгоритм Евклида в $GF(p)[x]$ (алгоритм 6.3.1, описанный в этом разделе) к полиномам $g(x)$ и $h(x)$ и вычислить полиномы $a(x)$ и $b(x)$, оба $\in GF(p)[x]$, такие, что $g(x)a(x) + h(x)b(x) = 1$ над $GF(p)$.

2. [Основной цикл] Для $i = 2, 3, \ldots, k$ выполнять {вычислить поправку $c(x) := [p(x) - g(x)h(x)]/p^{i-1} \pmod{p}$; затем применить алгоритм 6.3.2 (также описанный в этом разделе) к $g(x), h(x), a(x), b(x)$ и $c(x)$, все $\in GF(p)[x]$, чтобы вычислить полиномы $a'_{i-1}(x)$ и $b'_{i-1}(x)$, такие, что $g(x)a'_{i-1}(x) + h(x)b'_{i-1}(x) = c(x)$ над $GF(p)$, и положить $g(x) := g(x) + p^{i-1}b'_{i-1}(x)$ и $h(x) := h(x) + p^{i-1}a'_{i-1}(x)$}.

3. [Выход] Вернуть $g(x)$ и $h(x)$.

Пример [линейная гензелева конструкция; $p(x)$ не обязательно нормирован]. Рассмотрим полином $p(x) = 112x^4 + 58x^3 - 31x^2 + 107x - 66$ в $\mathbb{Z}[x]$, для которого имеет место сравнение

$$p(x) \equiv (8x^2 + 12x + 10)(x^2 + 9x + 9) \pmod{13}.$$

Мы поднимем это разложение с $GF(13)$ до $\mathbb{Z}_{13^2}$, пользуясь линейной гензелевой конструкцией и арифметикой неотрицательных вычетов. Для лучшего понимания процесса сообщим, что над целыми числами имеет место разложение $p(x) = (8x^2 + 7x - 6)(14x^2 - 5x + 11)$; эти два сомножителя полинома $p(x)$ найдены путем применения алгоритма подъема. Очевидно, что в нашем примере $g_1(x) = 8x^2 + 12x + 10$ и $h_1(x) = x^2 + 9x + 9$. Пользуясь алгоритмом 6.3.1, вычисляем полиномы $a(x) = x + 11$ и $b(x) = 5x + 11$ в $GF(13)[x]$, такие, что $g_1(x)a(x) + h_1(x)b(x) = 1$ над $GF(13)$. [В действительности из алгоритма 6.3.1 мы получаем $a(x) = 5x + 3$, $b(x) = 12x + 3$, но тогда $g_1(x)a(x) + h_1(x)b(x) = 5$ в поле $GF(13)$, следовательно, мы должны подправить их, домножив на соответствующую константу.]

Затем из соотношения $p(x) - g_1(x)h_1(x) = 13c(x)$ (где $g_1(x), h_1(x)$ рассматриваются в $\mathbb{Z}[x]$) получаем $c(x) = 8x^4 + 11x^3 + 9x^2 + 6x + 1$ над $GF(13)$. Заметим, что $p \cdot c(x)$ — это разность между $p(x)$ и $g_1(x)h_1(x)$ в $\mathbb{Z}_{p^2}[x]$, и нам нужно теперь подкорректировать наши сомножители, т.е. прибавить к $g_1(x)$ и $h_1(x)$, чего им не хватает для того, чтобы разность исчезла. Используя алгоритм 6.3.2, вычисляем полиномы

$a'(x) = x+9$ и $b'(x) = 8x^2+12x+6$ в $GF(13)[x]$, такие, что $g_1(x)a'(x)+h_1(x)b'(x) = c(x)$ над $GF(13)$. Затем определяем

$$\begin{aligned} g_2(x) &= g_1(x) + p \cdot b'(x) = (8x^2 + 12x + 10) + 13(8x^2 + 12x + 6) \\ &= 112x^2 + 168x + 88, \\ h_2(x) &= h_1(x) + p \cdot a'(x) = (x^2 + 9x + 9) + 13(x + 9) = x^2 + 22x + 126. \end{aligned}$$

Здесь $p(x) = g_2(x)h_2(x)$ над $\mathbb{Z}_{169=13^2}$, что легко может быть проверено, но ни $g_2(x)$, ни $h_2(x)$ не делят $p(x)$ в $\mathbb{Z}[x]$. Однако заметим, что $g_2(x) = 8(14x^2+21x+11)$, и если мы скомбинируем сомножитель 8 с $h_2(x)$, то получим $8 \cdot h_2(x) = 8(x^2+22x+126) = 8x^2+176x+1008 \equiv 8x^2+7x+163 \equiv 8x^2+7x-6 \pmod{169}$. Здесь $(8x^2+7x-6)|p(x)$ над целыми числами, и, следовательно, это один из сомножителей; разделив, мы получаем второй сомножитель $14x^2-5x+11$.

Следует отметить, что в приведенном выше примере требовались некоторые усилия для определения истинных сомножителей, поскольку полином $p(x)$ не был нормирован. К счастью, для этого имеется простой способ, а именно, из разложения $p(x) = p_1(x)p_2(x)$ следует разложение $\mathrm{lc}[p(x)] \cdot p(x) = p_1'(x)p_2'(x)$, где $\mathrm{lc}[p_1'(x)] = \mathrm{lc}[p_2'(x)] = \mathrm{lc}[p(x)]$. («Я надеюсь, что Вы не будете возражать, если я домножу полином на его старший коэффициент перед тем, как разложить его на множители».) Мы можем действовать, как и прежде, но используя границу для коэффициентов сомножителей полинома $\mathrm{lc}[p(x)] \cdot p(x)$, а не $p(x)$.

Прежде чем мы перейдем ко второму алгоритму подъема, целесообразно сейчас просмотреть различные рассуждения, используемые при попытке разложить на множители полином $p(x) = 112x^4+58x^3-31x^2+107x-66$ над целыми числами (см. также разд. 6.1.2).

Прежде всего нам нужно убедиться, что над целыми числами $p(x)$ свободен от квадратов, т.е. мы должны вычислить $\gcd[p(x), p'(x)]$, где $p'(x) = 448x^3+174x^2-62x+107$. Величина используемых коэффициентов нам известна, так что мы используем один из модулярных gcd-алгоритмов, описанных в разд. 5.1.2, т.е. мы должны вычислить $\gcd[p(x), p'(x)]$ над различными полями $GF(p_i)$, где $p_1, p_2, \ldots, p_k$, таковы, что $p_1 \cdot p_2 \ldots p_k > 2B$, где B — максимальный по абсолютной величине коэффициент полиномов $p(x)$ и $p'(x)$: а именно, мы попытаемся вычислить $\gcd[p^{(p)}(x), p'^{(p)}(x)]$ для $p = 5, 7, 11$. [Напомним, что $p^{(p)}(x) = p(x) \pmod p$.]

$p = 5$. В этом случае $p^{(5)}(x) \equiv 2x^4+3x^3+4x^2+2x+4 \pmod 5$,

$p'^{(5)}(x) \equiv 3x^3+4x^2+3x+2 \pmod 5$ и $\gcd[p^{(5)}(x), p'^{(5)}(x)] \equiv 4x+4 \pmod 5$.

$p = 7$. Это число *не* разрешается, поскольку $7|112$.

$p = 11$. В этом случае $p^{(11)}(x) \equiv 2x^4 + 3x^3 + 2x^2 + 8x \pmod{11}$, $p'^{(11)}(x) \equiv 8x^3 + 9x^2 + 4x + 8 \pmod{11}$ и $\gcd[p^{(11)}(x), p'^{(11)}(x)] \equiv 2 \pmod{11}$.

Поскольку степень последнего gcd меньше степени полинома $\gcd[p^{(5)}(x), p'^{(5)}(x)]$, мы приходим к выводу, что 5 — несчастливое простое число; кроме того, мы заключаем, что нам не нужно испытывать еще какие-либо простые числа, поскольку $\gcd[p(x), p'(x)] = 1$ и $p(x)$ свободен от квадратов над целыми числами. (Это также легко проверить, используя **PSQFF**.)

Убедившись, что полином $p(x)$ свободен от квадратов, мы должны затем выбрать соответствующее простое число p для того, чтобы применить алгоритм Берлекэмпа. Простое число p должно быть таким, чтобы полином $p(x) \pmod p$ оставался свободным от квадратов и имел ту же степень, что и $p(x)$. Сделаем следующие замечания.

$p = 2$. Это простое число использовать нельзя, поскольку $2|112$ и $\deg[p^{(2)}(x)] < 4$.

$p = 3$. Это простое число использовать нельзя, поскольку полином $p^{(3)}(x)$ не свободен от квадратов, т.е. $p^{(3)}(x) = x^4 + x63 + 2x^2 + 2x = x(x+2)(x+1)^2$.

$p = 5$. Это простое число использовать нельзя, поскольку полином $p^{(5)}(x)$ не свободен от квадратов, т.е. $p^{(5)}(x) = 2x^4 + 3x^3 + 4x^2 + 2x + 4 = (x+4)(x+1)^2(2x+1)$.

$p = 7$. Это простое число использовать нельзя, поскольку $7|112$ и $\deg[p^{(7)}(x)] < 4$.

$p = 11$. Это простое число *можно* использовать для наших вычислений, поскольку полином $p^{(11)}(x)$ свободен от квадратов над $GF(11)$ (проверить) и разлагается на три сомножителя, а именно $p^{(11)}(x) = x(x+2)(2x^2+10x+4)$.

$p = 13$. Это простое число *можно* использовать для наших вычислений, поскольку полином $p^{(13)}(x)$ свободен от квадратов над $GF(13)$ (проверить) и разлагается на два сомножителя, а именно $p^{(13)}(x) = (8x^2+12x+10)(x^2+9x+9)$.

Как мы уже видели, нам следует работать с разложением по модулю 13, поскольку оно дает наименьшее количество сомножителей и их подъем потребует меньшей работы.

Ниже мы представим второй пример подъема разложения по mod p до разложения по mod p^j для некоторого j, пользуясь линейной конструкцией Гензеля; теперь используется симметрическая система вычетов (Yun, 1973).

Пример [линейная конструкция Гензеля; $p(x)$ — нормированный полином]. Рассмотрим полином $p(x) = x^5 + 12x^4 - 22x^3 - 163x^2 + 309x - 119$ в $\mathbb{Z}[x]$. При $p = 5$ мы имеем

$$p(x) \equiv g_1(x)h_1(x) \pmod 5,$$

где $g_1(x) = x3 + 2$ и $h_1(x) = x^2 + 2x - 2$. Мы собираемся выполнить две итерации конструкции Гензеля, т.е. мы найдем $g_3(x)$, $h_3(x)$, такие, что $p(x) \equiv g_3(x)h_3(x) \pmod{5^3}$; в действительности в этом случае $p(x) = g_3(x)h_3(x)$ в $\mathbb{Z}[x]$. Как и прежде, для того чтобы лучше понять, как получаются коэффициенты полиномов $g_3(x), h_3(x)$, мы приоткроем ответ, а именно

$$g_3(x) = x^3 - 15x + 17, \quad h_3(x) = x^2 + 12x - 7.$$

Пользуясь алгоритмом 6.3.1, получаем полиномы $a(x) = -x + 1$ и $b(x) = x^2 + 2x - 2$ в $GF(5)[x]$, такие, что $g_1(x)a(x) + h_1(x)b(x) = 1$ в $GF(5)$. Затем из соотношения $p(x) - g_1(x)h_1(x) = 5 \cdot c(x)$ (где $g_1(x), h_1(x)$ рассматриваются в кольце $\mathbb{Z}[x]$) получаем

$$p(x) - g_1(x)h_1(x) = 10x^4 - 20x^3 - 165x^2 + 305x - 115,$$

откуда

$$c(x) = \frac{p(x) - g_1(x)h_1(x)}{5} = 2x^4 + x^3 + 2x^2 + x + 2 \text{ в } GF(5).$$

Заметим, что $p \cdot c(x)$ — разность между $p(x)$ и $g_1(x)h_1(x)$ в $\mathbb{Z}_{p^2}[x]$, и нам нужно подкорректировать сомножители, т.е. прибавить к $g_1(x)$ и $h_1(x)$ то, чего им не хватает для того, чтобы разность исчезла. Пользуясь алгоритмом 6.3.2, получаем полиномы

$$a'(x) = 2x - 1, \quad b'(x) = 2x - 2,$$

такие, что $g_1(x)a'(x) + h_1(x)b'(x) = c(x)$ над $GF(5)$, а затем определим

$$g_2(x) = g_1(x) + p \cdot b'(x) = x^3 + 10x - 8,$$
$$h_2(x) = h_1(x) + p \cdot a'(x) = x^2 + 12x - 7.$$

Теперь легко проверить, что $p(x) \equiv g_2(x)h_2(x) \pmod{p^2}$; кроме того, заметим, что $h_2(x) \equiv h_1(x)$ и $g_2(x) \equiv g_1(x) \pmod{p}$ и в дополнение к этому (поскольку мы знаем окончательный ответ) $h_2(x) \equiv h_3(x)$ и $g_2(x) \equiv g_3(x) \pmod{p^2}$ [в действительности мы уже имеем $h_2(x) = h_3(x)$].

Мы имеем теперь для второй итерации новый полином

$$c(x) = \frac{p(x) - g_2(x)h_2(x)}{p^2} \pmod{p} = -(x^3 + x^2 + x + 2),$$

и, как это было раньше, нам нужно найти новые полиномы $a'(x)$ и $b'(x)$ в $GF(p)[x]$, такие, что $g_2(x)a'(x) + h_2(x)b'(x) = c(x)$ над $GF(5)$. Пользуясь алгоритмом 6.3.2, получаем $a'(x) = 0$ и $b'(x) = -x + 1$ в $GF(p)[x]$, а затем определяем

$$g_3(x) = g_2(x) + p^2 \cdot b'(x) = x^3 - 15x + 17,$$
$$h_3(x) = h_2(x) = x^2 + 12x - 7,$$

а это и есть искомые полиномы.

Мы видим, что если обозначить последовательные пары полиномов $\{a'(x), b'(x)\}$ через $\{a'_j(x), b'_j(x)\}$, то имеют место следующие соотношения:

$$g_3(x) = g_1(x) + p \cdot b'_1(x) + p^2 \cdot b'_2(x),$$
$$h_3(x) = h_1(x) + p \cdot a'_1(x) + p^2 \cdot a'_2(x),$$

очевидно, означающие, что $g_3(x) \equiv g_1(x)$ и $h_3(x) \equiv h_1(x) \pmod{p}$.

Перейдем теперь ко второму алгоритму подъема, предложенному Цассенхаузом в 1969 г.; это так называемый *квадратичный алгоритм подъема*, поскольку он расширяет разложение «за один шаг» от $\mathbb{Z}_{p^j}$ до $\mathbb{Z}_{p^{2j}}$, т.е. разложение последовательно поднимается $\bmod p^2, p^4, p^8$ и т.д. В приводимом ниже доказательстве мы придерживаемся соглашения, что нижние индексы $j, 2j$ обозначают полином в $\mathbb{Z}[x]$, коэффициенты которого однозначно определены только $\bmod p^j$, $\bmod p^{2j}$ соответственно.

Лемма 6.3.5 (Zassenhaus, 1969). Пусть $p(x)$ — полином в $\mathbb{Z}_{p^{2j}}[x]$, такой, что $\mathrm{lc}[p(x)] \neq 0$ в $GF(p)$, и пусть $p(x) \equiv g_j(x)h_j(x) \pmod{p^j}$, где для $g_j(x)$ и $h_j(x)$ в $\mathbb{Z}_{p^j}[x]$ существуют полиномы $a_j(x), b_j(x)$, такие, что $g_j(x)a_j(x) + h_j(x)b_j(x) = 1$ над $\mathbb{Z}_{p^j}$. Тогда мы можем определить полиномы $a_{2j}(x), b_{2j}(x), g_{2j}(x), h_{2j}(x)$ в $\mathbb{Z}_{p^{2j}}[x]$, такие, что $p(x) \equiv g_{2j}(x)h_{2j}(x) \pmod{p^{2j}}$, $g_{2j}(x)a_{2j}(x) + h_{2j}(x)b_{2j}(x) = 1$ над $\mathbb{Z}_{p^{2j}}$ и $g_{2j}(x) \equiv g_j(x) \pmod{p^j}$, $h_{2j}(x) \equiv h_j(x) \pmod{p^j}$.

Доказательство. Пусть полиномы $c_j(x)$ удовлетворяют соотношениям $p(x) - g_j(x)h_j(x) = p^j \cdot c_j(x)$ над $\mathbb{Z}_{p^{2j}}$. Пользуясь алгоритмом 6.3.2, вычислим $a'_j(x), b'_j(x)$ в $\mathbb{Z}_{p^j}[x]$, такие, что $g_j(x)a'_j(x) + h_j(x)b'_j(x) = c_j(x)$ над $\mathbb{Z}_{p^j}$. Положим $g_{2j}(x) = g_j(x) + p^j \cdot b'_j(x)$ и $h_{2j}(x) = h_j(x) + p^j \cdot a'_j(x)$ над $\mathbb{Z}_{p^{2j}}$. Проверка того, что полиномы $g_{2j}(x), h_{2j}(x)$ удовлетворяют выписанному выше условию, теперь тривиальна.

Для доказательства второго утверждения леммы предположим, что $c'_j(x)$ — такой полином в $\mathbb{Z}_{p^j}[x]$, что $g_{2j}(x)a_j(x) + h_{2j}(x)b_j(x) = 1 + p^j \cdot c'_j(x)$ над $\mathbb{Z}_{p^{2j}}$. Найдем $a''_j(x), b''_j(x)$ в $\mathbb{Z}_{p^j}[x]$, такие, что $g_j(x)a''_j(x) + h_j(x)b''_j(x) = c'_j(x)$ над $\mathbb{Z}_{p^j}$, а затем положим $a_{2j}(x) = a_j(x) - p^j \cdot a''_j(x)$ и $b_{2j}(x) = b_j(x) - p^j \cdot b''_j(x)$ над $\mathbb{Z}_{p^{2j}}$. Тогда

$$\begin{aligned} g_{2j}(x)a_{2j}(x) + h_{2j}(x)b_{2j}(x) =& g_{2j}(x)[a_j(x) - p^j \cdot a''_j(x)] \\ &+ h_{2j}(x)[b_j(x) - p^j \cdot b''_j(x)] \\ =& a_j(x)g_{2j}(x) + b_j(x)h_{2j}(x) \\ &- p^j \cdot [g_{2j}(x)a''_j(x) + h_{2j}(x)b''_j(x)] \\ =& 1 \text{ над } \mathbb{Z}_{p^{2j}}. \end{aligned}$$

□

Детали реализации алгоритма квадратичного подъема могут быть найдены в литературе. Продемонстрировано, что квадратичный алгоритм подъема существенно быстрее линейного [см. работы (Lenstra, 1982b, p. 184–185; Miola et al., 1974; Yun, 1974; Zassenhaus, 1978)].

6.3.2. Нахождение истинных сомножителей над целыми числами

Чтобы разложить полином $p(x)$ из $\mathbb{Z}[x]$ на множители, мы должны сделать следующее:

1. Выбрать подходящее простое число p и разлагать $p(x)$ над $GF(p)$.

2. Выбрать минимальное значение k, такое, что

$$p^k \geq 2 \cdot |\operatorname{lc}[p(x)]| \cdot \binom{\lfloor \frac{n}{2} \rfloor}{\lfloor \frac{n}{4} \rfloor} \cdot |p(x)|_2,$$

и поднять разложение $p(x)$ над $GF(p)$ до разложения над $\mathbb{Z}_{p^k}$, $p(x) \equiv \operatorname{lc}[p(x)] \cdot \prod_{1 \leq i \leq r} h_i(x) \pmod{p^k}$. Значение k, определяемое этим соотношением, получено из (M2), принимая во внимание факт, что мы ищем делители полинома $\operatorname{lc}[p(x)] \cdot p(x)$ (в соответствии с механизмом

восстановления старшего коэффициента, как мы видели в рассуждениях разд. 6.3.1 после первого примера подъема) и что мы можем ограничиться делителями степени $\leq \lfloor n/2 \rfloor$. После получения разложения полинома $p(x) \pmod{p^k}$, работая с полиномами, которые не обязательно являются нормированными, мы должны сделать следующий шаг 3.

3. Вычислить полином $h(x) = \mathrm{lc}[p(x)] \cdot \prod_{i \in S} h_i(x)$ над $\mathbb{Z}_{p^k}$ для каждого подмножества S множества $\{1, \ldots, r\}$, такого, что $\deg[h(x) \leq \lfloor n/2 \rfloor$, и проверить, является ли $h(x)$ делителем полинома $\mathrm{lc}[p(x)] \cdot p(x)$. Если да, то примитивная часть полинома $h(x)$ является сомножителем полинома $p(x)$ над целыми числами.

В настоящее время в общем случае не существует способа узнать, какое подмножество неприводимых сомножителей по модулю p^k образует неприводимый сомножитель полинома $p(x)$ над целыми числами, и, таким образом, нам нужно проверить для всех возможных комбинаций степени $\leq \lfloor n/2 \rfloor$, дают ли они сомножитель полинома $p(x)$ над $\mathbb{Z}$. Если $p(x)$ неприводим над $\mathbb{Z}$, то ни одно из этих пробных делений не будет успешным, и, как мы уже упоминали, тест экспоненциально зависит от числа r сомножителей по модулю p^k. Однако можно значительно уменьшить число пробных делений в $\mathbb{Z}[x]$, если (1) сначала проверить делимость свободных членов, а затем (2) разложить $p(x)$ по модулю нескольких простых чисел. На шагах 2 и 3 в приведенном выше алгоритме надо использовать простое число с наименьшим числом сомножителей и сочетать информацию о степенях сомножителей по модулю различных простых чисел, чтобы уменьшить разноообразие возможностей для степеней сомножителей над $\mathbb{Z}$. Как мы уже видели, если $p(x)$ разлагается по модулю p_1 на 3 сомножителя степени 2 и если $p(x)$ разлагается по модулю p_2 на 2 сомножителя степени 3, то полином $p(x)$ неприводим над целыми числами.

Наконец, существуют два очевидных способа реализации шага 3. Первый — это *процедура мощности*, т.е. перебираем сочетания по s сомножителей для $s = 1, 2, \ldots, \lfloor r/2 \rfloor$. Второй подход — это *процедура степени*, т.е. перебираем сочетания общей степени s для $s = 1, 2, \ldots, \lfloor r/2 \rfloor$. Обнаружено, что процедура мощности предпочтительнее.

Упражнения

Раздел 6.1.1

1. Пользуясь методом Шуберта–Кронекера, разложите на множители следующие полиномы:
   - **a.** $x^4+x^3+x^2-x-2$ (*указание*: сомножители суть x^2+x+2, $x-1$, $x+1$);
   - **b.** x^4-x^2+1;
   - **c.** x^6+1;
   - **d.** $x^4+x^3+x^2+x+1$.

Раздел 6.2.1

1. Рассмотрим полином $p(x)=112x^4+58x^3-31x^2+107x-66$ в $\mathbb{Z}[x]$. Свободен ли он от квадратов в $\mathbb{Z}_p[x]$, $p=3,5,11,13$? (*Указание.* Для ответа на эту часть упражнения посмотрите шаг 2 разд. 6.1.2, т.е. выполните вычисление gcd.) Используйте алгоритм **PSQFFF** для нахождения свободных от квадратов сомножителей полинома $p(x)$ в $\mathbb{Z}_p[x]$, $p=3,5,11,13$.
2. Рассмотрим полином $p(x)=x^5+x+1$ в $\mathbb{Z}[x]$. Свободен ли он от квадратов в $\mathbb{Z}_p[x]$, $p=3,5,7,11,13$? Используйте алгоритм **PSQFFF** для нахождения свободных от квадратов сомножителей полинома $p(x)$ в $\mathbb{Z}_p[x]$, $p=3,5,7,11,13$.

Раздел 6.2.2

1. Докажиту теорему 6.2.7 в обратном направлении, т.е. докажите, что если функция F определена на натуральных числах, а функция f определена равенством $f(n)=\sum_{d|n}\mu(d)F(n/d)$, то $F(n)=\sum_{d|n}f(d)$.
2. Вычислите $i_3(4)$ и найдите все неприводимые полиномы степени 4 в $\mathbb{Z}_3[x]$.

Раздел 6.2.3

1. Докажите лемму 6.2.11.
2. Используя методы, приведенные в этом разделе, разложите на множители полиномы
   - **a.** x^9-x в $\mathbb{Z}_3[x]$;
   - **b.** $x^{25}-x$ в $\mathbb{Z}_5[x]$;

c. $x^5 + x^4 + 1$ в $\mathbb{Z}_2[x]$;
d. $x^6 + x^5 + x^3 + x^2 + x + 1$ в $\mathbb{Z} - 2[x]$.

Раздел 6.2.4

1. Покажите, что если $b_1(x)$ и $b_2(x)$ — взаимно простые полиномы в кольце $\mathbb{Z}_p[x]$ (или над произвольным полем) и $p(x)$ принадлежит $\mathbb{Z}_p[x]$, то

$$\gcd[p(x), b_1(x)b_2(x)] = \gcd[p(x), b_1(x)] \gcd[p(x), b_2(x)].$$

2. Докажите, что если $\mathbf{V}$ — векторное пространство размерности r над полем $GF(p)$, то оно содержит p^r элементов.
3. Пользуясь алгоритмом Берлекэмпа, разложите на множители полиномы
a. $x^5 + x^4 + 1$ в $\mathbb{Z}_2[x]$;
b. $x^{12} + x^9 + x^6 + x^3 + 1$ в $\mathbb{Z}_2[x]$;
c. $x^8 - x^7 + x^5 - x^4 + x^3 - x + 1$ в $\mathbb{Z}_2[x]$;
d. $x^9 - x$ в $\mathbb{Z}_3[x]$;
e. $x^{25} - x$ в $\mathbb{Z}_5[x]$;
f. $2x^4 + 3x^3 + 2x^2 + 8$ в $\mathbb{Z}_{11}[x]$.

Раздел 6.3

1. Найдите границы для коэффициентов сомножителей в $\mathbb{Z}[x]$ полиномов
a. $x^5 + x^4 + 1$;
b. $x^6 + x^5 + x^3 + x^2 + x + 1$;
c. $x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$;
d. $x^9 - x$;
e. $x^{25} - x$;
f. $2x^4 + 3x^3 + 2x^2 + 8$.

Раздел 6.3.1

1. Рассмотрим полиномы
a. $x^5 + x^4 + 1$;
b. $x^6 + x^5 + x^3 + x^2 + x + 1$;
c. $x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$;
d. $x^9 - x$;

e. $x^{25} - x$;

f. $2x^4 + 3x^3 + 2x^2 + 8$.

В упражнениях разд. 6.2.4 мы просили разложить некоторые из этих полиномов на множители в $\mathbb{Z}_p[x]$ для выделенных значений p; кроме того, в упражнениях разд. 6.3 мы просили вычислить соответствующие границы для коэффициентов их сомножителей в $\mathbb{Z}[x]$. Возьмите полином по своему выбору и выполните линейный подъем его разложения по модулю p до разложения над целыми числами.

2. Какова функция времени вычисления для алгоритма **HLLA**?

3. Опишите основные шаги алгоритма **HQLA** (**H**ensel's **Q**uadratic-**L**ifting **A**lgorithm).

В следующих упражнениях мы применяем понятие подъема к решению полиномиального сравнения $f(x) \equiv 0 \pmod{p^k}$.

4. Докажите, что произведение любых k последовательных целых чисел делится на $k!$ (*Указание.* Воспользуйтесь тем, что, согласно формуле бинома, коэффициент при $x^k y^{n-k}$ в разложении бинома $(x+y)^n$ является целым числом $n!/k!(n-k)!$.)

5. С помощью упр. 3 докажите, что если $x = a$ — решение сравнения $f(x) \equiv 0 \pmod{p^j}$, то решение по модулю p^{j+1} получается из линейного сравнения

$$tf'(a) \equiv -f(a)/p^j \pmod{p},$$

где $f'(x)$ — первая производная функции $f(x)$. Заметим, что это линейное сравнение может не иметь решений, иметь одно решение или p решений. (*Указание.* Для данного a решение по модулю p^{j+1} будет иметь вид $a + tp^j$, где t определяется с использованием разложения Тейлора.)

Упр. 5 используется в следующих упражнениях.

6. Решите сравнение $x^2 + x + 7 \equiv 0 \pmod{3^3}$.

7. Решите сравнение $x^3 + x + 57 \equiv 0 \pmod{5^3}$.

8. Решите сравнение $x^2 + x + 47 \equiv 0 \pmod{7^3}$.

9. (Подъем квадратного корня.) Пусть $m = (p_1)^{e_1}(p_2)^{e_2}\ldots(p_r)^{e_r}$ — нечетное целое число, и предположим, что $\gcd(a, m) = 1$ и a — квадрат некоторого целого числа по модулю m. Мы хотим найти x, такой, что $x^2 \equiv a \pmod{m}$, в предположении, что для каждого i нам известен невычет по модулю p_i, $i = 1, 2, \ldots, r$ (см. также упр. 22 и 23 разд. 3.3.1).

a. Для каждых фиксированных $p := p_i$ и $e := e_i$ воспользуйтесь алгоритмом упр. 23 к разд. 3.3.1, чтобы определить некоторое x_0, такое, что $(x_0)^2 \equiv a \pmod{p}$. Вычислив такое x_0, найдите некоторое $x = x_0 + x_1 p + x_2 p^2 + \cdots + x_{e-1} p^{e-1}$, такое, что $x^2 \equiv a \pmod{p^e}$.

b. Опишите, как найти x, такой, что $x^2 \equiv a \pmod{m}$.

Раздел 6.3.2

1. Можно ли разложить на множители полином $p(x) = x^3 - 7x + 7$ над целыми числами?

2. Разложите на множители полиномы $p(x) = x^5 + x^4 + 1$ и $p(x) = x^5 + x + 1$ над целыми числами.

Упражнения по программированию

Раздел 6.3.1

1. Разработайте процедуру, которая будет решать сравнения по модулю некоторого целого числа, не обязательно простого. (Воспользуйтесь системой компьютерной алгебры и пакетом для разложения полиномов над конечными полями.)

Исторические замечания и литература

1. Существует также (представленная ниже) теорема *Штикельбергера*, позволяющая нам определить, четное или нечетное число неприводимых сомножителей имеет данный полином в $\mathbb{Z}_p[x]$, если p — нечетное простое число.

Теорема Штикельбергера. Пусть p — нечетное простое число, $p(x)$ — нормированный свободный от квадратов {так что его дискриминант $\operatorname{discr}[p(x)]$ отличен от 0} полином степени n в $\mathbb{Z}_p[x]$, и пусть r — число неприводимых сомножителей полинома $p(x)$ в $\mathbb{Z}_p[x]$. Тогда $r \equiv n \pmod{2}$ в том и только том случае, когда дискриминант $\operatorname{discr}[p(x)]$ является квадратом в $\mathbb{Z}_p$.

[Доказательство ее можно найти в (Childs, 1979).]

2. Мак-Элис (McEliece, 1969) описывает простой для программирования алгоритм определения множества полиномов $\{B_1(X), \ldots, b_r(x)\}$, которые разделяют сомножители полинома $p(x)$; это множество полиномов может быть вычислено без матричной диагонализации, и каждый полином этого множества имеет вид $b(x) = x^i + x^{ip} + x^{ip^2} + \cdots + x^{ip^s} \pmod{p(x)}$ для некоторых значений i и s, так что $[b(x)]^p \equiv b(x) \pmod{p(x)}$. Недостаток этого метода состоит в том, что время его работы зависит от степеней сомножителей полинома $p(x)$, в то время как алгоритм Берлекэмпа зависит от n, p и числа сомножителей.

3. Следует заметить, что теорема 6.3.2 соответствует теореме 1 в статье Миньотта 1974 г. В действительности эта теорема впервые упоминалась Ландау в 1905 г., а затем Шпехтом в 1949 г. В сочетании с хорошо известной теоремой 6.3.1 получается граница для коэффициентов сомножителей полинома.

Панкратьев Е.В. Компьютерная алгебра. Факторизация многочленов. — М.: Изд-во МГУ, 1988.

Bender E.A., Goldman J.R. On the application of Moebius inversion in combinatorial analysis. *American Mathematical Monthly* **82**, 789–802, 1975.

Berlekamp E.R. Factoring polynomials over finite fields. *The Bell System Technical Journal* **46**, 1853–1859, 1967.

Berlekamp E.R. *Algebraic coding theory.* McGraw-Hill, New York, 1968. [Имеется перевод: Берлекэмп Э. Алгебраическая теория кодирования. — М.: Мир, 1974.]

Berlekamp E.R. Factoring polynomials over large finite fields. *Mathematics of Computation* **24**, 713–735, 1970.

Butler M.C.R. On the reducibility of polynomials over a finite field. *The Quarterly Journal of Mathematics Oxford Second Series* **5**, 102–107, 1954.

Calmet J., Loos R. An improvement of Rabin's probabilistic algorithm for generating irreducible polynomials over $GF(p)$. *Information Processing Letters* **11**, 94–95, 1980.

Calmet J., Loos R. Deterministic versus probabilistic factorization of integral polynomials. In *Computer Algebra, EUROCAM 1982 Lecture Notes in Computer Science*, Springer-Verlag, New York, 1982, pp. 117–125.

Camion P. Un algorithm de construction des idempotents primitifs d'ideaux d'algebres sur F_q. *Comptes Rendus des Séances de l'Académie des Sciences* **291**, 479–482, 1980.

Cantor D.G., Zassenhaus H. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation* **36**, 587–592, 1981.

Cantor M. *Geschichte der Mathematik*, Vol. 4. Teubner Verlag, Leipzig, 1908.

Childs L. *A concrete introduction to higher algebra.* Springer Verlag, New York, 1979.

Gunji H., Arnon D. On polynomial factorization over finite fields. *Mathematics of Computation* **36**, 281–287, 1981.

Hensel K. *Theorie der algebraischen Zahlen.* Teubner Verlag, Leipzig, 1908.

Kaltofen E. Factorization of polynomials. *Computing (Suppl. 4)* 95–113, 1982.

Knuth D.E. *The art of computer programming*, Vol. 2, 2nd ed. *Seminumerical Algorithms.* Addison-Wesley, Reading, MA 1981. [Имеется перевод первого издания: Кнут Д. *Искусство программирования для ЭВМ.* Т. 2. — М.: Мир, 1977.]

Landau E. Sur quelques theoremes de M. Petrovitch relatifs aux zeros des fonctions analytiques. *Bulletin de la Societe Mathematique de France* **33**, 251–261, 1905.

Lauer M. Computing by homomorphic images. *Computing (Suppl. 4)*, 139–168, 1982.

Lazard D. On polynomial factorization. In *Computer Algebra, EUROCAM 1982 Lecture Notes in Computer Science*, Springer-Verlag, New York, 1982, pp. 126–134.

Lenstra A.K. *Lattices and factorization of polynomials.* Technical Report 190/81, Department of Computer Science, Mathematisch Centrum, Amsterdam 1981.

Lenstra A.K. *Factoring polynomials over algebraic number fields.* Technical Report 213/82, Department of Computer Science, Mathematisch Centrum, Amsterdam, 1982a.

Lenstra A.K. Factorization of polynomials. In *Computational methods in number theory.* H.W. Lenstra, Jr and R. Tijdeman, eds. *Mathematical Centre Tracts*, Vol. 154, Amsterdam, 1982b, pp. 169–198.

Lenstra A.K., Lenstra H.W. Jr., Lovasz L. Factoring polynomials with rational coefficients. *Mathematische Annalen* **261**, 515–534, 1982.

Lidl R., Pilz G. *Applied abstract algebra.* Springer-Verlag, New York, 1984.

McEliece R.J. Factorization of polynomials over finite fields. *Mathematics of Computation* **23** 861–867, 1969.

Mignotte M. An inequality about factors of polynomials. *Mathematics of Computation* **28**, 1153–1157, 1974.

Miola A., Yun D.Y.Y. Computational aspects of Hensel-type univariate polynomial greatest common divisor algorithms. *Proceedings of EUROSAM '74*, 46–54, 1974.

Moenck R.T. On the efficiency of algorithms for polynomial factoring. *Mathematics of Computation* **31**, 235–250, 1977.

Musser D.R. *Algorithms for polynomial factorization.* Ph.D. Thesis, University of Wisconsin-Madison, Department of Computer Science, 1971.

Musser D.R. Multivariate polynomial factorization. *Journal of the Association for Computing Machinery* **22**, 291–308, 1975.

Musser D.R. On the efficiency of a polynomial irreducibility test. *Journal of the Association for Computing Machinery* **25**, 271–282, 1978.

Petr K. Ueber die Reduzibilitaet eines Polynoms mit ganzzahligen Koeffizienten nach einem Primzahlmodul. *Casopis pro Pestovani Matematiky a Fysiky* **66**, 85–94, 1936.

Rabin M.O. Probabilistic algorithms in finite fields. *SIAM Journal of Computing* **9**, 273–280, 1980.

Schwarz S. Sur le nombre des racines et des facteurs irréductibles d'une congruence donné. *Casopis pro Pestovani Matematiky a Fysiky* **69**, 128–145, 1939.

Schwarz S. On the reducibility of polynomials over a finite field. *The Quarterly Journal of Mathematics Oxford, Second Series* **7**, 110–124, 1956.

Sims C.C. *Algebra, a computational approach.* Wiley, New York, 1984.

Simmons G. On the number of irreducible polynomials of degree d over $GF(p)$. *American Mathematical Monthly* **77**, 743–745, 1970.

Smith S. On the value of a certain arithmetical determinant. *Proceedings of the London Mathematical Society* **7**, 208–212, 1876.

Specht W. Abschaetzungen der Wurzeln algebraischer Gleichungen. *Mathematische Zeitschrift* **52**, 310–321, 1949.

Swinnerton-Dyer H.P.F. Cited in Berlekamp's article «Factoring polynomials over large finite fields». Mathematics of Computation **24**, 733–734, 1970.

Wang P.S. An improved multivariate polynomial factoring algorithm. *Mathematics of Computation* **32**, 1215–1231, 1978.

Yun D.Y.Y. *The Hensel lemma in algebraic manipulation.* Ph.D. Thesis, Department of Mathematics, M.I.T., 1973.

Zassenhaus H. On Hensel factorization. *Journal of Number Theory* **1**, 291–311, 1969.

Zassenhaus H. A remark on the Hensel factorization method. *Mathematics of Computation* **32**, 287–292, 1978.

Отделение и аппроксимация вещественных корней полиномиальных уравнений

Методы нахождения вещественных корней, или решений, полиномиальных уравнений в $\mathbb{Z}[x]$ любой степени можно найти в любой работе по численному анализу и теории уравнений. Наш подход к рассматриваемому вопросу принципиально отличается в трех главных пунктах:

1. Мы явно различаем отделение и аппроксимацию корней (разд. 7.1), в то время как в большинстве работ аппроксимация корней эквивалентна решению уравнений.

2. Мы даем точные формулировки теорем Бюдана и Фурье и рассматриваем их взаимозависимость (см. также рис. 7.1.1); следует отметить, что формулировку теоремы Бюдана нелегко найти в литературе, в то время как теорема Фурье часто приписывается Бюдану.

3. Мы представляем теорему Винсента 1836 г. и получающиеся на ее основе два метода цепных дробей для отделения вещественных корней уравнения; первый из этих методов принадлежит Винсенту (Vincent, 1836) и имеет экспоненциальное время вычислений, в то время как второй принадлежит автору (Akritas, 1978) и имеет, если справедлива некоторая гипотеза, полиномиальное время счета (лучший из классических методов). Теорема Винсента была так хорошо забыта, что даже такая фундаментальная работа, как *Enzyclopaedie der mathematischen Wissenschaften*, игнорирует ее.

Исследуем прежде всего детально метод бисекций Штурма 1829 г. и метод цепных дробей 1978 г., являющийся самым быстрым из существующих методов. Это — два классических метода, описанные в литературе, для отделения вещественных корней полиномиального уравнения с целыми коэффициентами (см. историческое замечание 1 и рис. 7.1.1). Затем исследуем два соответствующих метода аппроксимации этих (изолированных) корней с любой желаемой степенью точности, а именно, метод бисекций и метод, использующий цепные дроби.

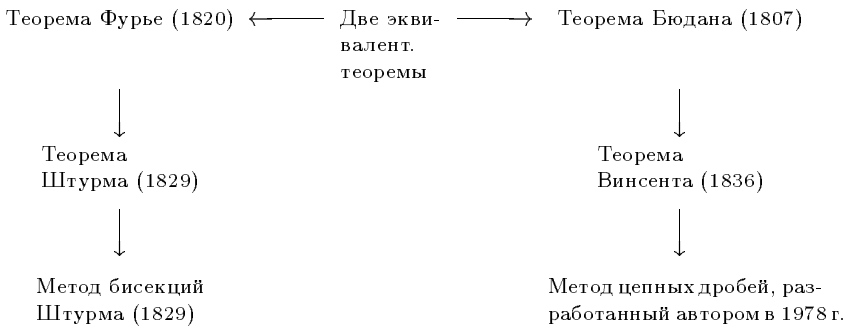


Рис. 7.1.1.

Краткий обзор исторического развития двух классических методов отделения вещественных корней полиномиального уравнения с целыми коэффициентами. Метод цепных дробей Винсента 1836 г. не показан, поскольку он экспоненциален (см. также историческое замечание 7).

7.1

Введение и обоснования

Самые ранние из известных аналогов алгебраических уравнений содержатся в папирусе Ринда и, очевидно, компилированы из более ранних работ египтянином Амесом приблизительно в 1650 или 1700 г. до нашей эры. Мы находим, например, следующую задачу: «Количество и его седьмая часть, сложенные вместе, дают 19. Чему равно количество?». Очевидно, задача состоит в том, чтобы решить уравнение $x + (1/7)x = 19$, как мы сказали бы сегодня. Из-за недостатка удобных алгебраических обозначений египтяне пользовались громоздким методом, известным впоследствии как метод «ложного положения». Хотя иногда утверждают, что греки умели решать уравнения второй степени, в общем ни египтяне, ни греки не сделали сколько-нибудь заметного продвижения с современной точки зрения. Арабы достигли большего, но только во времена Ренессанса, когда итальянские математики пятнадцатого и шестнадцатого столетий (Тарталья, Кардано и Феррари) преуспели в решении через радикалы общих уравнений третьей и четвертой степени, были выполнены работы, представляющие длительный интерес. [Интересующийся историей читатель может проследить по другим источникам за борьбой за приоритет между Тартальей и Кардано (см., например, книгу (Burnside, Panton, 1960,

p. 271–274). Другие источники — (Dickson, 1922; Todhunter, 1882; Turnbull, 1957; Weisner, 1938).)]

В семнадцатом и восемнадцатом столетиях предпринимались многочисленные попытки решить общее уравнение пятой степени; было также достигнуто более глубокое понимание природы корней алгебраического уравнения (особенно после работы Декарта), но, несмотря на все это, никому не удалось решить в радикалах уравнение пятой степени. Естественно, что математики задались вопросом, возможно ли вообще такое решение. Ответ дал Руффини (Ruffini, 1804), первым показавший невозможность алгебраического решения уравнения пятой степени. Позднее Абель (Abel, 1826) доказал, что невозможно в общем случае решать алгебраические уравнения степени выше четвертой.

В начале девятнадцатого века внимание математиков уже сосредоточилось на численных методах решения общих полиномиальных уравнений с целыми коэффициентами. В этот период у Фурье возникла идея разделить процесс на *два* шага, т.е. сначала отделить вещественные корни, а затем аппроксимировать их с любой желаемой степенью точности. (В соответствии с хорошо известным в математике подходом: если проблема слишком трудна, чтобы решить ее целиком, расщепляют ее на более простые.)

Отделение вещественных корней полиномиального уравнения — это процесс нахождения вещественных непересекающихся интервалов, таких, что каждый из них содержит точно один вещественный корень и каждый вещественный корень содержится ровно в одном интервале. *Аппроксимация*, с другой стороны, — это процесс, когда изолирующие интервалы делаются как угодно малыми, приближая, таким образом, корни с требуемой степенью точности. Главной проблемой, привлекавшей внимание математиков, было отделение.

Как только прояснились цели, последовали и успехи. В начале девятнадцатого столетия Бюдан и Фурье представили две различные (но эквивалентные) теоремы, позволяющие определять *максимум* возможного числа вещественных корней уравнения с вещественными коэффициентами на данным интервале. Теорема Бюдана появилась в 1807 г. в работе «Nouvelle méthode pour la résolution des équations numériques», а теорема Фурье была впервые опубликована в 1820 г. в «Le bulletin des sciences par la société philomatique de Paris». В силу значимости этих двух теорем возникли большие разногласия относительно приоритетных прав. В своей книге *Biographies of distinguished scientific men*, с. 383, Араго (Arago, 1859) сообщает, что Фурье «счел необходимым получить подтверждение бывших студентов Политех-

нической школы или профессоров университета», чтобы доказать, что он преподавал свою теорему в 1796, 1797 и 1803 гг.

Как мы увидим в разд. 7.2, основываясь на предложении Фурье, Штурм представил в 1829 г. улучшенную теорему, применение которой дает *точное* число вещественных корней полиномиального уравнения из $\mathbb{Z}[x]$ без кратных нулей на вещественном интервале; таким образом, он решил проблему отделения вещественных корней (Bocher, 1911–1912). С 1830 г. метод Штурма становится единственным широко известным и используемым, и, следовательно, теорема Бюдана предается забвению. По нашим сведениям, теорему Бюдана можно найти только в статье Винсента и в работах автора, в то время как предложение Фурье появляется почти во всех текстах по теории уравнений. Теорема Бюдана заслуживает специального внимания, поскольку, как мы увидим в разд. 7.3, она составляет основу теоремы Винсента 1836 г. которая, в свою очередь, составляет основу метода цепных дробей 1978 г. (Akritas, 1980a, 1980b) отделения вещественных корней уравнения, метода, превосходящего метод Штурма по эффективности. [См. также рис. 7.1.1 и (Akritas, 1982).]

7.2

Теорема Фурье и метод бисекций Штурма отделения вещественных корней

Этот раздел мы начинаем теоремой Фурье, которая дает нам возможность определять *максимум* возможного числа вещественных корней уравнения с целыми коэффициентами на данном интервале, а затем мы увидим, как Штурм модифицировал этот метод для получения *точного* числа вещественных корней на этом интервале.

7.2.1. Теорема Фурье

Теорема Фурье, опубликованная впервые в 1820 г., также была включена в книгу «*Analyse des Equations*», опубликованную впоследствии Навье в 1831 г. Эту теорему, иногда под названием теоремы Бюдана–Фурье или даже просто Бюдана, можно найти почти во всех работах по теории уравнений. [Гурвиц (Hurwitz, 1912) представляет ее как частный случай более общей теоремы, а Обрешкоф (Obreschkoff, 1963) обобщает ее на случай комплексных корней.] Мы приведем ее после того, как докажем следующие две леммы; хотя нас интересуют

полиномы с целыми коэффициентами, мы докажем эти леммы для общего случая вещественных коэффициентов.

Лемма 7.2.1. Пусть $p(x) = 0$ — полиномиальное уравнение степени $n > 0$ с вещественным коэффициентами, и пусть α — один из его вещественных корней (если такие существуют). Для достаточно малого $\varepsilon > 0$ полиномы $p(x)$ и $p'(x)$ [производная от полинома $p(x)$] имеют *противоположные* знаки в интервале $(\alpha - \varepsilon, \alpha)$ и *одинаковые* знаки в интервале $(\alpha, \alpha + \varepsilon)$.

Доказательство. Лемма непосредственно следует из рассмотрения формулы разложения Тейлора $p(a \pm x) = \sum_{0 \le i \le n} [p^{(i)}(a)/i!]x^i$, где $p^{(i)}(a)$ есть i-я производная. Предположим, что α имеет кратность $m \ge 1$; тогда для $p(\alpha \pm \varepsilon)$ и $p'(\alpha \pm \varepsilon)$ мы имеем

$$p(\alpha \pm \varepsilon) = \left[\frac{p^{(m)}(\alpha)}{m!}\right](\pm\varepsilon)^m + \ldots$$

и

$$p'(\alpha \pm \varepsilon) = \left[\frac{p^{(m)}(\alpha)}{(m-1)!}\right](\pm\varepsilon)^{m-1} + \ldots,$$

откуда видно, что знаки этих выражений [которые для достаточно малого ε зависят от знака члена $p^{(m)}(\alpha)$] противоположны на $(\alpha - \varepsilon, \alpha)$ и совпадают на $(\alpha, \alpha + \varepsilon)$. □

Применяя предыдущий результат к последовательным производным, получаем такой результат.

Лемма 7.2.2. Пусть $p(x) = 0$ — полиномиальное уравнение с вещественными коэффициентами степени $n > 0$, и пусть α — один из его вещественных корней кратности m. В последовательности m функций $p(x), p^{(1)}(x), \ldots, p^{(m-1)}(x)$ [где $p^{(i)}(x)$ есть i-я производная] для достаточно малого $\varepsilon > 0$ мы можем заменить x на $\alpha - \varepsilon$ так, чтобы знаки в получающейся числовой последовательности *чередовались*, и мы можем заменять x на $\alpha + \varepsilon$ так, чтобы все знаки в получающейся числовой последовательности *совпадали* со знаком $p^{(m)}(\alpha)$, первой функции, для которой α не является корнем.

Доказательство. Доказательство мы оставляем читателю в качестве упражнения. □

Определение 7.2.3. Мы говорим, что между двумя ненулевыми членами c_p и c_q $(p < q)$ конечной или бесконечной последовательности вещественных чисел $c_1, c_2, c_3, \ldots$ имеет место перемена знаков, если выполнено одно из следующих условий:

i. Для $q = p + 1$ c_p и c_q имеют противоположные знаки.

ii. Для $q \geq p + 2$ все числа $c_{p+1}, \ldots, c_{q-1}$ равны нулю, а c_p и c_q имеют противоположные знаки.

Пример. Рассмотрим полином $p(x) = x^3 - 7x + 7$, коэффициенты которого образуют конечную последовательность чисел $\{1, 0, -7, 7\}$. Ясно, что в этой последовательности имеются 2 перемены знаков.

Определение 7.2.4. Пусть $p(x) = 0$ — полиномиальное уравнение степени $n > 0$ с вещественными коэффициентами. Тогда последовательность $n + 1$ функций

$$\text{fseq}(x) = \{p(x), p^{(1)}(x), p^{(2)}(x), \ldots, p^{(n)}(x)\},$$

где $p^{(i)}(x)$ есть i-я производная, называется *последовательностью Фурье*.

Пример. Рассмотрим полиномиальное уравнение $p(x) = x^3 - 7x + 7 = 0$. Последовательность Фурье, соответствующая этому полиному, — это $\text{fseq}(x) = \{x^3 - 7x + 7, 3x^2 - 7, 6x, 6\}$.

Мы готовы теперь сформулировать и доказать теорему Фурье. Читатель должен тщательно изучить доказательство, потому что оно полезно для понимания.

Верхняя грань числа вещественных корней уравнения в открытом интервале.

Теорема 7.2.5 (Фурье 1820). Пусть $p(x) = 0$ — полиномиальное уравнение с вещественными коэффициентами степени $n > 0$. Если в последовательности Фурье $\text{fseq}(x) = \{p(x), p^{(1)}(x), p^{(2)}(x), \ldots, p^{(n)}(x)\}$ мы заменим x любыми двумя вещественными числами p, q $(p < q)$, то для двух получившихся числовых последовательностей $\text{fseq}(p)$ и $\text{fseq}(q)$ выполняются такие условия:

i. $\text{fseq}(p)$ не может иметь меньше перемен знаков, чем $\text{fseq}(q)$.

ii. Число вещественных корней уравнения $p(x) = 0$, размещенных между p и q, никогда не может быть больше, чем число перемен знаков, потерянных в $\text{fseq}(x)$ при переходе от подстановки $x := p$ к подстановке $x := q$.

iii. Если число вещественных корней уравнения $p(x) = 0$, размещенных между p и q, меньше числа перемен знаков, потерянных в $\mathrm{fseq}(x)$ при переходе от подстановки $x := p$ к подстановке $x := q$, то разность — четное число.

Доказательство. При изменении x число перемен знаков в последовательности Фурье может измениться *только* тогда, когда x — нуль полинома $p(x)$ или какой-то из его производных. Эти два случая мы будем исследовать раздельно.

Случай 1. Пусть α — нуль полинома $p(x)$ кратности m. Пользуясь леммой 7.2.2, мы видим, что при изменении x в интервале $(\alpha-\varepsilon, \alpha+\varepsilon)$ для достаточно малого $\varepsilon > 0$ мы имеем в первых $m+1$ членах его последовательности Фурье непосредственно перед переходом через α m перемен знаков и нуль перемен знаков немедленно после этого перехода. Знаки остальных членов последовательности $\mathrm{fseq}(x)$ не меняются. Таким образом, в $\mathrm{fseq}(x)$ теряются m перемен знаков.

Случай 2. Пусть теперь α — нуль кратности m одной из производных, т.е. для некоторого $i \neq 0$ и $i \neq n$ мы имеем $p^{(i-1)}(\alpha) \neq 0$ и $p^{(i)}(\alpha) = 0$. Рассмотрим последовательность

$$p^{(i-1)}(x), p^{(i)}(x), p^{(i+1)}(x), \ldots, p^{(i+m)}(x),$$

где $p^{(i+m)}(x)$ — первая функция, для которой α не является корнем. Заметим, что при изменении x в $(\alpha-\varepsilon, \alpha+\varepsilon)$ для достаточно малого $\varepsilon > 0$ знаки $p^{(i-1)}(x)$ и $p^{(i+m)}(x)$ остаются неизменными, поскольку эти полиномы не обращаются в нуль. Разбирая различные случаи, когда знаки $p^{(i-1)}(x)$ и $p^{(i+m)}(x)$ одинаковы или противоположны и кратность m четная или нечетная, мы убеждаемся, что общее число потерянных перемен знаков — четное число.

Таким образом, мы убедились, что при прохождении x данного интервала число потерянных перемен знаков должно либо быть равным числу корней полинома $p(x)$ в этом интервале, либо превосходить его на четное число. □

Пример. Рассмотрим полином $p(x) = x^3 - 7x + 7 = 0$, последовательностью Фурье которого, как мы уже видели, является $\mathrm{fseq}(x) = \{x^3 - 7x + 7, 3x^2 - 7, 6x, 6\}$. Тогда, чтобы найти верхнюю границу числа вещественных корней в интервале $(0, 2)$, мы вычисляем значения последовательности Фурье в точках 0 и 2 и получаем следующие числовые последовательности:

$$\mathrm{fseq}(0) = \{7, -7, 0, 6\}, \quad \mathrm{fseq}(2) = \{1, 5, 12, 6\},$$

где fseq(0) имеет 2 перемены знаков, а fseq(2) — ни одной. По теореме Фурье полином $p(x)$ или имеет два вещественных корня в интервале $(0, 2)$ или не имеет ни одного; это определяется дальнейшими исследованиями.

Очевидно, теорема Фурье дает нам верхнюю границу числа вещественных корней уравнения $p(x) = 0$ (с вещественными коэффициентами и степени n) внутри интервала (p, q); заметим, что непосредственной подстановкой мы можем легко определять, являются ли p и q корнями.

Теорема 7.2.5 может быть использована для простого доказательства правила знаков Кардано–Декарта.

Верхняя граница числа положительных корней уравнения.

Теорема 7.2.6 (Кардано–Декарт). Пусть $p(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0 = 0$ — полиномиальное уравнение с вещественными коэффициентами. Если v — число перемен знаков в последовательности коэффициентов $c_n, c_{n-1}, \ldots, c_0$ (нулевые коэффициенты просто опускаются) и p — число положительных корней уравнения $p(x) = 0$, то

$$v = p + 2\lambda,$$

где $\lambda \geq 0$ — целое число.

Доказательство. Мы ищем верхнюю границу числа корней уравнения $p(x) = 0$ в интервале $(0, \infty)$. Последовательность Фурье в этом случае имеет вид

$$\begin{aligned}
p(x) &= c_n x^n + \cdots + c_0, \\
p^{(1)}(x) &= n c_n x^{n-1} + \cdots + c_1, \\
p^{(2)}(x) &= n(n-1) c_n x^{n-2} + \cdots + (2!)c_2, \\
&\cdots \\
p^{(n)}(x) &= (n!)c_n.
\end{aligned}$$

Применяя теорему Фурье, при $x = 0$ мы получаем последовательность

$$\text{fseq}(0) = \{c_0, c_1, (2!)c_2, \ldots, (n!)c_n\},$$

которая имеет v перемен знаков, в то время как последовательность fseq(∞) не имеет ни одной перемены знаков (поскольку знаки всех ее членов совпадают со знаком c_n). Поэтому по теореме 7.2.5

$$v = p + 2\lambda,$$

где $\lambda \geq 0$ — целое число, а это и есть правило знаков Кардано–Декарта. □

Пример. Рассмотрим полиномиальное уравнение $p(x) = x^3 - 7x + 7 = 0$. Поскольку в последовательности его коэффициентов имеются две перемены знаков, из правила знаков Кардано–Декарта мы знаем, что $p(x)$ либо имеет два положительных корня, либо не имеет ни одного; это определяется дальнейшими исследованиями.

7.2.2. Теорема Штурма и метод бисекций Штурма отделения вещественных корней

Напомним читателю, что двумя основными темами исследований Фурье в течение всей его жизни были теория теплоты и теория численного решения уравнений. Обе эти темы были развиты Штурмом, который имел личные и научные связи с Фурье. В 1829 г. рукопись Фурье о численном решении уравнений была представлена нескольким специалистам, среди которых был и Штурм, явно указывавший на большое влияние, оказанное на него этой работой.

Заслуга Штурма состоит в том, что он заменил последовательность Фурье последовательностью

$$\operatorname{sseq}(x) = \{p(x), p^{(1)}(x), r_1(x), \ldots, r_k(x)\}, \tag{S}$$

которая называется *последовательностью Штурма* или *цепью*. (Это — классическая последовательность Штурма; см. определение 7.2.7 для обобщения определения последовательности Штурма.) Эта новая последовательность получается применением алгоритма Евклида к полиномам $p(x)$ и $p^{(1)}(x)$, если взять в качестве $r_i(x)$, $i = 1, 2, \ldots, k$ полиномы, *противоположные* к полиномиальным остаткам, т.е. она определяется следующими соотношениями:

$$\begin{aligned} p(x) &= p^{(1)}(x)q_1(x) - r_1(x), \\ p^{(1)}(x) &= r_1(x)q_2(x) - r_2(x), \\ &\ldots \\ r_{k-2}(x) &= r_{k-1}(x)q_k(x) - r_k(x). \end{aligned} \tag{S1}$$

Преимущество последовательности Штурма состоит в том, что мы можем теперь получить *точное* число вещественных корней уравнения $p(x) = 0$ на данном интервале. Заметим, что если n — степень полинома $p(x)$, то обычно получается n *дополнительных* функций $p^{(1)}(x), r_1(x), \ldots, r_k(x)$, потому что при вычислении наибольшего

общего делителя полиномов $p(x)$ и $p^{(1)}(x)$ степень каждого остатка обычно на единицу меньше степени предыдущего остатка. Кроме того, если нет кратных корней, то $r_k(x)$ — константа. [В гл. 5 мы подробно рассмотрим различные способы построения последовательности (S) над целыми числами.]

Пример. Последовательность Штурма, соответствующая полиному $p(x) = x^3 - 7x + 7$ — это $\text{sseq}(x) = \{x^3 - 7x + 7, 3x^2 - 7, 2x - 3, 1\}$.

Мы имеем следующее

Определение 7.2.7 (определение обобщенной последовательности Штурма). Пусть $p(x) = 0$ — полиномиальное уравнение с рациональными коэффициентами без кратных корней. Тогда, начиная с $p(x)$, можно построить последовательность полиномов (как описано ниже), называемую *(обобщенной) последовательностью Штурма*,

$$\text{general-sseq}(x) = \{p(x), p_1(x), \ldots, p_{k+1}(x)\}$$

со следующими свойствами в интервале (p, q) (т.е. когда x возрастает от p до q):

i. В достаточно малой окрестности вещественного корня α полинома $p(x)$ знаки полиномов $p(x)$ и $p_1(x)$ различны, если $x < \alpha$, и одинаковы, если $x > \alpha$.

ii. Два последовательных члена последовательности Штурма general-sseq(x) не могут одновременно обращаться в нуль.

iii. Если одна из функций в последовательности Штурма general-sseq(x) обращается в нуль при некотором значении x_0, то значения соседних функций этой последовательности, вычисленные в той же точке, имеют противоположные знаки.

iv. Последняя функция $p_{k+1}(x)$ не обращается в нуль, а следовательно, не меняет знак.

После теоремы 7.2.9 мы представим способ построения бесконечного числа наборов функций [отличных от последовательности, задаваемой соотношениями (S1)], которые удовлетворяют условиям i–iv определения 7.2.7; кроме того, как мы убедимся при доказательстве теоремы 7.2.8, последовательность sseq(x), определяемая соотношениями (S1), также удовлетворяет этим условиям.

Хотя теоремы 7.2.8 и 7.2.9 верны для любой general-sseq(x), они будут сформулированы и доказаны только для «классической» последовательности Штурма sseq(x) (S).

Точное число вещественных корней уравнения в открытом интервале.

Теорема 7.2.8 (Штурм, 1829 г.). Рассмотрим полиномиальное уравнение $p(x) = 0$ с целыми коэффициентами, имеющее только простые корни. Тогда число его вещественных корней в интервале (p, q) равно разности $v(p) - v(q)$, где $v(\xi)$ обозначает число перемен знаков в последовательности Штурма $\mathrm{sseq}(x)$ для $x = \xi$.

Доказательство. Доказательство основано на условиях i–iv определения 7.2.7, которые характеризуют функции последовательности Штурма $\mathrm{sseq}(x)$, определенной соотношениями (S1).

a. В достаточно малой окрестности вещественного корня α полинома $p(x)$ знаки полиномов $p(x)$ и $p^{(1)}(x)$ противоположны, если $x < \alpha$, и одинаковы, если $x > \alpha$ (см. лемму 7.2.1).

b. Два соседних элемента последовательности Штурма не могут одновременно обращаться в нуль. Чтобы в этом убедиться, предположим противное, т.е. пусть $r_i(x_0)$ и $r_{i+1}(x_0)$ — оба нули для некоторого значения x_0. Тогда из (S1) видно, что для того же самого значения x_0 мы также имеем $r_{i+1}(x_0) = r_{i+2}(x_0) = \cdots = r_k(x_0) = 0$. Это, однако, противоречит тому, что $r_k(x)$ — ненулевая константа. [Очевидно, что тот же самый результат имеет место также для $p(x)$ и $p^{(1)}(x)$.]

c. Если одна из функций в последовательности Штурма обращается в нуль при некотором значении x_0, то в той же точке соседние функции в этой последовательности имеют противоположные знаки. Действительно, если $r_i(x_0) = 0$, то

$$r_{i-1}(x_0) = r_i(x_0)q_{i+1}(x_0) - r_{i+1}(x_0),$$

откуда следует, что

$$r_{i-1}(x_0) = -r_{i+1}(x_0). \quad [\text{Это применимо также к } p^{(1)}(x).]$$

Установив эти основные свойства, нам нужно теперь показать, что при изменении x от p до q последовательность Штурма теряет одну перемену знаков, если x проходит через корень α уравнения $p(x) = 0$, и что в отличие от последовательности Фурье, она не теряет перемену знаков, если x проходит через корень другого элемента последовательности. Действительно, из свойства (a) мы видим, что, когда x проходит через корень α уравнения $p(x) = 0$, мы теряем ровно одну перемену знаков. Предположим теперь, что x проходит через α_i, корень уравнения $r_i(x) = 0$. Из свойств (b) и (c) следует, что числа $r_{i-1}(\alpha_i)$ и $r_{i+1}(\alpha_i)$ отличны от нуля и имеют противоположные знаки.

x	r_{i-1}	r_i	r_{i+1}	r_{i-1}	r_i	r_{i+1}
$\alpha_i - \varepsilon$	$+$	$\pm$	$-$	$-$	$\pm$	$+$
α_i	$+$	0	$-$	$-$	0	$+$
$\alpha_i + \varepsilon$	$+$	$\mp(\pm)$	$-$	$-$	$\mp(\pm)$	$+$

Мы можем тогда выбрать малую окрестность $(\alpha_i - \varepsilon, \alpha_i + \varepsilon)$, $\varepsilon > 0$, где эти две функции не меняют знаков, и построить следующую таблицу:

[Отметим, что α_i *не* обязан быть простым корнем уравнения $r_i(x) = 0$.]

Из этой таблицы видно, что в группе трех функций $r_{i-1}(x)$, $r_i(x)$, $r_{i+1}(x)$ нет потерь перемен знаков при прохождении x через корень полинома $r_i(x)$, и это завершает доказательство. □

Пример. Рассмотрим полиномиальное уравнение $p(x) = x^3 - 7x + 7 = 0$, последовательность Штурма которого равна $\text{sseq}(x) = \{x^3 - 7x + 7, 3x^2 - 7, 2x - 3, 1\}$. Тогда, пользуясь теоремой 7.2.8, мы можем с уверенностью утверждать, что у $p(x)$ есть два вещественных корня в интервале $(0, 2)$, поскольку, вычисляя значения последовательности Штурма при $x = 0$, мы получаем $\text{sseq}(0) = \{7, -7, -3, 1\}$ с двумя переменами знаков, а вычисляя ее при $x = 2$, мы получаем $\text{sseq}(2) = \{1, 5, 1, 1\}$ без перемен знаков, и $v(0) - v(2) = 2 - 0 = 2$.

Сам Штурм утверждал, что теорема 7.2.8 была просто побочным продуктом его интенсивных исследований в области линейных разностных уравнений второго порядка. Требование, чтобы уравнение $p(x) = 0$ имело только простые корни, не является ограничением общности, поскольку мы можем сначала выполнить разложение на свободные от квадратов множители (см. разд. 3.2.4), а затем воспользоваться теоремой Штурма. Более того, всякий раз, когда последовательность Штурма имеет $n + 1$ элементов, где $n = \deg[p(x)]$, мы можем легко определять число пар комплексных корней уравнения $p(x) = 0$ следующим образом.

Теорема 7.2.9 (Sturm 1835). Рассмотрим полиномиальное уравнение $p(x) = 0$ степени n с целыми коэффициентами без кратных корней. Тогда $p(x)$ имеет столько же пар комплексных корней, сколько имеется перемен знаков в последовательности первых членов n функций $p^{(1)}(x), r_1(x), \ldots, r_k(x)$ последовательности Штурма $\text{sseq}(x)$.

Доказательство. Справедливость этого правила основана на том, что одна из любых двух соседних функций последовательности Штурма

имеет четную степень, а другая — нечетную. Поэтому, если эти две функции имеют один и тот же знак при $x = +\infty$, то они будут иметь противоположные знаки при $x = -\infty$, и наоборот. Таким образом, если мы вычисляем последовательность Штурма $\mathrm{sseq}(x)$ в $x = +\infty$ и в $x = -\infty$, то каждая перемена знаков в любой из этих последовательностей будет соответствовать *постоянству* в другой, т.е. число постоянств в последовательности Штурма, вычисленной в $x = -\infty$, равняется числу перемен знаков в последовательности Штурма, вычисленной в $x = +\infty$.

Пусть i — число перемен знаков в $\mathrm{sseq}(+\infty)$. Эти перемены получены из знаков коэффициентов при самых высоких степенях x в n дополнительных функциях $p^{(1)}(x), r_1(x), \ldots, r_k(x)$, где первые члены полиномов $p(x)$ и $p^{(1)}(x)$ предполагаются положительными.

Однако мы видели, что $\mathrm{sseq}(-\infty)$ будет содержать i постоянств или, эквивалентно, $n - i$ перемен знаков. [Здесь мы пользуемся тем, что последовательность Штурма содержит $n + 1$ функций и что в $\mathrm{sseq}(x)$ число перемен плюс число постоянств в сумме дают n.]

Пользуясь теоремой 7.2.8, мы видим, что число вещественных корней уравнения $p(x) = 0$ между $-\infty$ и $+\infty$ равно числу перемен знаков в $\mathrm{sseq}(-\infty)$ минус число перемен знаков в $\mathrm{sseq}(+\infty)$. Следовательно, уравнение $p(x)$ имеет $n - 2i$ вещественных корней, а значит, $2i$ комплексных корней, которые появляются попарно. Таким образом, мы имеем i пар комплексных корней. □

Пример. Рассмотрим полином из последнего примера $p(x) = x^3 - 7x + 7$, последовательность Штурма которого равна $\mathrm{sseq}(x) = \{x^3 - 7x + 7, 3x^2 - 7, 2x - 3, 1\}$. Ясно, что коэффициенты $\{1, 3, 2, 1\}$ первых членов всех элементов последовательности $\mathrm{sseq}(x)$ положительны, и, поскольку перемен знаков нет, полином $p(x)$ не имеет комплексных корней.

Для полноты мы представим сейчас другой способ построения $\mathrm{general\text{-}sseq}(x)$; таким образом, мы покажем, что существует бесконечное число наборов функций, которые могут использоваться, чтобы отделить вещественные корни полиномиального уравнения.

Пусть $p_1(x)$ — производная полинома $p(x), \deg[p(x)] = n$; умножим $p_1(x)$ на двучлен $(p_1 x + q_1)$, где p_1 и q_1 — неизвестные, и вычтем из произведения $p(x)$. В результате получим полином степени n, который разделим на полином $a_1 x^2 + b_1 x + c_1$, где a_1, b_1, c_1 — заданные числа, такие, что $a_1 x^2 + b_1 x + c_1$ остается положительным для всех вещественных значений x [или по крайней мере обращается в нуль не

более чем для одного значения x, которое не является корнем полинома $p_1(x)$, и остается положительным для всех других значений x]. Разделив $p_1(x)(p_1x+q_1)-p(x)$ на $a_1x^2+b_1x+c_1$, получаем частное $p_2(x)$, $\deg[p_2(x)]=n-2$, содержащее p_1 и q_1 в первой степени во всех своих членах, и остаток первой степени вида $Kx+L$, где коэффициенты K, L также содержат p_1 и q_1 в первой степени. Приравнивая эти коэффициенты K, L нулю, получаем численные значения для p_1 и q_1; после подстановки этих значений p_1 и q_1 в $p_2(x)$ последний полином становится полностью определенным. Поэтому мы имеем соотношение

$$p_1(x)(p_1x+q_1)-p(x)=p_2(x)(a_1x^2+b_1x+c_1),$$

или

$$p(x)=p_1(x)(p_1x+q_1)-p_2(x)(a_1x^2+b_1x+c_1).$$

Если коэффициент при x^{n-2} в полиноме $p_2(x)$ ненулевой, то мы можем продолжать этот процесс и получить функцию $p_3(x)$, такую, что

$$p_1(x)=p_2(x)(p_2x+q_2)-p_3(x)(a_2x^2+b_2x+c_2).$$

Однако, если $\deg[p_2(x)]=n-3$, то мы заменяем p_2x+q_2 трехчленом $p_2x^2+q_2x+r_2$ и получаем соотношение

$$p_1(x)=p_2(x)(p_2x^2+q_2x+r_2)-p_3(x)(a_2x^2+b_2x+c_2).$$

Если $p(x)$ не содержит кратных корней, то в конце мы получаем функцию $p_{k+1}(x)$, которая является числовой константой. Читателю следует проверить, что эта новая последовательность удовлетворяет условиям определения 7.2.7 и, следовательно, является последовательностью Штурма.

Пример. Применим описанную процедуру к полиному $p(x)=x^3-7x+7$, где, очевидно, $p_1(x)=3x^2-7$. Тогда $p_1(x)(p_1x+q_1)-p(x)=(3p_1-1)x^3+3q_1x^2+(7-7p_1)x-7q_1-7$, и, разделив этот полином на x^2+x+2, мы получим частное $p_2(x)=(3p_1-1)x-3p_1+3q_1+1$ и остаток $[-10p_1-3q_1+8]x+[6p_1-13q_1-9]$. Приравнивая остаток нулю, получаем $p_1=131/148$ и $q_1=-21/74$, а подставляя эти значения в $p_2(x)$, получаем $p_2(x)=(245/148)x-371/148$, т.е.

$$x^3-7x+7=(3x^2-7)\left[\left(\frac{131}{148}\right)x-\frac{21}{74}\right]-\left[\left(\frac{245}{148}\right)x-\frac{371}{148}\right](x^2+x+2).$$

Читателю в качестве упражнения предлагается вычислить свободный член и закончить пример.

Теорема Штурма может использоваться, чтобы отделить вещественные корни полиномиального уравнения $p(x) = 0$ с целыми коэффициентами и без кратных корней. Наиболее эффективный способ сделать это — сначала отделить положительные корни, а затем — отрицательные (оригинальное предложение Штурма). Этот процесс совсем простой и описан ниже; для анализа времени вычисления этого алгоритма нам потребуется следующее

Определение 7.2.10. Пусть $p(x)$ — полином с целыми коэффициентами от одной переменной, $\deg[p(x)] = n \geq 1$; более того, предположим, что $p(x)$ имеет k, $k \leq n$, различных корней $\rho_1, \ldots, \rho_k$. Если $k \geq 2$, то определим *сепаратор* $\Delta > 0$ полинома $p(x)$ формулой

$$\Delta = \min_{1 \leq i < j \leq k} |\rho_i - \rho_j|.$$

Если $k = 1$, то $\Delta = \infty$.

Как мы увидим в разд. 7.2.4, при $n \geq 2$ нижняя граница сепаратора Δ дается формулой

$$\Delta \geq \sqrt{3} \cdot n^{-(n+2)/2} \cdot |p(x)|_1^{-(n-1)}. \tag{S2}$$

STURM. Метод бисекций Штурма отделения вещественных корней уравнения (**Sturm**'s Bisection Method for Isolation of the Real Roots of an Equation)

Вход: $p(x) = 0$, полиномиальное уравнение с целыми коэффициентами и без кратных корней.

Выход: Изолирующие интервалы вещественных корней полинома $p(x)$ или точные значения его корней.

1. [Инициализация] Положить $p_w(x) := p(x)$; если $p_w(0) = 0$, то вернуть замкнутый интервал $[0, 0]$ и положить $p_w(x) := p_w(x)/x$. Вычислить последовательность Штурма $\mathrm{sseq}(x)$ по формуле (S1), соответствующую полиному $p_w(x) = 0$, и положить $pn\text{-}flag := 0$. (Когда $pn\text{-}flag = 0$, мы отделяем положительные корни, а когда $pn\text{-}flag = 1$, отделяем отрицательные.)

2. [Вычисление границы корней] Пользуясь правилом Коши, описанным в разд. 7.2.3, вычислить верхнюю границу b значений положительных корней полинома $p_w(x)$, если $pn\text{-}flag = 0$, или значений положительных корней полинома $p_w(-x)$, если $pn\text{-}flag = 1$, так что положительные или отрицательные корни

полинома $p_w(x)$ находятся в интервале $(0, b]$ или $[-b, 0)$ соответственно. [В $p_w(-x)$ отрицательные корни полинома $p_w(x)$ становятся положительными; $(0, b]$ — замкнутый справа интервал, а $[-b, 0)$ — интервал, замкнутый слева.] Если $pn\text{-}flag = 0$, то выполнять {если $p_w(b) = 0$, то выдать замкнутый интервал $[b, b]$; $l_1 := 0$; $r_1 := b$}, иначе выполнять {если $p_w(-b) = 0$, то вернуть замкнутый интервал $[-b, -b]$; $l_1 := -b$; $r_1 := 0$}. Положить $I := \emptyset$, где I — список интервалов.

3. [Обновление интервала] Положить $l := l_1$ и $r := r_1$.

4. [Основной цикл] Пользуясь теоремой 7.2.8, вычислить число положительных корней в интервале (l, r). Если имеется только один корень, то (l, r) — его изолирующий интервал, он подается на выход. Если интервал содержит более одного корня, то он делится на два подынтервала $(l, (l + r)/2)$ и $((l + r)/2, r)$ равной длины, которые добавляются к списку I; если $p_w[(l + r)/2] = 0$, то замкнутый интервал $[(l + r)/2, (l + r)/2]$ подается на выход.

5. [Конец?] Если $I \neq \emptyset$, то удалить из этого списка первый интервал (l_1, r_1) и перейти к шагу 3; если $I = \emptyset$ и pn-flag $= 1$, то выход.

6. [Отделение отрицательных корней] Если $p_w(x) \neq p_w(-x)$, то выполнять {положить $pn\text{-}flag := 1$; перейти к шагу 2}, иначе выход. (Если мы выходим здесь, то отрицательные корни симметричны положительным, и мы уже знаем их изолирующие интервалы. Конечно, в этом случае интервалы, которые мы получаем для отрицательных корней, находятся в положительной полуплоскости и должны отображаться на соответствующие интервалы в отрицательной полуплоскости, а это тривиально.)

Анализ времени работы алгоритма ***STURM***. Из описания алгоритма становится очевидно, что метод Штурма — это по существу метод *бисекций*. Наша реализация метода, когда мы вычисляем раздельно границы для положительных и отрицательных корней и изолируем их отдельно, является очень эффективной, потому что мы минимизируем число бисекций, которые должны быть выполнены. Более того, если $p(x) = p(-x)$, то нам ничего не нужно делать, чтобы изолировать отрицательные корни, поскольку они симметричны положительным.

На шаге 1 мы вычисляем последовательность Штурма $\mathrm{sseq}(x)$, а в гл. 5 мы видели, что это выполняется за время $O\{n^5 L^2 [|p(x)|_\infty]\}$.

На шаге 2 мы вычисляем верхнюю границу b значений положительных корней полинома $p(x)$ [или $p(-x)$], и, как мы увидим в разд. 7.2.3,

это выполняется за время $O\{n^2 L[|p(x)|_\infty]\}$. Кроме того, мы берем границу b в виде двоично-рационального числа, т.е. его знаменатель — это степень 2, и, следовательно, на шаге 4 все рациональные числа, получающиеся при бисекциях исходного интервала $(0, b)$ [или $(-b, 0)$], будут также двоично-рациональными.

Вычислим теперь время, необходимое для выполнения шага 4. Заметим, что, пользуясь теоремой Штурма, мы фактически *не должны* вычислять значение каждого элемента последовательности Штурма в данной рациональной точке, поскольку нам нужен только знак значения полинома в этой точке. Отсюда легко следует, что если $p(x) = \sum_{0 \le i \le n} c_i x^i$ и a/d — ненулевое рациональное число, $d > 0$, то знак числа $p(a/d)$ совпадает со знаком $\sum_{0 \le i \le n} c_i a_i d^{n-i}$ (см. также численный пример ниже). Поэтому мы можем определить знак полинома в рациональной точке, пользуясь только целочисленной арифметикой. Из разд. 3.1.2 легко вывести, что время, необходимое для этого вычисления, равно

$$O\{n^2 L^2(e) L[|p(x)|_\infty]\}, \tag{S3}$$

где $e = \max\{|a|, d\}$. Мы берем худший возможный случай, а именно что *каждое* значение полинома вычисляется с наибольшим значением e, появляющимся в вычислениях; конечно, e зависит от числа делений пополам, которые должны быть выполнены, чтобы отделить корни. Чтобы вычислить границу для $L(e)$, прологарифмируем (S2) и легко убедимся, что

$$L(e) \le |L(\Delta^{-1})| = O\{nL[|p(x)|_\infty] + nL(n)\}.$$

Более того, n в большинстве представляющих интерес случаев маленькое, $L(n) = 1$, и мы можем считать, что

$$|L(\Delta^{-1})| = O\{nL(|p(x)|_\infty]\},$$

что также ограничивает число делений пополам. Поэтому

$$L(e) = O\{nL(|p(x)|_\infty]\},$$

и, подставляя последнее выражение в (S3), видим что в худшем случае *каждое* полиномиальное значение вычисляется за время

$$O\{n^4 L^2[|p(x)|_\infty]\}.$$

При условии, что мы имеем $O\{nL(|p(x)|_\infty]\}$ делений пополам каждого из не более, чем n, интервалов, содержащих корни, и что мы имеем не более, чем $n+1$, полиномов в последовательности Штурма, мы видим, что

$$T_{\mathbf{STURM}}[p(x)] = O\{n^7 L^3[|p(x)|\infty]\}.$$

(См. также историческое замечание 3.)

Пример. Отделим корни уравнения $p(x) = x^3 - 7x + 7 = 0$, где $p(0) \neq 0$; в этом примере $p_w(x) = p(x)$. Мы уже знаем, что $\mathrm{sseq}(x) = \{x^3 - 7x + 7, 3x^2 - 7, 2x - 3, 1\}$, и будем этим пользоваться, чтобы отделить как положительные, так и отрицательные корни.

Отделение положительных корней. Пользуясь алгоритмом **BPR** разд. 7.2.3, получаем $b = 4$ как верхнюю границу положительных корней полинома $p(x)$ (см. также соответствующий пример в разд. 7.2.3), т.е. они все находятся в интервале $(0, 4)$, где $p(0) \neq 0$ и $p(4) \neq 0$. Пользуясь теоремой Штурма, находим, что в интервале $(0, 4)$ есть два корня; а именно, в $x = 0$ мы получаем $\mathrm{sseq}(0) = \{7, -7, -3, 1\}$ с двумя переменами знаков, а в $x = 4$ мы получаем $\mathrm{sseq}(4) = \{43, 41, 5, 1\}$ без перемен знаков. Разделив пополам интервал $(0, 4)$, видим, что $p(2) \neq 0$ и что $\mathrm{sseq}(2) = \{1, 5, 1, 1\}$ не имеет перемен знаков; поэтому оба корня находятся в $(0, 2)$ и интервал $(2, 4)$ исключается из рассмотрения. Разделив пополам теперь интервал $(0, 2)$, видим, что $p(1) \neq 0$ и что $\mathrm{sseq}(1) = \{1, -4, -1, 1\}$ также имеет две перемены знаков. Поэтому оба корня находятся в $(1, 2)$ и интервал $(0, 1)$ игнорируется. Затем делим пополам интервал $(1, 2)$ и видим, что $p(3/2) \neq 0$ и $\mathrm{sseq}(3/2) = \{-1/8, -1/4, 0, 1\}$ имеет одну перемену знаков; таким образом, изолирующие интервалы для двух положительных корней суть $(1, 3/2)$ и $(3/2, 2)$. Заметим, что при вычислении $\mathrm{sseq}(3/2)$ нам не обязательно прибегать к арифметике рациональных чисел. Как мы установили при анализе времени работы алгоритма Штурма, достаточно вычислить только знаки последовательности, а именно $\mathrm{sseq}(3/2) = \{-, -, 0, +\}$, где, например, первый знак соответствует знаку *числителя* числа $[3^3 - 7 \cdot 3 \cdot 2^2 + 7 \cdot 2^3]/2^3 = (3/2)^3 - 7 \cdot (3/2) + 7$, полученного с использованием только целочисленной арифметики.

Отделение отрицательных корней. Мы заменяем x на $-x$ в $p(x) = 0$ и получаем $p(-x) = x^3 - 7x - 7 \neq p(x)$. Мы сделаем вид, будто не знаем, что имеется только один отрицательный корень, и вычислим верхнюю границу значений положительных корней полинома $p(-x)$; на этот раз снова $b = 4$, и, поскольку $p(-4) \neq 0$, все отрицательные корни полинома $p(x)$ находятся в интервале $(-4, 0)$. Пользуясь теоремой Штурма, находим, что в $(-4, 0)$ имеется только один корень; а именно, в $x = -4$ мы получаем $\mathrm{sseq}(-4) = \{-29, 41, -11, 1\}$ с тремя переменами знаков, а в $x = 0$ мы получаем $\mathrm{sseq}(0) = \{7, -7, -3, 1\}$ с двумя переменами знаков. Поэтому изолирующий интервал для отрицательного корня полинома $p(x)$ — это $(-4, 0)$.

Из приведенного примера становится очевидным, что мы долж-

ны сосредоточиться на том, как (1) вычислять верхнюю границу b значений положительных корней и (2) определять число делений пополам, необходимых, чтобы отделить корни. Как мы видели при обсуждении времени вычислений метода Штурма, второй вопрос связан с тем, насколько близко расположены корни, и имеет чрезвычайно важное значение как для гарантии того, что процесс завершается после конечного числа шагов, так и при вычислении его теоретических временных оценок. Обе темы рассматриваются ниже.

В разд. 7.2.3 мы представляем правило Коши, очень эффективное средство для вычисления верхней границы b значений *положительных* корней уравнения, а затем, в разд. 7.2.4, мы доказываем соотношение (S2), результат, принадлежащий Малеру (Mahler, 1964), который дает нам нижнюю границу сепаратора корней; см. также (Rump, 1979).

7.2.3. Вычисление верхней (нижней) границы значений положительных корней полиномиального уравнения

В этом разделе мы сформулируем и докажем правило Коши вычисления верхней границы значений положительных корней полиномиального уравнения с целыми коэффициентами. Затем следует обсуждение того, как это правило лучше всего реализовать. Вычисляемая граница является рациональным числом, знаменатель которого — степень 2 (см. историческое замечание 4).

Теорема 7.2.11 (правило Коши). Пусть $p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0 = 0$ — нормированное полиномиальное уравнение степени $n > 0$ с целыми коэффициентами, где $c_{n-k} < 0$ по крайней мере для одного k, $1 \le k \le n$, и пусть λ — число его отрицательных коэффициентов. Тогда

$$b = \max_{\{1 \le k \le n : c_{n-k} < 0\}} \{|\lambda c_{n-k}|^{1/k}\} \tag{C}$$

является верхней границей значений положительных корней уравнения $p(x) = 0$.

Доказательство. Из определения b мы заключаем, что

$$b^k \ge \lambda |c_{n-k}|$$

для каждого k, такого, что $c_{n-k} < 0$; для этих k последнее неравенство можно переписать в виде

$$b^n \ge \lambda |c_{n-k}| b^{n-k}.$$

Суммируя по всем соответствующим k, получаем

$$\lambda b^n \geq \lambda \sum_{\{1 \leq k \leq n : c_{n-k} < 0\}} |c_{n-k}| b^{n-k},$$

или

$$b^n \geq \sum_{\{1 \leq k \leq n : c_{n-k} < 0\}} |c_{n-k}| b^{n-k}.$$

Из последнего неравенства мы заключаем, что если мы подставим b вместо x в $p(x) = 0$, то первый член, т.е. b^n, будет не меньше суммы абсолютных значений всех отрицательных слагаемых. Поэтому $p(x) \neq 0$ для всех $x > b$. □

С первого взгляда может показаться, что теорема 7.2.11 требует большого количества вычислений, поскольку кажется, что необходимо вычислять корни k-й степени. Однако это не так, потому что для каждого k, такого, что $c_{n-k} < 0$, мы вычисляем наименьшее целое число k', такое, что

$$\left| \frac{\lambda c_{n-k}}{c_n} \right|^{1/k} \leq 2^{k'},$$

а затем мы полагаем $b = 2^{k''+1}$, где k'' — максимум из всех k'. [Заметим, что мы рассматриваем общий случай, когда $p(x)$ — не обязательно нормированный полином и $c_n > 0$; см. (Akritas, 1981b).] Вычисление каждого k' осуществляется следующим образом.

Пусть

$$\frac{b}{c} = \left| \frac{\lambda c_{n-k}}{c_n} \right|$$

— частное для некоторого k, $1 \leq k \leq n$, и предположим, что

$$2^i \leq b < 2^{i+1} \quad \text{и} \quad 2^j \leq c < 2^{j+1}.$$

Тогда, очевидно,

$$2^{i-j-1} < \frac{b}{c} < 2^{i-j+1}.$$

Если положить $p = i - j - 1$ и $p + 2 = i - j + 1$, то это неравенство примет вид

$$2^p < \frac{b}{c} < 2^{p+2}.$$

Извлечение корня k-й степени из последнего выражения дает

$$2^{p/k} < (b/c)^{1/k} < 2^{(p+2)/k}. \qquad \text{(C1)}$$

Если $p = q \cdot k + r$, $0 \le r < k$, то, очевидно,

$$2^q \le 2^{p/k}.$$

Более того, $p + 2 = q \cdot k + r + 2$ и

$$2^{(p+2)/k} = 2^q \cdot 2^{(r+2)/k} \le 2^{q+2}, \tag{C2}$$

поскольку $(r+2)/2 \le 2$ и $r \le k-1$. Объединяя (C1) и (C2), получаем $k' = q + 2$. Фактически мы можем получить меньшее значение для k', если $r \le k - 2$. В этом случае $(r+2)/k \le 1$ и

$$2^{(p+q)/k} \le 2^{q+1}, \tag{C3}$$

так что $k' = q + 1$.

Из приведенного обсуждения мы заключаем, что главные операции в правиле Коши следующие:

1. Вычисление $\lfloor \log_2 |i| \rfloor$, наибольшего целого $\le \log_2 |i|$, для любого целого $i \ne 0$.

2. Вычисление 2^k для неотрицательного целого числа k.

3. Вычисление $[|i|/2^k]$ для любых целых i и k, положительных или отрицательных, где по определению

$$[x] = \lfloor x \rfloor, \quad \text{если} \quad x \ge 0, \quad \text{и} \quad [x] = \lceil x \rceil, \quad \text{если} \quad x < 0.$$

Предполагая, что у нас есть эффективные алгоритмы для этих операций (см. также упражнения по программированию к этому разделу), мы имеем следующий алгоритм:

BPR. Верхняя граница значений положительных корней полиномиального уравнения (Upper **B**ound on Values of **P**ositive **R**oots of a Polynomial Equation)

Вход: Полиномиальное уравнение $p(x) = c_n x^n + c_{n-1} x^{n-1} + \ldots + c_0 = 0$ с целыми коэффициентами.

Выход: Рациональное число b, знаменатель которого — степень 2 (b — наименьшая степень 2, такая, что $|\lambda c_{n-k}/c_n|^{1/k} \le b$ для $1 \le k \le n$, и $c_{n-k} < 0$). Если у $p(x)$ нет отрицательных коэффициентов, то $b = 1$.

1. [Инициализация] Если $\mathrm{lc}[p(x)] < 0$, положить $p(x) := (-1)p(x)$, положить λ равным числу отрицательных коэффициентов полинома $p(x)$ и $k'' := 0$. Если $\lambda = 0$ или $\deg[p(x)] = 0$, то перейти к шагу 3; иначе {положить $j := \lfloor \log_2(c_n) \rfloor$; $t := 0$}.

2. [Обработка отрицательных членов] Для каждого члена $c_i x^{n_i}$, $c_i < 0$, выполнять следующее: $k := n - n_i$; $c_i' := |\lambda c_i|$; $i := \lfloor \log_2(c_i') \rfloor$; $p := i - j - 1$; $q := \mathbf{QUO}(p, k)$; $r := p - kq$; если $r < 0$, то выполнять $\{r := r + k$; $q := q - 1\}$; $k' := q + 1$; если $r = k - 1$, то выполнять $\{c_n' := [c_n / 2^{-k'k}]$; если $c_i' > c_n'$, то $k' := k' + 1\}$; если $t = 0$ или $k' > k''$, то $k'' := k'$; $t := 1$.

3. [Выход] Вернуть рациональное число $b := 2^{k''}$.

Анализ времени работы алгоритма ***BPR***. Принимая во внимание упражнение по программированию к этому разделу, мы видим следующее:

Шаг 1 выполняется за время $O\{L[|p(x)|_\infty]\}$.

Одно выполнение шага 2, на котором вычисляется k', такое, что $|\lambda c_{n-k}/c_n| \le 2^{k'}$, выполняется за время $\sim c\{k + L[|p(x)|_\infty)]\}$. Поскольку шаг 2 выполняется самое большее n раз, мы легко заключаем (суммируя по $k = 1, \ldots, n$), что его время работы есть $O\{n^2 L[|p(x)|_\infty]\}$.

Шаг 3 выполняется за время $O(|k''| + 1)$. Поскольку k'' — целое число с одинарной точностью, мы видим, что время выполнения шага 2 доминирует над временем выполнения всего алгоритма, и, следовательно,

$$t_{\mathbf{BPR}}[p(x)] = O\{n^2 L[|p(x)|_\infty]\}.$$

Пример. Воспользуемся правилом Коши, чтобы вычислить верхние границы положительных и отрицательных корней уравнения $p(x) = x^3 - 7x + 7 = 0$.

Для положительных корней мы используем полином $p(x)$, как он есть. Применяя алгоритм **BPR**, мы получаем на первом шаге $\lambda = 1$, $j = 0$ и $t = 0$. Второй шаг выполняется только один раз, и мы имеем $k = 2$, $c_{3-2}' = c_1' = 7$, $i = 2$, $p = 1$, $q = 0$, $r = 1$, $k' = 1$, $c_3' = 4$, а поскольку $c_1' > c_n'$, мы модифицируем $k' := 2$; в заключение мы полагаем $k'' := 2$ и из третьего шага получаем $b = 4$.

Для отрицательных корней мы заменяем x на $-x$, так что они становятся положительными. Мы тогда получаем полином $-x^3 + 7x + 7$ и снова применяем **BPR**. На этот раз на первом шаге полином заменяется на $x^3 - 7x - 7$ и т.д. Подробности оставлены читателю в качестве упражнения; ответ: $b = 4$.

Пока мы вычислили верхнюю границу b значений положительных корней уравнения $p(x) = 0$. Пользуясь той же самой процедурой

BPR, мы можем также вычислить *нижнюю* границу b_{lo} значений положительных корней уравнения $p(x) = 0$. Легко можно проверить, что $b_{lo} = 1/b_{lo\text{-inv}}$, где $b_{lo\text{-inv}}$ — верхняя граница значений положительных корней уравнения $p(1/x) = 0$. В разд. 7.3.2 мы будем обсуждать эффективный способ получения $p(1/x)$.

7.2.4. Вычисление нижней границы расстояния между любыми двумя корнями полиномиального уравнения

Дается полиномиальное уравнение $p(x) = 0$; в этом разделе мы будем вычислять выражение [в терминах степени и нормы полинома $p(x)$] для нижней границы расстояния между любыми двумя корнями (сепаратор корней) уравнения $p(x) = 0$ (см. также определение 7.2.10). Этот результат выводится из более общей теоремы Малера и играет важную роль в анализе времени работы не только метода Штурма, но вообще любого метода, используемого для отделения вещественных корней уравнения. [Представляет интерес работа (Mignotte et al., 1979).]

Теорема 7.2.12 (Малер). Если $p(x) = \sum_{0 \le i \le n} c_{n-i} x^i$ — свободный от квадратов полином степени $n \ge 2$ от одной переменной с целыми коэффициентами и Δ — сепаратор его корней, то

$$\Delta \ge \sqrt{3} n^{-(n+2)/2} |p(x)|_1^{-(n-1)}. \tag{M1}$$

Доказательство. Главные инструменты в этом доказательстве — неравенство теоремы 6.3.2 и теорема Адамара об определителях, которая может быть сформулирована следующим образом:

Если элементы d_{ij}, $i, j = 1, 2, \ldots, n$, определителя

$$d = \det \begin{bmatrix} d_{11} & \ldots & d_{1n} \\ & \ldots & \\ d_{n1} & \ldots & d_{nn} \end{bmatrix}$$

являются произвольными комплексными числами, то

$$|d| \le \{ \prod_{1 \le j \le n} (\sum_{1 \le i \le n} |d_{ij}|^2) \}^{1/2}, \tag{M2}$$

и равенство имеет место, если и только если

$$\sum_{1 \le i \le n} d_{ij} \bar{d}_{ik} = 0 \quad \text{для} \quad 1 \le j < k \le n,$$

где $\bar{d}_{ik}$ обозначает комплексно-сопряженное к d_{ik}. [Доказательство теоремы Адамара можно найти в книге (Marcus, Minc, 1965), упр. 5, с. 208.]

Для доказательства теоремы пусть $\rho_1, \rho_2, \ldots, \rho_n$ — корни уравнения $p(x) = \sum_{0 \le i \le n} c_{n-i} x^i$, и положим

$$r[p(x)] = |c_0| \prod_{1 \le i \le n} \max(|\rho_i|, 1).$$

Тогда из теоремы 6.3.2 работы (Specht, 1949) мы имеем неравенство

$$r[p(x)] \le |p(x)|_2, \tag{M3}$$

где из определения 1.2.6 следует, что $|p(x)|_\infty \le |p(x)|_2 \le |p(x)|_1 \le (n+1)|p(x)|_\infty$.

Пусть теперь корни полинома $p(x)$ пронумерованы так, что

$$|\rho_1| \ge |\rho_2| \ge \cdots \ge |\rho_N| > 1 \ge |\rho_{N+1}| \ge \cdots \ge |\rho_n|, \tag{M4}$$

и пусть

$$v[p(x)] = \prod_{1 \le i < j \le n} (\rho_i - \rho_j) = \prod_{1 \le i \le n} \prod_{i < j \le n} (\rho_i - \rho_j)$$

при условии, что $v[p(x)] = 1$ при $n = 1$. Хорошо известно, что $v[p(x)] = d_v$, где d_v — определитель Вандермонда

$$d_v = \det \begin{bmatrix} 1 & 1 & \ldots & 1 \\ \rho_1 & \rho_2 & \ldots & \rho_n \\ \rho_1^2 & \rho_2^2 & \ldots & \rho_n^2 \\ \ldots & & & \\ \rho_1^{n-1} & \rho_2^{n-1} & \ldots & \rho_n^{n-1} \end{bmatrix}.$$

Более того, положим $v''[p(x)] = (\rho_1 \rho_2 \ldots \rho_N)^{-(n-1)} v[p(x)]$. Тогда $v''[p(x)] = d_v''$, где

$$d_v'' = \det \begin{bmatrix} \rho_1^{-(n-1)} & \ldots & \rho_N^{-(n-1)} & 1 & \ldots & 1 \\ \rho_1^{-(n-2)} & \ldots & \rho_N^{-(n-2)} & \rho_{N+1} & \ldots & \rho_n \\ \ldots & & & & & \\ \rho_1^{-1} & \ldots & \rho_N^{-1} & \rho_{N+1}^{n-2} & \ldots & \rho_n^{n-2} \\ 1 & \ldots & 1 & \rho_{N+1}^{n-1} & \ldots & \rho_n^{n-1} \end{bmatrix}.$$

То есть d''_v получается из d_v умножением i-го столбца матрицы, соответствующей d_v, на $\rho_i^{-(n-1)}$, $1 \le i \le N$. Поскольку теперь абсолютное значение каждого элемента в вышеупомянутом определителе ≤ 1, из неравенства Адамара следует, что

$$|v''[p(x)]| = |d''_v| \le n^{n/2}. \tag{M5}$$

Равенство имеет место, если одновременно

$$|\rho_1| = |\rho_2| = \cdots = |\rho_n| = 1$$

и

$$\sum_{0 \le k \le n-1} \bar{\rho}_i^k \rho_j^k = 0, \quad \text{для} \quad 1 \le i < j \le n. \tag{M6}$$

Пусть $i = 1$; тогда, принимая во внимание хорошо известное соотношение между комплексными числами, мы видим, что (M6) принимает вид

$$\sum_{0 \le k \le n-1} \bar{\rho}_i^k \rho_j^k = \sum_{0 \le k \le n-1} (\bar{\rho}_1^{-1} \rho_j)^k = 0,$$

а умножая это выражение на $\bar{\rho}_1^{-1}\rho_j - 1$, получаем $(\bar{\rho}_1^{-1}\rho_j)^n = 1$, откуда видно, что n частных ρ_j/ρ_1, $1 \le j \le n$ — это n различных корней уравнения $x^n - 1 = 0$. Однако

$$x^n - 1 = \prod_{1 \le j \le n} \left(x - \frac{\rho_j}{\rho_1} \right),$$

или

$$\rho_1^n x^n - \rho_1^n = \prod_{1 \le j \le n} (x - \rho_j).$$

Таким образом, мы видим, что $|v''[p(x)]| = n^{n/2}$ только в случае $p(x) = c_0 x^n + c_n$, где $|c_0| = |c_n| > 0$.

Теперь пусть r и s фиксированы, $1 \le r < s \le n$. Мы намерены получить границу для $|d''_v/(\rho_r - \rho_s)|$ методом, аналогичным тому, который мы только что применяли к $|d''_v|$.

Сначала в матрице, соответствующей определителю Вандермонда d_v, вычтем s-й столбец из r-го, так что новый r-й столбец состоит из элементов

$$0, \rho_r - \rho_s, \rho_r^2 - \rho_s^2, \ldots, \rho_r^{n-1} - \rho_s^{n-1};$$

все они делятся на $\rho_r - \rho_s$. Затем разделим этот столбец на $\rho_r - \rho_s$ и получим в качестве новых элементов r-го столбца матрицы, соответствующей $|d_v/(\rho_r - \rho_s)|$, величины $q_0, q_1, \ldots, q_{n-1}$, где $q_0 = 0$ и $q_i = (\rho_r^i - \rho_s^i)/(\rho_r - \rho_s) = \rho_r^{i-1} + \rho_r^{i-2}\rho_s + \cdots + \rho_r \rho_s^{i-2} + \rho_s^{i-1}$, $i \ge 1$.

Поэтому частное $d_v/(\rho_r - \rho_s)$ может теперь быть переписано как определитель, в котором r-й столбец состоит из элементов $q_0, q_1, \ldots, q_{n-1}$, а остальные $n-1$ столбцов — те же, что и в исходном определителе d_v. Разделив теперь снова 1-й, 2-й, ..., N-й столбец матрицы, соответствующей новому определителю, на множители $\rho_1^{n-1}, \rho_2^{n-2}, \ldots, \rho_N^{n-1}$ соответственно, мы получаем определитель со значением $|d_v''/(\rho_r - \rho_s)|$. За исключением r-го столбца, этот определитель совпадает с определителем d_v''. Его r-й столбец состоит из элементов

$$q_0\rho_r^{-(n-1)}, q_1\rho_r^{-(n-1)}, \ldots, q_{n-1}\rho_r^{-(n-1)},$$

если $r \le N$, и элементов

$$q_0, q_1, \ldots, q_{n-1},$$

если $r > N$. Поскольку

$$|\rho_r| \ge |\rho_s| \quad \text{и} \quad |\rho_r| \begin{cases} > 1 & \text{для} \quad r \le N, \\ < 1 & \text{для} \quad r > N, \end{cases}$$

абсолютные значения последовательных элементов r-го столбца определителя не превосходят $0, 1, 2, \ldots, n-2, n-1$ соответственно. Поэтому по неравенству Адамара мы имеем

$$\left|\frac{d_v''}{\rho_r - \rho_s}\right|^2 \le n^{n-1}\{0^2 + 1^2 + 2^2 + \cdots + (n-1)^2\}.$$

Поскольку сумма в скобках равна $n(n-1)(2n-1)/6$, что $< n^3/3$, окончательный результат принимает вид

$$\left|\frac{d_v''}{\rho_r - \rho_s}\right|^2 < (1/3)n^{(n+2)}. \tag{M7}$$

Решая (M7) относительно $|\rho_r - \rho_s|^2$, получаем

$$|\rho_r - \rho_s|^2 > 3n^{-(n+2)}(d_v'')^2,$$

или

$$|\rho_r - \rho_s|^2 > 3n^{-(n+2)}(\rho_1\rho_2 \ldots \rho_N)^{-2(n-1)}\{v[p(x)]\}^2.$$

Из разд. 5.2.2 нам известно, что дискриминант $\operatorname{discr}[p(x)]$ полинома $p(x)$ равен

$$c_0^{2n-2} \prod_{1 \le i \le n} \prod_{i < j \le n} (\rho_i - \rho_j)^2 = c_0^{2n-2}\{v[p(x)]\}^2,$$

и, следовательно, мы можем также написать

$$|\rho_r - \rho_s|^2 > 3n^{-(n+2)}(c_0\rho_1\rho_2\ldots\rho_N)^{-2(n-1)}c_0^{2n-2}\{v[p(x)]\}^2,$$

или

$$|\rho_r - \rho_s|^2 > 3n^{-(n+2)}\{r[p(x)]\}^{-2(n-1)}\operatorname{discr}[p(x)].$$

Наконец, поскольку коэффициенты полинома $p(x)$ — целые числа, мы имеем $|\operatorname{discr}[p(x)]| \geq 1$, так как $\operatorname{discr}[p(x)]$ не равен нулю. Более того, выбирая r и s так, чтобы $|\rho_r - \rho_s| = \Delta$, и пользуясь неравенством (M3), мы завершаем доказательство (M1). □

Пример. Для полинома $p(x) = x^3 - 7x + 7$ мы получаем

$$\Delta > \sqrt{3}\cdot 3^{-5/2}\cdot 15^{-2} = \frac{\sqrt{3}}{\sqrt{3^5}\cdot 15^2} \approx \frac{1\cdot 7}{\sqrt{243}\cdot 225} \approx \frac{1\cdot 7}{3510} \approx 0.0005.$$

Это означает, что наименьшее расстояние между любыми двумя корнями полинома $p(x)$ больше, чем 0.0005.

7.3

Теорема Бюдана и два метода цепных дробей для отделения вещественных корней

До сих пор мы рассматривали метод Штурма отделения вещественных корней уравнения. Исторически это был первый метод, подлежащий развитию, и он являлся весьма крупным достижением. В этом разделе мы исследуем два метода цепных дробей, описанные в литературе, для отделения вещественных корней уравнения; первый из этих методов был разработан в 1836 г. Винсентом (Vincent) и является экспоненциальным, в то время как второй был разработан в 1978 г. автором (Akritas, 1978a, 1980a, 1980b) и существенно быстрее метода Штурма (и всех остальных). Мы начнем с теоремы Бюдана, которая, как уже упоминалось, эквивалентна теореме Фурье.

7.3.1. Теорема Бюдана

Хотя теорема Бюдана появилась гораздо раньше теоремы Фурье, она не была замечена и почти не появлялась в стандартных текстах по теории уравнений. Однако она имеет очень большое значение, потому что составляет основу теоремы Винсента. Следующая формулировка этой теоремы взята из статьи Винсента 1836 г.

Другая верхняя граница числа для вещественных корней уравнения в открытом интервале.

Теорема 7.3.1 (Бюдан, 1807 г.). Если в уравнении $p(x) = 0$ относительно x степени $n > 0$ мы сделаем две подстановки $x := p + x'$ и $x := q + x''$, где p и q — вещественные числа, такие, что $p < q$, то будут справедливы следующие утверждения:

i. Преобразованное уравнение относительно $x' = x - p$ не может иметь меньше перемен знаков, чем преобразованное уравнение относительно $x'' = x - q$.

ii. Число вещественных корней уравнения $p(x) = 0$, расположенных между p и q, не может быть больше числа перемен знаков, потерянных в последовательности коэффициентов при переходе от преобразованного уравнения относительно $x' = x - p$ к преобразованному уравнению относительно $x'' = x - q$.

iii. Если число вещественных корней уравнения $p(x) = 0$, лежащих между p и q, меньше числа перемен знаков, потерянных в последовательности коэффициентов при переходе от преобразованного уравнения относительно $x' = x - p$ к преобразованному уравнению относительно $x'' = x - q$, то разность — четное число.

Доказательство. Доказательство аналогично доказательству теоремы 7.2.5. □

Теоремы 7.2.5 и 7.3.1 эквивалентны. В этом легко убедиться, если в последовательности Фурье заменить x любым вещественным числом α. Тогда $n + 1$ получающихся чисел пропорциональны соответствующим коэффициентам преобразованного полиномиального уравнения $p(x + \alpha) = \sum_{0 \le i \le n} [p^{(i)}(\alpha)/i!]x^i$, полученным с помощью формулы разложения Тейлора.

Теорема Бюдана так же, как теорема 7.2.5, дает нам верхнюю границу для числа вещественных корней уравнения $p(x) = 0$ внутри интервала (p, q). Однако она использует только подстановки $x := p + x'$ и $x := q + x''$ и не зависит от какой-либо последовательности полиномов; эти подстановки называются *подстановками Мёбиуса* или *дробно-линейными подстановками* и из-за их значимости ниже они исследуются по отдельности.

7.3.2. Подстановки Мёбиуса и их воздействие на корни уравнения

Существеннейшую роль в обсуждении теоремы Винсента (в разд. 7.3.3) играют подстановки вида $x := a + 1/x$. Они принадлежат классу дробно-линейных подстановок, или подстановок Мёбиуса, названных

в честь Августа Ф. Мёбиуса (1790–1868), который первым изучал соответствующие преобразования в проективной геометрии.

Определение 7.3.2. *Общая подстановка* определяется выражением

$$x := \mathbf{M}(x) = \frac{ax + b}{cx + d},$$

где a, b, c, d — комплексные числа, такие, что их определитель $ad - bc$ отличен от нуля; она называется также *подстановкой Мёбиуса* и сокращенно обозначается $x := \mathbf{M}(x),\ \det(\mathbf{M}) \neq 0$.

Для любого $x \in \mathbb{C}$ подстановка $x := \mathbf{M}(x)$ задается непосредственно выражением из определения 7.3.2 при условии, что $cx + d \neq 0$; в противном случае мы полагаем $\mathbf{M}(-d/c) = \infty$. Если $c = 0$, то мы должны иметь $ad \neq 0$, поскольку определитель коэффициентов должен быть ненулевым, и выражение, определяющее подстановку, принимает вид

$$x := \mathbf{M}(x) = \left(\frac{a}{d}\right) x + \frac{b}{d};$$

более того, в этом случае $\mathbf{M}(\infty) = \infty$, в то время как если $c \neq 0$, то $\mathbf{M}(\infty) = a/c$.

Легко видеть, что каждой подстановке Мёбиуса соответствует квадратная матрица ее коэффициентов, и это полностью определяет подстановку. Пусть

$$\boldsymbol{M} = \Bigg\{ \mathbf{M}(x), \quad \text{такая, что} \quad x := \mathbf{M}(x), \quad \mathbf{M} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}, \\ \det(\mathbf{M}) \neq 0,\ x \in \mathbb{C}' \Bigg\}$$

— множество всех подстановок Мёбиуса, где $\mathbb{C}' = \mathbb{C} \cup \{\infty\}$ — пополненная комплексная плоскость. В $\boldsymbol{M}$ вводится отношение равенства условием, что две подстановки $\mathbf{A}(x), \mathbf{B}(x) \in \mathbf{M}$ совпадают [другими словами $\mathbf{A}(x) = \mathbf{B}(x)$ для всех $x \in \mathbb{C}'$], если и только если существует элемент $\lambda \in \mathbb{C},\ \lambda \neq 0$, такой, что $\mathbf{A} = \lambda\mathbf{B}$ (последнее равенство — равенство матриц).

Теорема 7.3.3. Множество $\boldsymbol{M}$ подстановок Мёбиуса образует группу, изоморфную группе квадратных матриц ранга 2.

Доказательство. Мы уже определили отношение равенства на $\boldsymbol{M}$. Теперь определим произведение двух подстановок $\mathbf{A}(x), \mathbf{B}(x) \in \boldsymbol{M}$

как произведение соответствующих матриц. Так определенное произведение также является подстановкой, потому что

$$\begin{aligned} x := \mathbf{AB}(x) = \mathbf{A}[\mathbf{B}(x)] &= \frac{a_{11}\mathbf{B}(x) + a_{12}}{a_{21}\mathbf{B}(x) + a_{22}} \\ &= \frac{(a_{11}b_{11} + a_{12}b_{21})x + a_{11}b_{12} + a_{12}b_{22}}{(a_{21}b_{11} + a_{22}b_{21})x + a_{21}b_{12} + a_{22}b_{22}} \\ &= \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix}(x) = (\mathbf{AB})(x). \end{aligned}$$

Отметим, что подстановки применяются последовательно *слева направо*.

В качестве единичного элемента в $\boldsymbol{M}$ возьмем тождественную подстановку, $x := x$, матрица которой имеет вид

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Из нашего определения равенства видно, что $\lambda\mathbf{I}$, где $\lambda \neq 0$ и $\lambda \in \mathbb{C}$, совпадает с тождественной подстановкой. Для $\mathbf{M}(x) \in \boldsymbol{M}$ обратная подстановка $x := \mathbf{M}(x)$ — это $x := \mathbf{M}^{-1}(x)$, где

$$\mathbf{M}^{-1} = \begin{bmatrix} \dfrac{1}{\det(\mathbf{M})} \end{bmatrix} \begin{bmatrix} m_{22} & -m_{12} \\ -m_{21} & m_{11} \end{bmatrix}$$

является обратной матрицей для $\mathbf{M}$; очевидно, что $\det(\mathbf{M}^{-1}) \neq 0$. [Отметим, что по определению равенства мы можем взять

$$\mathbf{M}^{-1} = \begin{bmatrix} -m_{22} & m_{12} \\ m_{21} & -m_{11} \end{bmatrix},$$

что немедленно получается, если решить $w = \mathbf{M}(z)$ относительно z]. □

Группа $\boldsymbol{M}$ неабелева, поскольку в общем случае $\mathbf{A}(x)\mathbf{B}(x) \neq \mathbf{B}(x)\mathbf{A}(x)$.

Определение 7.3.4. Следующие три подстановки Мёбиуса называются *порождающими подстановками* группы $\boldsymbol{M}$:

i. *Сдвиг*: $x := a + x = \begin{bmatrix} -1 & a \\ 0 & 1 \end{bmatrix}(x)$.

ii. *Растяжение*: $x := ax = \begin{bmatrix} a & 0 \\ 0 & 1 \end{bmatrix}(x)$.

iii. *Инверсия*: $x := 1/x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}(x)$.

Если a — комплексное число, то растяжение называется *вращением*.

Следующая теорема проясняет значение порождающих подстановок.

Теорема 7.3.5. Каждая подстановка $\mathbf{M}(x) \in \boldsymbol{M}$ получается перемножением подходящих порождающих подстановок (*умножения выполняются последовательно слева направо*). Таким образом, каждая общая подстановка осуществляется последовательностью порождающих подстановок.

Доказательство. Пусть

$$\mathbf{M} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

Чтобы доказать теорему, рассмотрим отдельно следующие два случая:

Случай (i): $c = 0$. В этом случае мы легко убеждаемся, что $\mathbf{M} = \mathbf{M}_1\mathbf{M}_2$, где

$$\mathbf{M}_1 = \begin{bmatrix} 1 & \frac{b}{d} \\ 0 & 1 \end{bmatrix}, \quad \mathbf{M}_2 = \begin{bmatrix} \frac{a}{d} & 0 \\ 0 & 1 \end{bmatrix},$$

поскольку

$$\mathbf{M}_1\mathbf{M}_2 = \begin{bmatrix} \frac{a}{d} & \frac{b}{d} \\ 0 & 1 \end{bmatrix} = (1/d) \begin{bmatrix} a & b \\ 0 & d \end{bmatrix}.$$

Другими словами, при $c = 0$ подстановка $\mathbf{M}(x)$ эквивалентна сдвигу $\mathbf{M}_1(x)$, *сопровождаемому* растяжением $\mathbf{M}_2(x)$.

Случай (ii): $c \neq 0$. В этом случае подстановка $\mathbf{M}(x)$ эквивалентна сдвигу $\mathbf{M}_1(x)$, сопровождаемому инверсией $\mathbf{M}_2(x)$, сопровождаемой другим сдвигом $\mathbf{M}_3(x)$, за которым, наконец, следует растяжение $\mathbf{M}_4(x)$, где

$$\mathbf{M}_1 = \begin{bmatrix} 1 & \frac{a}{c} \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{M}_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$
$$\mathbf{M}_3 = \begin{bmatrix} 1 & \frac{cd}{bc-ad} \\ 0 & 1 \end{bmatrix}, \quad \mathbf{M}_2 = \begin{bmatrix} \frac{c^2}{bc-ad} & 0 \\ 0 & 1 \end{bmatrix}.$$

Ясно, что их произведение равно

$$\begin{aligned} \mathbf{M}_1\mathbf{M}_2\mathbf{M}_3\mathbf{M}_4 &= \begin{bmatrix} \frac{a}{c} & 1 \\ 1 & 0 \end{bmatrix} \mathbf{M}_3\mathbf{M}_4 \\ &= \begin{bmatrix} \frac{a}{c} & \frac{bc}{bc-ad} \\ 1 & \frac{cd}{bc-ad} \end{bmatrix} \mathbf{M}_4 = \begin{bmatrix} \frac{ac}{bc-ad} & \frac{bc}{bc-ad} \\ \frac{c^2}{bc-ad} & \frac{dc}{bc-ad} \end{bmatrix} \\ &= \frac{c}{bc-ad} \begin{bmatrix} a & b \\ c & d \end{bmatrix}. \end{aligned}$$

□

Пример. Рассмотрим подстановку $x := 1/(1+x)$, матрица которой имеет вид

$$\mathbf{M} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

Применяя теорему 7.3.5, получаем

$$\mathbf{M}_1(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}(x), \text{ тождественная подстановка,}$$

$$\mathbf{M}_2(x) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}(x), \text{ инверсия,}$$

$$\mathbf{M}_3(x) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}(x), \text{ сдвиг,}$$

$$\mathbf{M}_4(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}(x), \text{ тождественная подстановка.}$$

Поэтому подстановка $x := 1/(1+x)$ эквивалентна инверсии $\mathbf{M}_2(x)$, сопровождаемой сдвигом $\mathbf{M}_3(x)$.

Из предыдущего примера и упр. 1 к этому разделу мы легко получаем, что подстановка $x := 1 + x_1$, $x_1 := 1 + x_2, \ldots, x_{a-1} := 1 + x_a$, за которой следует подстановка $x_a := 1/(1+x)$, эквивалентна подстановке $x := a + 1/x$, сопровождаемой подстановкой $x := 1 + x$.

Мы теперь подготовлены к исследованию действия подстановок Мёбиуса, или, эквивалентно, порождающих подстановок, на корнях полиномиального уравнения $p(x)$. Пусть

$$p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0 = (x-\alpha_1)(x-\alpha_2)\ldots(x-\alpha_n) = 0.$$

Когда имеет место сдвиг, мы заменяем x на $x+k$ для некоторого вещественного k и получаем полином[1)]

$$\begin{aligned} p(x+k) = p_t(x) &= (x+k-\alpha_1)\ldots(x+k-\alpha_n) \\ &= [x-(\alpha_1-k)]\ldots[x-(\alpha_n-k)] = 0. \end{aligned}$$

Таким образом, при сдвиге вещественная часть корней преобразованного полиномиального уравнения будет увеличиваться или уменьшаться в соответствии с тем, является k отрицательным или положительным.

[1)] Где индекс t — от $t$ranslation (сдвиг). — *Прим. перев.*

Когда выполняется растяжение, x заменяется на kx для некоторого вещественного k, $k \neq 0$. Следовательно,[2)]

$$\begin{aligned} p(kx) = p_s(x) &= k^n x^n + c_{n-1}k^{n-1}x^{n-1} + \cdots + c_1 kx + c_0 \\ &= (kx - \alpha_1)(kx - \alpha_2)\ldots(kx - \alpha_n) \\ &= k^n \left(x - \frac{\alpha_1}{k}\right)\left(x - \frac{\alpha_2}{k}\right)\ldots\left(x - \frac{\alpha_n}{k}\right) = 0. \end{aligned}$$

Разделив на k^n, получаем

$$\begin{aligned} \left(\frac{1}{k^n}\right) p(kx) &= x^n + \left(\frac{1}{k}\right) c_{n-1}x^{n-1} + \cdots + \left(\frac{1}{k^{n-1}}\right) c_1 x + \left(\frac{1}{k^n}\right) c_0 \\ &= \left(x - \frac{\alpha_1}{k}\right)\left(x - \frac{\alpha_2}{k}\right)\ldots\left(x - \frac{\alpha_n}{k}\right) = 0. \end{aligned} \tag{M1}$$

Поэтому после растяжения вида $x := kx$ или $x := x/k$ $(k \neq 0)$ корни преобразованного полиномиального уравнения будут соответственно разделены или умножены на k. [Заметим, что если $k = b > 0$, где b — верхняя граница абсолютных значений корней полинома $p(x)$, то все корни полинома $p(bx) = p_s(x) = 0$ будут находиться внутри единичного круга.]

Наконец, если имеет место инверсия, то преобразованное полиномиальное уравнение имеет вид[1)]

$$\begin{aligned} p\left(\frac{1}{x}\right) = p_i(x) &= \frac{1}{x^n} + \frac{c_{n-1}}{x^{n-1}} + \cdots + \frac{c_1}{x} + c_0 \\ &= \left(\frac{1}{x^n}\right)(1 - x\alpha_1)(1 - x\alpha_2)\ldots(1 - x\alpha_n) \\ &= \left[\frac{(-1)^n}{x^n}\right]\left(x - \frac{1}{\alpha_1}\right)\left(x - \frac{1}{\alpha_2}\right)\ldots\left(x - \frac{1}{\alpha_n}\right) = 0. \end{aligned}$$

Умножая на x^n, получаем

$$\begin{aligned} x^n p\left(\frac{1}{x}\right) &= c_0 x^n + c_1 x^{n-1} + \cdots + c_{n-1}x + 1 \\ &= \left(x - \frac{1}{\alpha_1}\right)\left(x - \frac{1}{\alpha_2}\right)\ldots\left(x - \frac{1}{\alpha_n}\right) = 0, \end{aligned} \tag{M2}$$

[2)] Где индекс s — от $s$tratching (растяжение). — *Прим. перев.*

[1)] Где индекс i — от $i$version (инверсия). — *Прим. перев.*

откуда видно, что с помощью инверсии мы получаем преобразованное полиномиальное уравнение, корни которого мультипликативно обратны корням исходного полиномиального уравнения.

Рассмотрев, как действуют порождающие подстановки на корнях полиномиального уравнения от одной неизвестной, мы теперь кратко обсудим, как эти подстановки фактически выполняются.

Сдвиг полиномиального уравнения представляет наибольший интерес. Аналитически он может быть получен с помощью формулы разложения Тейлора $p(\alpha + x) = \sum_{0 \le k \le n} [p^{(k)}(\alpha)/k!] x^k$. Если $p(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + c_1 x + c_0 = 0$ и вместо x подставляется $\alpha + x$, то коэффициенты преобразованного полинома суть

$$b_k = \frac{p^{(k)}(\alpha)}{k!}, \quad k = 0, 1, \ldots, n,$$

и могут быть получены из формулы

$$b_k = \sum_{k \le j \le n} \binom{j}{k} \alpha^{j-k} c_j, \quad k = 0, 1, \ldots, n.$$

Эти вычисления несколько упрощаются при $\alpha = 1$, но даже в этом случае требуемое количество вычислений легко может обескуражить. К счастью, мы видели, что существует метод Руффини–Горнера [см. разд. 3.1.2 и (Cajori, 1911)].

Инверсию очень просто реализовать. В силу (M2) легко видеть, что для того, чтобы выполнить подстановку $x := 1/x$, достаточно изменить на обратный порядок коэффициентов полинома.

Растяжение может достигаться [см. также (M1)] масштабированием коэффициентов степенями k, начиная с коэффициента, соответствующего второй по старшинству степени.

7.3.3. Теорема Винсента: расширение и приложения

В этом разделе мы обсуждаем теорему Винсента 1836 г., которая является основой обоих методов цепных дробей отделения вещественных корней уравнения.

Мы начинаем с тщательного исследования правила о знаках Кардано–Декарта (теорема 7.2.6), утверждающего, что число p положительных корней полиномиального уравнения $p(x) = 0$ не может превосходить числа v перемен знаков в последовательности его коэффициентов, и если $n = v - p > 0$, то n — четное число.

Более внимательное рассмотрение теоремы 7.2.6 показывает, что это довольно слабое предложение; оно дает точное число положительных корней полиномиального уравнения $p(x) = 0$ только в следующих двух частных случаях [ниже, когда мы допускаем терминологическую вольность и говорим «перемена знаков», мы имеем в виду перемены знаков в последовательности коэффициентов полинома $p(x)$]:

i. Если нет перемен знаков, то нет и положительных корней.

ii. Если имеется одна перемена знаков, то имеется один положительный корень.

Как мы увидим впоследствии, эти два частных случая играют важную роль. Более того, справедливо также утверждение, обратное к **i**, поскольку имеет место следующая

Лемма 7.3.6 (Стодола). Если полиномиальное уравнение

$$p(x) = c_0 x^d + c_1 x^{d-1} + \cdots + c_d = 0 \qquad (c_0 > 0)$$

с вещественными коэффициентами c_j, $j = 0, 1, 2, \ldots, d$, имеет только корни с отрицательными вещественными частями, то все его коэффициенты положительны и, следовательно, не дают перемен знаков.

Доказательство. Пусть $-\alpha_n$, $n = 1, 2, \ldots, k$, — вещественные, а $-\gamma_m \pm i\delta_m$, $m = 1, 2, \ldots, s$, — комплексные корни уравнения $p(x) = 0$, где по предположению α_n и $\gamma_m > 0$ для всех n и m. Полином $p(x)$ может быть выражен в виде произведения $c_0, (x+\alpha)$ и $[(x+\gamma_m)^2 + \delta_m^2]$ для всех $n = 1, \ldots, k$ и $m = 1, \ldots, s$. Однако все коэффициенты этих сомножителей положительны, и, следовательно, коэффициенты произведения также будут все положительны, и, таким образом, перемены знаков отсутствуют. □

Относительно второго частного случая теоремы 7.2.6 (случай **ii**, упомянутый выше) заметим, что его обращение в общем случае неверно, в чем можно убедиться на примере полиномиального уравнения $x^3 - x^2 - 1 = (x-1)(x-i)(x+i) = 0$, у которого имеется один положительный корень, но три перемены знаков. Однако при более ограничительных условиях обратное также верно; формально это выражается следующим образом:

Лемма 7.3.7 (Akritas–Danielopoulos, 1985). Пусть $p(x) = 0$ — полиномиальное уравнение степени $n > 1$ с вещественными коэффициентами без кратных корней, имеющее один положительный корень $\xi \neq 0$ и $n-1$ корней $\xi_1, \xi_2, \ldots, \xi_{n-1}$ с отрицательными вещественными

частями (комплексные корни появляются попарно сопряженными) и такое, что его корни могут быть выражены в виде

$$\xi_j = -(1+\alpha_j), \qquad j = 1, 2, \ldots, n-1,$$

с $|\alpha_j| < \varepsilon_n$, где

$$\varepsilon_n = \left(1 + \frac{1}{n}\right)^{1/(n-1)} - 1.$$

Тогда $p(x)$ в форме распределения по степеням содержит ровно одну перемену знаков в последовательности своих коэффициентов.

Доказательство. С точностью до постоянного множителя полином $p(x)$ может быть записан в виде

$$\begin{aligned} p(x) &= (x-\xi)(x-\xi_1)\ldots(x-\xi_{n-1}) \\ &= (x-\xi)(x+1+\alpha_1)\ldots(x+1+\alpha_{n-1}) \\ &= (x-\xi)(x^{n-1} + c_1 x^{n-2} + \cdots + c_{n-1}), \end{aligned} \tag{AD1}$$

где

$$c_k = \sum (1+\alpha_1)(1+\alpha_2)\ldots(1+\alpha_k)$$

— сумма, состоящая из $\binom{n-1}{k}$ членов ($k \le n-1$). Ясно, что (AD1) можно переписать в виде

$$p(x) = x^n + (c_1 - \xi)x^{n-1} + (c_2 - c_1\xi)x^{n-2} + \cdots + (c_{n-1} - c_{n-2}\xi)x - c_{n-1}\xi.$$

Если $c_k > 0$, $k = 1, 2, \ldots, n-1$, и отношение c_k/c_{k-1}, где $c_0 = 1$, уменьшается с увеличением k, то очевидно, что $p(x)$ имеет ровно одну перемену знаков. Чтобы показать, что $c_k > 0$, $k = 1, 2, \ldots, n-1$, заметим, что для каждого из $\binom{n-1}{k}$ членов мы имеем

$$|(1+\alpha_1)(1+\alpha_2)\ldots(1+\alpha_k) - 1| \le (1+|\alpha_1|)(1+|\alpha_2|)\ldots(1+|\alpha_k|) - 1,$$

а поскольку по предположению $|\alpha_j| < \varepsilon_n$ для $j = 1, 2, \ldots, n-1$, мы получаем

$$(1+|\alpha_1|)(1+|\alpha_2|)\ldots(1+|\alpha_k|) - 1 \le (1+\varepsilon_n)^k - 1 \le (1+\varepsilon_n)^{n-1} - 1 = \frac{1}{n}.$$

Поэтому можно написать

$$c_k = \binom{n-1}{k}(1+\delta_k), \tag{AD2}$$

где $|\delta_k| \le 1/n$ и, следовательно $c_k > 0$, $k = 1, 2, \ldots, n-1$.

Затем нам нужно показать, что отношение c_k/c_{k-1} уменьшается с увеличением k, иными словами, $c_k/c_{k-1} > c_{k+1}/c_k$, $k = 1, 2, \ldots, n-1$. Однако это тривиально, поскольку, пользуясь (AD2), мы получаем

$$\frac{c_k}{c_{k-1}} = \left(\frac{n-k}{k}\right) \cdot \left(\frac{1+\delta_k}{1+\delta_{k-1}}\right)$$

и

$$\frac{c_{k+1}}{c_k} = \left(\frac{n-k-1}{k+1}\right) \cdot \left(\frac{1+\delta_{k+1}}{1+\delta_k}\right),$$

и нам нужно теперь доказать, что

$$\frac{k(n-k-1)}{(k+1)(n-k)} < \frac{(1+\delta_k)^2}{(1+\delta_{k-1})(1+\delta_{k+1})}.$$

Действительно, это неравенство справедливо, поскольку, с одной стороны,

$$\frac{k(n-k-1)}{(k+1)(n-k)} = 1 - \frac{n}{(k+1)(n-k)} \le 1 - \frac{4n}{(n+1)^2} = \frac{(n-1)^2}{(n+1)^2}$$

и, с другой стороны,

$$\frac{(1+\delta_k)^2}{(1+\delta_{k-1})(1+\delta_{k+1})} > \frac{(1-1/n)^2}{(1+1/n)^2} = \frac{(n-1)^2}{(n+1)^2}.$$

Таким образом, так как $c_k > 0$, $k = 1, 2, \ldots, n-1$, и отношение c_k/c_{k-1} уменьшается с увеличением k, мы доказали теорему. □

Тщательно проанализировав два частных случая теоремы 7.2.6, мы можем теперь сформулировать теорему Винсента, существенно от них зависящую [см. также историческое замечание 5, (Akritas et al., 1978; Lloyd, 1979; Poggendorff, 1863)].

Теорема 7.3.8 (Vincent, 1836). Если в полиномиальном уравнении с рациональными коэффициентами и без кратных корней сделать последовательно подстановки вида

$$x := a_1 + \frac{1}{x'}, \quad x' := a_2 + \frac{1}{x''}, \quad x'' := a_3 + \frac{1}{x'''}, \ldots$$

где a_1 — произвольное неотрицательное целое число, а $a_2, a_3, \ldots$ — любые положительные целые числа, то получающееся в результате

уравнение либо не имеет перемен знаков, либо имеет одну перемену знаков. В последнем случае уравнение имеет единственный положительный корень, представляемый цепной дробью

$$a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ddots}}}$$

в то время как в первом случае корней нет.

Доказательство. Доказательство этой теоремы можно найти в оригинальной работе Винсента, и здесь оно опускается. Вместо этого непосредственно ниже мы представляем доказательство более общей теоремы. □

Очевидно, что эта теорема обрабатывает только положительные корни; отрицательные корни исследуют, заменяя x на $-x$ в исходном полиномиальном уравнении. Общность этой теоремы не ограничивается предположением, что не должно быть кратных корней, потому что, как и в случае теоремы Штурма, мы можем сначала применить разложение на свободные от квадратов множители. Сам Винсент говорил, что теорему 7.3.8 сформулировал в 1827 г. Фурье, но он не дал какого-либо ее доказательства (или если и дал, то оно никогда не было найдено); более того, Лагранж использовал главную идею этой теоремы гораздо раньше.

Зависимость теоремы Винсента от теоремы Бюдана легко заметить, если каждую подстановку вида $x := a_i + 1/x$ заменить эквивалентной парой подстановок $\{x := a_i + x,\ x := 1/x\}$.

Интуитивно цель ряда последовательных подстановок вида $x := a_i + 1/x$, применяемых к уравнению $p(x) = 0$, состоит в том, чтобы один из его положительных корней оказался внутри интервала $(0, 1)$, а все остальные — в $(1, \infty)$, или наоборот, за исключением, конечно, случая, когда 1 — корень. В первом случае последующая подстановка $x := 1/(1 + x)$ даст в результате уравнение с одним только корнем в $(0, \infty)$, в то время как во втором случае тот же результат достигается с помощью подстановки $x := 1 + x$.

В теореме 7.3.8 естественно возникает вопрос относительно максимума числа подстановок вида $x := a_i + 1/x$, необходимых, чтобы получить полиномиальное уравнение с не более чем одной переменой знаков. Ответ дается следующей теоремой (см. историческое замечание 6).

Теорема 7.3.9 (Винсент–Успенский–Акритас). Пусть $p(x) = 0$ — полиномиальное уравнение степени $n > 1$ с рациональными коэффициентами и без кратных корней, и пусть $\Delta > 0$ — наименьшее расстояние между любыми двумя из его корней. Пусть m — наименьший индекс, такой, что

$$F_{m-1}\frac{\Delta}{2} > 1, \quad F_{m-1}F_m\Delta > 1 + \frac{1}{\varepsilon_n}, \tag{V1}$$

где F_k есть k-й элемент последовательности Фибоначчи 1, 1, 2, 3, 5, 8, 13, 21, ... и

$$\varepsilon_n = \left(1 + \frac{1}{n}\right)^{1/(n-1)} - 1. \tag{V2}$$

Пусть a_1 — произвольное неотрицательное целое число, и пусть $a_2, \ldots, a_m$ — произвольные положительные целые числа. Тогда подстановка

$$x := a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \cfrac{1}{a_m + \cfrac{1}{\xi}}}} \tag{V3}$$

(которая эквивалентна ряду последовательных подстановок вида $x := a_i + 1/\xi$, $i = 1, 2, \ldots, m$) преобразует уравнение $p(x) = 0$ в уравнение $p_{ti}(\xi) = 0$, которое имеет не более одной перемены знаков.

Доказательство. Чтобы доказать теорему, достаточно показать, что после последовательных подстановок вида $x := a_i + 1/\xi$ вещественные части всех комплексных корней, также как все вещественные корни, за исключением, может быть, одного, становятся отрицательными. (Чрезвычайно важно отметить, что корни преобразованного уравнения *группируются вокруг* -1.)

Действительно, пусть p_k/q_k есть k-я подходящая дробь для цепной дроби

$$a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ddots}}}$$

Из разд. 2.2.3 нам известно, что для $k \geq 0$, $p_0 = 1$, $p_{-1} = 0$, $q_0 = 0$ и $q_{-1} = 1$ мы имеем

$$p_{k+1} := a_{k+1}p_k + p_{k-1},$$
$$q_{k+1} := a_{k+1}q_k + q_{k-1}.$$

Так как $q_1 = 1$ и $q_2 = a_2 \geq 1$, то, следовательно, $q_k \geq F_k$. Далее, (V3) может быть выражено в виде

$$x = \frac{p_m\xi + p_{m-1}}{q_m\xi + q_{m-1}},$$

откуда следует, что

$$\xi = -\left(\frac{p_{m-1} - q_{m-1}x}{p_m - q_m x}\right). \tag{V4}$$

Ясно, что если x_0 — любой корень уравнения $p(x) = 0$, то величина ξ_0, определяемая формулой (V4), является соответствующим корнем преобразованного уравнения $p_{ti}(\xi) = 0$.

a. Предположим, что x_0 — комплексный корень уравнения $p(x) = 0$, т.е. $x_0 = a \pm ib$, $b \neq 0$. В этом случае вещественная часть соответствующего корня ξ_0 равна

$$rp(\xi_0) = -\left[\frac{(p_{m-1} - q_{m-1}a)(p_m - q_m a) + q_{m-1}q_m b^2}{(p_m - q_m a)^2 + q_m^2 b^2}\right] \tag{V5}$$

и обязательно отрицательна, если $(p_{m-1} - q_{m-1}a)(p_m - q_m a) \geq 0$. Если, наоборот, $(p_{m-1} - q_{m-1}a)(p_m - q_m a) < 0$, то значение a, очевидно, содержится между двумя последовательными подходящими дробями p_{m-1}/q_{m-1} и p_m/q_m, абсолютное значение разности между которыми равно $1/q_{m-1}q_m$. Следовательно,

$$\left|\frac{p_{m-1}}{q_{m-1}} - a\right| < \frac{1}{q_{m-1}q_m} \quad \text{и} \quad \left|\frac{p_m}{q_m} - a\right| < \frac{1}{q_{m-1}q_m},$$

откуда следует, что

$$|(p_{m-1} - q_{m-1}a)(p_m - q_m a)| < \frac{1}{q_{m-1}q_m} \leq 1. \tag{V6}$$

Из (V5) и (V6) мы заключаем, что значение $rp(\xi_0)$ отрицательно, если $q_{m-1}q_m b^2 > 1$. Чтобы доказать, что это так в нашем случае, прежде всего заметим, что, поскольку Δ — минимальное

расстояние между любыми двумя корнями уравнения $p(x) = 0$, мы имеем

$$|(a + ib) - (a - ib)| = |2ib| = 2|b| \geq \Delta,$$

откуда получаем $|b| \geq \Delta/2$; более того, мы знаем, что $q_m \geq q_{m-1} \geq F_{m-1}$ и, согласно (V1), что $F_{m-1}\Delta/2 > 1$. Тогда, очевидно, $F_{m-1}|b > 1$, откуда следует, что $q_{m-1}|b| > 1$ и $q_m|b| > 1$. Из последних двух неравенств мы получаем $q_{m-1}q_m b^2 > 1$, доказав, таким образом, что $rp(\xi_0) < 0$; это очевидным образом верно для всех комплексных корней преобразованного уравнения $p_{ti}(\xi) = 0$.

b. Предположим теперь, что x_0 — вещественный корень уравнения $p(x) = 0$, и рассмотрим сначала случай, когда для всех вещественных корней x_i

$$(p_{m-1} - q_{m-1}x_i)(p_m - q_m x_i) > 0.$$

Из (V4) следует, что все вещественные корни преобразованного уравнения $p_{ti}(\xi) = 0$ будут отрицательными; более того, по предположению все комплексные корни уравнения $p_{ti}(\xi) = 0$ имеют отрицательные вещественные части. Следовательно, по лемме 7.3.6 $p_{ti}(\xi)$ не имеет перемен знаков.

Предположим, теперь, что для некоторого вещественного корня x_0

$$(p_{m-1} - q_{m-1}x_0)(p_m - q_m x_0) \leq 0. \tag{V7}$$

Тогда, очевидно, x_0 содержится между двумя последовательными подходящими дробями p_{m-1}/q_{m-1} и p_m/q_m, и, следовательно, $|p_m/q_m - x_0| \leq 1/q_{m-1}q_m$. Пусть x_k, $k \neq 0$, — любой другой корень, вещественный или комплексный, уравнения $p(x) = 0$ и ξ_k — соответствующий корень преобразованного уравнения. Тогда, с учетом того, что

$$p_m q_{m-1} - p_{m-1}q_m = (-1)^m,$$

из (V4) следует, что

$$\xi_k + \frac{q_{m-1}}{q_m} = \frac{(-1)^m}{q_m(p_m - q_m x_k)},$$

или

$$\xi_k = -\left(\frac{q_{m-1}}{q_m}\right)\left[1 - \frac{(-1)^m}{q_{m-1}q_m(p_m/q_m - x_k)}\right] = -\left(\frac{q_{m-1}}{q_m}\right)(1+\alpha_k),$$

где

$$\alpha_k = \frac{(-1)^{m-1}}{q_{m-1}q_m(p_m/q_m - x_k)}.$$

Теперь

$$\left|\frac{p_m}{q_m} - x_k\right| = \left|\frac{p_m}{q_m} - x + x - x_k\right| \geq |x - x_k| - \left|\frac{p_m}{q_m} - x\right|$$
$$\geq \Delta - \frac{1}{q_{m-1}q_m} > 0$$

и, следовательно,

$$|\alpha_k| \leq \frac{1}{q_{m-1}q_m\Delta - 1} \leq \frac{1}{F_{m-1}F_m\Delta - 1}.$$

Из последнего выражения и второго неравенства формулы (VI) мы заключаем, что $|\alpha_k| < \varepsilon_n$. Таким образом, корни ξ_k, $k = 1, 2, \ldots, n-1$, преобразованного уравнения $p_{ti}(\xi) = 0$, соответствующие корням x_k, $k = 1, 2, \ldots, n-1$, уравнения $p(x) = 0$, которые все отличны от x_0, имеют вид

$$\xi_k = -\left(\frac{q_{m-1}}{q_m}\right)(1 + \alpha_k), \quad |\alpha_k| < \varepsilon_n, \tag{V8}$$

т.е. корни преобразованного уравнения имеют отрицательные вещественные части и *группируются вокруг* -1. Если сделать подстановку

$$\xi := -\left(\frac{q_{m-1}}{q_m}\right)u, \quad \xi_k := -\left(\frac{q_{m-1}}{q_m}\right)\xi'_k, \quad k = 0, 1, \ldots, n-1,$$

где

$$\xi'_0 > 0, \quad \xi'_k = -(1 + \alpha_k), \quad k = 1, 2, \ldots, n-1,$$

то преобразованный полином $p_{ti}(\xi)$ может быть записан в виде

$$p_{ti}(\xi) = \left(\frac{q_{m-1}}{q_m}\right)^n p_{ti}(u) = c\left(\frac{q_{m-1}}{q_m}\right)^n (u - \xi'_0)\ldots(u - \xi'_{n-1}).$$

Поскольку полином $p_{ti}(u)$ удовлетворяет всем предположениям леммы 7.3.7, он содержит в точности одну перемену знаков, и, очевидно, то же верно для преобразованного полинома $p_{ti}(\xi)$.

Нам осталось рассмотреть теперь только случай, когда в (V7) имеет место равенство, т.е.

$$(p_{m-1} - q_{m-1}x_0)(p_m - q_m x_0) = 0.$$

Если $p_{m-1} - q_{m-1}x_0 = 0$, то из (V4) мы видим, что $\xi_0 = 0$, и, очевидно, у преобразованного уравнения $p_{ti}(\xi) = 0$ нет перемен знаков (лемма 7.3.6). В случае $p_m - q_m x_0 = 0$ мы имеем $\xi_0 = \infty$, и преобразованное уравнение редуцируется к степени $n-1$. Поскольку снова все корни имеют отрицательные вещественные части, мы заключаем из леммы 7.3.6, что $p_{ti}(\xi) = 0$ не имеет перемен знаков. Таким образом, мы полностью доказали теорему. □

Из теоремы 7.3.9 ясно видно, что m — желаемая граница числа подстановок вида $x := a_i + 1/x$, которые должны быть выполнены, чтобы получить уравнение с не более чем одной переменой знаков в последовательности его коэффициентов.

Следствие 7.3.10. В предположениях теоремы 7.3.9

$$m = O\{nL[|p(x)|_\infty] + nL(n)\}.$$

Доказательство. По определению m — наименьший индекс, такой, что одновременно выполняются оба неравенства (V1). Ясно, что одно из этих неравенств (а возможно, оба) не будет выполняться, если мы уменьшим m на единицу; предположим, что не выполняется первое, так что

$$F_{m-2}\frac{\Delta}{2} \le 1. \tag{V9}$$

Применяя соотношение $F_k = \phi^k/\sqrt{5}$ (с правой частью уравнения, округленной до ближайшего целого), где $\phi = 1.618\ldots$, из (V9) получаем, что $\phi^{m-2} \le 2\sqrt{5}\cdot(1/\Delta)$, откуда заключаем, что

$$m \le 2 + \log_\phi 2 + (1/2)\log_\phi 5 - \log_\phi \Delta. \tag{V10}$$

Более того, напомним, что из теоремы 7.2.12 (Mahler, 1964) мы имеем

$$\Delta \ge \sqrt{3}n^{-(n+2)/2}|p(x)|_1^{-(n-1)}. \tag{V11}$$

Если мы объединим (V10) и (V11) и примем во внимание тот факт, что $L[|p(x)|2] \sim L[|p(x)|_\infty]$, то следствие будет доказано. [Тот же результат получается, если предположить, что не выполнено второе неравенство (V1).] □

В большинстве представляющих интерес случаев $L(n) = 1$, и, таким образом, мы можем считать, что

$$m = O\{nL[|p(x)|_\infty]\}. \tag{V12}$$

Теорема 7.3.9 может использоваться для отделения вещественных корней полиномиального уравнения. Чтобы видеть, как она применяется, заметим следующее (для ясности и лучшего понимания мы повторяем некоторые части доказательства теоремы 7.3.9, где ξ теперь заменяется на y):

i. Подстановка цепной дроби (V3) может также быть записана как

$$x := \frac{p_m y + p_{m-1}}{q_m y + q_{m-1}}, \tag{V13}$$

где p_k/q_k есть k-я подходящая дробь для цепной дроби

$$a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ddots}}}$$

и, как уже говорилось, для $k \geq 0$, $p_0 = 1$, $p_{-1} = 0$, $q_0 = 0$ и $q_{-1} = 1$ мы имеем

$$p_{k+1} := a_{k+1}p_k + p_{k-1}, \quad q_{k+1} := a_{k+1}q_k + q_{k-1}. \tag{V14}$$

ii. Расстояние между двумя последовательными подходящими дробями равно $|p_{m-1}/q_{m-1} - p_m/q_m| = 1/q_{m-1}q_m$. Ясно, что наименьшее значение q_i встречается, когда $a_i = 1$ для всех i. Тогда $q_m = F_m$, m-е число Фибоначчи. Это объясняет, почему существует связь между числами Фибоначчи и расстоянием Δ в теореме 7.3.9.

iii. Пусть $p_{ti}(y) = 0$ — уравнение, полученное из $p(x) = 0$ после подстановки вида (V13), соответствующей ряду сдвигов и инверсий. Заметим, что (V13) отображает интервал $0 < y < \infty$ на x-интервал, неупорядоченные концевые точки которого — последовательные подходящие дроби p_{m-1}/q_{m-1} и p_m/q_m. Если длина этого x-интервала меньше, чем Δ, то он содержит не более одного корня уравнения $p(x) = 0$, а соответствующее уравнение $p_{ti}(y) = 0$ имеет не более одного корня в интервале $(0, \infty)$.

iv. Если y' — этот положительный корень уравнения $p_{ti}(y) = 0$, то соответствующий корень x' уравнения $p(x) = 0$ может легко быть получен из (V13). Мы знаем только, что y' находится в интервале $(0, \infty)$; поэтому, подставляя вместо y в (V13) один раз 0, а другой ∞, мы получаем для положительного корня x изолирующий интервал, неупорядоченные концевые точки которого — это p_{m-1}/q_{m-1} и p_m/q_m. Каждому положительному корню соответствует своя цепная дробь; нужно вычислить не более m неполных частных для отделения любого положительного корня. (Как мы уже упоминали, отрицательные корни могут быть отделены, если мы заменим x на $-x$ в исходном уравнении.)

7.3.4. Два метода цепных дробей отделения вещественных корней

Из приведенного обсуждения очевидно, что вычисление неполных частных $a_1, a_2, \ldots, a_m$ для подстановок вида (V3), которые приводят к уравнению ровно с одной переменой знаков, составляет процедуру отделения вещественного корня. [Из теоремы Бюдана нам известно, что значение какого-то неполного частного a_i вычислено, если в последовательности коэффициентов уравнения $p(a_i + x) = 0$ больше перемен знаков, чем у уравнения $p(a_i + 1 + x) = 0$.]

Имеются два метода цепных дробей: Винсента, 1836 г., и автора 1978 г., соответствующие двум различным способам вычисления неполных частных a_i (см. историческое замечание 7). Как мы увидим ниже, различие между этими двумя методами можно рассматривать как аналог различия между интегралами Римана и Лебега. То есть хорошо известно, что сумма $1 + 1 + 1 + 1 + 1$ может вычисляться следующими двумя способами:

i. $1 + 1 = 2,\ 2 + 1 = 3,\ 3 + 1 = 4,\ 4 + 1 = 5$ (Риман).

ii. $5 \cdot 1 = 5$ (Лебег).

В основе метода цепных дробей Винсента 1836 г. лежит вычисление какого-то неполного частного a_i с помощью серии единичных приращений, $a_i := a_i + 1$, с каждым из которых мы должны выполнить сдвиг $p_{ti}(x) := p_{ti}(1 + x)$ [для некоторого полиномиального уравнения $p_{ti}(x) = 0$] и проверить изменение числа перемен знаков. Этот подход «в лоб» приводит к методу с экспоненциальным поведением, и, следовательно, его практическая ценность невелика. Специального алгоритма для этого метода мы не приводим.

В качестве примера метода Винсента отделим корни полиномиального уравнения

$$p(x) = (x - \alpha)(x - \beta) = 0, \tag{A1}$$

где $\alpha = 5 \cdot 10^9 + \varepsilon$ и $\beta = \alpha + 1$. Рассмотрим $a_1^{(\alpha)}$, первое неполное частное для α, которое равно $5 \cdot 10^9$. При использовании метода Винсента мы первоначально полагаем $a_1^{(\alpha)} := 1$, $p_{ti}(x) := p(x)$ и вычисляем $p_{ti}(x) := p_{ti}(1 + x)$. Поскольку число перемен знаков в последовательности коэффициентов преобразованного полинома $p_{ti}(x)$ не изменилось, мы даем приращение $a_1^{(\alpha)} := a_1^{(\alpha)} + 1$, вычисляем новый полином $p_{ti}(x) := p_{ti}(1 + x)$, снова проверяя число перемен знаков. Этот процесс повторяется $5 \cdot 10^9$ раз, и на самой быстрой ЭВМ это займет несколько лет. Однако метод Винсента весьма эффективен, когда значения неполных частных маленькие.

Напротив, метод цепных дробей, разработанный автором в 1978 г., заключается в немедленном вычислении какого-либо неполного частного a_i в качестве *нижней* границы b значений положительных корней некоторого полинома $p_{ti}(x)$, т.е., пользуясь правилом Коши (разд. 7.2.3), мы немедленно определяем b и полагаем $a_i := b$. [Напомним, что вычисление нижней границы b_{lo} (от lower bound — *Перев.*) значений положительных корней некоторого полиномиального уравнения $p(x) = 0$ эквивалентно вычислению верхней границы $b_{lo\text{-inv}}$ значений положительных корней уравнения $p(1/x) = 0$ с последующим обращением, т.е. $b_{lo} := 1/b_{lo\text{-inv}}$.] Положив $a_i := b$, $b \geq 1$, мы должны только выполнить для соответствующего полинома $p_{ti}(x)$ подстановку $x := b + x$, для чего потребуется приблизительно столько же времени, что и для подстановки $x := 1 + x$; поэтому, пользуясь этим методом, мы экономим огромное количество машинного времени, и (A1) решается за несколько секунд. Подробный алгоритм этого метода представлен ниже.

Отметим, что для всех i мы имеем $a_i = \lfloor \alpha_s \rfloor$, где α_s — наименьший положительный корень некоторого полиномиального уравнения. Следовательно, очевидно, что в общем случае, чтобы вычислить $\lfloor \alpha_s \rfloor$, правило Коши применяется несколько раз; например, чтобы вычислить $[5 \cdot 10^9 + \varepsilon]$ для полинома (A1), его нужно применить 18 раз. Однако, поскольку число применений правила Коши очень мало по сравнению со значением a_i и не может быть предопределено, мы можем безопасно считать, что в нашем обсуждении $b = \lfloor \alpha_s \rfloor$.

Эта интерпретация каждого a_i как нижней границы значений положительных корней полинома $p(x) = 0$ становится более понятной, если мы примем во внимание, что наша цель — загнать один из его

положительных корней внутрь интервала $(0,1)$, а все остальные — внутрь интервала $(1,\infty)$ или наоборот. Следующие леммы существенны для дальнейшего.

Лемма 7.3.11. Пусть $p(x)=0$ — полиномиальное уравнение степени $d\geq 2$ от одной неизвестной с целыми коэффициентами и без кратных корней, имеющее p вещественных корней внутри интервала $(0,1)$, $2\leq p\leq d$, и пусть $\Delta_p>0$ — наименьшее расстояние между любыми двумя из этих корней. Тогда инверсия $x:=1/x$, примененная к $p(x)=0$, отображает эти p корней в интервал $(1,\infty)$, где теперь наименьшее расстояние между любыми двумя корнями равно $\Delta'_p>\Delta_p$.

Доказательство. Пусть $0<\alpha_1<\cdots<\alpha_i<\alpha_j<\cdots<\alpha_m<\alpha_n<\cdots<\alpha_p<1$ суть p корней уравнения $p(x)=0$ внутри интервала $(0,1)$, и предположим, что $\Delta_p=\alpha_j-\alpha_i$, в то время как $\Delta'_p=1/\alpha_m-1/\alpha_n$. Поскольку $\Delta'_p=(\alpha_n-\alpha_m)/\alpha_m\alpha_n>\alpha_n-\alpha_m\geq\alpha_j-\alpha_i=\Delta_p$, лемма доказана. □

Лемма 7.3.12. Пусть $p(x)=0$ — полиномиальное уравнение степени $d\geq 2$ от одной неизвестной с целыми коэффициентами и без кратных корней, имеющее два комплексно-сопряженных корня α_1 и α_2 внутри круга с центром $(1/2,0)$ и радиусом $1/2$; более того, пусть $\delta_p=|\alpha_1-\alpha_2|$. Тогда инверсия $x:=1/x$, примененная к уравнению $p(x)=0$, отображает α_1 и α_2 в полуплоскость с вещественной частью >1, где теперь расстояние между ними $\delta'_p>\delta_p$.

Доказательство аналогично предыдущему. □

Ниже мы продолжаем уточнять различия между двумя методами цепных дробей для отделения вещественных корней полиномиального уравнения.

Рассмотрим бесконечное двоичное дерево, с каждой вершиной которого мы ассоциируем тройку вида $\{p_{\rm M}(x),\mathbf{M}(x),v_{\rm M}\}$, где полином $p_{\rm M}(x)$ получен из исходного полинома $p(x)$ подстановкой $x:=\mathbf{M}(x)=(a_1x+a_0)/(b_1x+b_0)$ и $v_{\rm M}$ — число перемен знаков в последовательности коэффициентов полинома $p_{\rm M}(x)$. [Необходимо ассоциировать с каждым преобразованным полиномом соответствующую функцию $\mathbf{M}(x)$ так, чтобы в конце мы могли легко получить изолирующие интервалы корней; см. также (V13).] Если $p(x)=0$ — исходное полиномиальное уравнение с v переменами знаков в последовательности коэффициентов, то корень дерева соответствует тройке $\{p_{\rm M}(x):=p(x),\ \mathbf{M}(x):=x,\ v_{\rm M}:=v\}$.

Путь от каждой вершины к правому потомку соответствует подстановке $x := 1+x$, в то время как путь к левому потомку соответствует подстановке $x := 1/(1+x)$; отметим, что для любого неполного частного a_i ряд a_i последовательных подстановок вида $x := 1+x$, за которыми следует $x := 1/(1+x)$, эквивалентен подстановке $x := a_i + 1/x$, за которой следует $x := 1+x$. Все вершины, принадлежащие какому-либо пути, конечному или бесконечному, будут рассматриваться как элементы непересекающихся множеств трех типов. Множество типа V_0, V_1 или V_n содержит вершины, соответствующие полиномам с нулем, одной или несколькими переменами знаков соответственно. Множества типов V_0 или V_1 называются *терминальными множествами*. В случае множеств, принадлежащих одному и тому же пути, говорят, что множество X *предшествует* множеству Y, если и только если для всех x в X и всех y в Y длина_пути$(x)<$длина_пути(y). В терминальном множестве вершина, связанная кратчайшим путем с корнем, будет называться *терминальной вершиной* или *терминальным узлом*.

С учетом этих определений различие между двумя методами цепных дробей для отделения вещественных корней полиномиального уравнения показано на рис. 7.3.1.

Рис. 7.3.1.

Геометрическая интерпретация двух различных способов вычисления значения некоторого a_i (длины ветви).

Гипотеза 7.3.13. Пусть $p(x) = c_n x^n + \cdots + c_1 x + c_0 = 0$ — полиномиальное уравнение степени $n > 1$ с рациональными коэффициентами и без кратных корней, соответствующее корню двоичного дерева.

Предположим, кроме того, что подстановка

$$x := a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \cfrac{1}{a_h + \cfrac{1}{y}}}}$$

где a_1 — произвольное неотрицательное целое число, а $a_2, \ldots, a_h$, $1 \le h \le m$, — положительные целые элементы (где m определено условием (V1) в теореме 7.3.9), преобразует уравнение $p(x) = 0$ в новое уравнение, соответствующее терминальному узлу типа V_0 или V_1. Тогда для всех k, $1 \le k \le h$, мы имеем

$$a_k = O[|p(x)|_\infty]. \tag{A2}$$

Обсуждение. Экспоненциальность методов непрерывных дробей обусловлена одной или обеими из следующих двух причин: (a) *числом* подстановок вида $x \leftarrow x + 1$, которые нужно выполнить для вычисления неполного частного a_i, и (b) увеличением *размера* целых коэффициентов полиномов, получающихся после подстановок вида $x \leftarrow x + a_i$.

Метод непрерывных дробей Винсента 1836 г. экспоненциален как по (a), так и по (b), в то время как метод непрерывных дробей автора 1978 г. *теоретически* экспоненциален только из-за возрастания размера целых коэффициентов, т.е., пользуясь правилом Коши, автору удалось исключить экспоненциальность, обусловленную фактором (a). Однако по теореме Кузьмина (упоминаемой в разд. 2.2.4) мы не можем теоретически ограничить размер неполных частных a_i и, следовательно, не можем контролировать рост коэффициентов. Тем не менее, благодаря тому, что (для того, чтобы изолировать корни) мы вычисляем очень мало неполных частных a_i, снова пользуясь теоремой Кузьмина, мы видим, что вероятность больших неполных частных практически равна нулю, и, следовательно, наша гипотеза хорошо обоснована.

Суммируя сказанное выше, мы получаем следующий алгоритм, который является единственным алгоритмом с полиномиальным временем работы, использующим цепные дроби.

ACF1978. Метод цепных дробей 1978 г. для отделения вещественных корней уравнения (**C**ontinued **F**ractions Method of **1978** for Isolation of the Real Roots of an Equation)

Вход: $p(x) = 0$, полиномиальное уравнение с целыми коэффициентами и без кратных корней.

Выход: Изолирующие интервалы вещественных корней полинома $p(x)$ или точные значения корней.

1. [Инициализация] Положить $p_w(x) := p(x)$; если $p_w(0) = 0$, то выдать замкнутый интервал $[0, 0]$ и положить $p_w(x) := p_w(x)/x$; pn-flag $:= 0$; ti-flag $:= 0$; $T := \emptyset$ и вычислить число v перемен знаков в последовательности коэффициентов полинома $p_w(x)$. [T — множество определенных выше троек $\{p_{\mathrm{M}}(x), \mathbf{M}(x), v_{\mathrm{M}}\}$, где $\mathbf{M}(x) = (a_1x + a_0)/(b_1x + b_0)$. Если pn-flag $= 0$, то мы отделяем положительные корни, а если pn-flag $= 1$, то отделяем отрицательные; ti-flag — флаг сдвига–инверсии.]

2. [$v = 0$ или $v = 1$?] Если $v = 0$ или $v = 1$, то из правила знаков Кардано–Декарта (теорема 7.2.6) мы знаем, что у $p_w(x)$ нет положительных корней или есть ровно один положительный корень соответственно; в последнем случае $(0, \infty)$ — его изолирующий интервал — подается на выход. В любом из этих случаев подстановки не нужны; если pn-flag $= 0$, то перейти к шагу 10, иначе выход.

2a. [$v > 1$] В этой точке нам известно, что $v > 1$, и $p_w(x)$ требует дальнейшего исследования. Положить $p_{\mathrm{M}}(x) := p_w(x)$, $\mathbf{M}(x) := x$, $v_{\mathrm{M}} := v$ и перейти к шагу 4.

3. [Проверка завершения] Если $T \neq \emptyset$, то удалить первую тройку $\{p_{\mathrm{M}}(x), \mathbf{M}(x), v_{\mathrm{M}}\}$ и перейти к шагу 4; если $T = \emptyset$ и pn-flag $= 0$, то перейти к шагу 1, иначе выход.

4. [Вычисление b] Пользуясь правилом Коши (разд. 7.2.3) вычислить нижнюю границу b значений положительных корней полинома $p_{\mathrm{M}}(x)$; если $b < 1$, то перейти к шагу 6.

5. [$x := b + x$, $b \geq 1$] Положить $p_{\mathrm{M}}(x) := p_{\mathrm{M}}(b+x)$; $\mathbf{M}(x) := \mathbf{M}(b+x)$ (v_{M} не меняется). Если $p_{\mathrm{M}}(0) = 0$, то мы нашли (рациональный) корень исходного уравнения, в этом случае вывести замкнутый интервал $[a_0/b_0, a_0/b_0]$ [полученный из соответствующего полинома $\mathbf{M}(x)$] и положить $p_{\mathrm{M}}(x) := p_{\mathrm{M}}(x)/x$.

6. [$x := 1 + x$] Положить $v' := v_{\mathrm{M}}$; $p_{\mathrm{M}1}(x) := p_{\mathrm{M}}(1 + x)$; $\mathbf{M}1(x) := \mathbf{M}(1 + x)$. Если $p_{\mathrm{M}1}(0) = 0$, то мы нашли (рациональный) корень исходного уравнения, в этом случае вывести замкнутый

интервал $[a_0/b_0, a_0/b_0]$ [полученный из соответствующего полинома $\mathbf{M}(x)$] и положить $p_{\mathrm{M}1}(x) := p_{\mathrm{M}1}(x)/x$. Перейти к шагу 8.

7. [$x := 1/(1+x)$] Если $v' = v_{\mathrm{M}1}$, то выполнять $\{ti\text{-flag} := 0$; перейти к шагу 3$\}$. В этой точке нам известно, что $v' \neq v_{\mathrm{M}1}$, а это означает наличие корней уравнения $p_{\mathrm{M}}(x) = 0$ в интервале $(0, 1)$; положить $p_i(x) := p_{\mathrm{M}}(1/x)$, обеспечивая условие $\mathrm{lc}[p_i(x)] > 0$ [в случае необходимости умножить $p_i(x)$ на (-1)]; $\mathbf{M}_i(x) := \mathbf{M}(1/x)$; $p_{\mathrm{M}1}(x) := p_i(1+x)$; $\mathbf{M}1(x) := \mathbf{M}_i(1+x)$.

8. [Изменить ti-flag] ti-flag := ti-flag + 1; вычислить $v_{\mathrm{M}1}$, число перемен знаков в последовательности коэффициентов полинома $p_{\mathrm{M}1}(x)$. Как на шаге 2, если $v_{\mathrm{M}1} = 0$ или $v_{\mathrm{M}1} = 1$, то из правила знаков Кардано–Декарта мы знаем, что у $p_{\mathrm{M}1}(x)$ или нет положительных корней, или есть один положительный корень; в последнем случае вывести его изолирующий интервал, равный (i) $(0, a_0/b_0)$, если $a_1 = 0$ и $b_1 > 0$, (ii) $(a_0/b_0, \infty)$, если $a_1 > 0$ и $b_1 = 0$, и (iii) открытому интервалу с неупорядоченными концевыми точками a_0/b_0, a_1/b_1 в остальных случаях [полученному, конечно, во всех случаях по соответствующей функции $\mathbf{M}1(x)$]. Если $v_{\mathrm{M}1} > 1$, то добавить тройку $\{p_{\mathrm{M}1}(x), \mathbf{M}1(x), v_{\mathrm{M}1}\}$ к множеству T.

9. [Возврат к циклу] Если ti-flag $= 1$, то перейти к шагу 7; иначе выполнять $\{ti$-flag := 0; перейти к шагу 3$\}$.

10. [Отделить отрицательные корни] Если $p(x) \neq p(-x)$, то выполнять { положить pn-flag := 1; ti-flag := 0; $p_w(x) := p(-x)$, так что отрицательные корни становятся положительными; $T := \emptyset$; вычислить число v вариаций знаков в последовательности коэффициентов полинома $p_w(x)$ и перейти к шагу 2$\}$, иначе выход. (Если мы выходим здесь, то отрицательные корни симметричны положительным и мы уже знаем их изолирующие интервалы. Конечно, в этом случае интервалы, полученные нами для отрицательных корней, находятся в положительной полуплоскости и должны быть отображены на соответствующие интервалы в отрицательной полуплоскости — тривиальное действие.)

Анализ времени работы алгоритма ***ACF1978***. Из разд. 3.1.2 нам известно, что для данного полинома $p(x)$, $\deg[p(x)] = n$, подстановка $p(x) := p(a+x)$ выполняется за время

$$O\{n^3 L^2(a) + n^2 L(a) L[|p(x)|_\infty]\}.$$

Из (V12) нам известно, что для каждого вещественного корня уравнения $p(x) = 0$ мы должны выполнить не более m таких подстановок, где

$$m = O\{nL[|p(x)|_\infty]\}.$$

Кроме того, из условия (A2) гипотезы 7.3.13 нам известно, что для каждого вещественного корня полинома $p(x)$ и для каждой подстановки $p(x) := p(a + x)$

$$a = O[L(|p(x)|_\infty)].$$

Комбинируя эти три результата, мы получаем, что один вещественный корень полинома $p(x)$ может быть изолирован за время

$$O\{n^4 L^3[|p(x)|_\infty]\}.$$

Поскольку полином $p(x)$ имеет не более n вещественных корней, то

$$t_{\mathbf{ACF1978}}[p(x)] = O\{n^5 L^3[|p(x)|_\infty]\}.$$

Теоретически и эмпирически продемонстрировано, что использующий точную целочисленную арифметику метод цепных дробей 1978 г. является самым быстрым методом отделения вещественных корней полиномиального уравнения; см. также разд. 7.4.

Пример. Отделим вещественные корни уравнения $p(x) = x^3 - 7x + 7 = 0$, где $p(0) \neq 0$. Применяя алгоритм **ACF1978**, получаем следующие результаты:

Шаг 1. Здесь мы полагаем $p_w(x) := x^3 - 7x + 7$, pn-flag $:= 0$, ti-flag $:= 0$, $T := \emptyset$ и вычисляем значение v, равное 2. (Мы сначала отделяем положительные корни.)

Шаг 2а. Поскольку $v > 1$, мы полагаем $p_{\mathbf{M}}(x) := x^3 - 7x + 7$, $\mathbf{M}(x) := x$, $v_{\mathbf{M}} := 2$ и переходим к шагу 4.

Шаг 4. Чтобы использовать правило Коши, мы сначала полагаем $x := 1/x$ в $p_{\mathbf{M}}(x) = 0$ и получаем $7x^3 - 7x^2 + 1 = 0$; затем мы применяем **BPR** к последнему уравнению и получаем $1/b = 1$. Поэтому $b = 1$.

Шаг 5. На этом шаге мы модифицируем $p_{\mathbf{M}}(x) := p_{\mathbf{M}}(1 + x) = x^3 + 3x^2 - 4x + 1$ и $\mathbf{M}(x) := \mathbf{M}(1 + x) = x + 1$; очевидно, $p_{\mathbf{M}}(0) \neq 0$.

Шаг 6. Полагаем $v' := 2$, модифицируем $p_{\mathbf{M}1}(x) := p_{\mathbf{M}}(1 + x) = x^3 + 6x^2 + 5x + 1$ и $\mathbf{M}1(x) := \mathbf{M}(1 + x) = x + 2$ и переходим к шагу 8.

Шаг 8. На этом шаге мы полагаем ti-flag $:= 1$ и вычисляем $v_{\mathbf{M}1}$, равное 0.

Шаг 9. Поскольку ti-flag $= 1$, мы переходим к шагу 7.

Шаг 7. $v' \neq v_{\mathbf{M}1}$, и мы вычисляем новые значения $p_{\mathbf{M}1}(x) = x^3 - x^2 - 2x + 1$ и $\mathbf{M}1(x) = (x+2)/(x+1)$. [Заметим, что $p_{\mathbf{M}1}(x) := p_i(1+x)$ и $\mathbf{M}1(x) := \mathbf{M}_i(1+x)$, где $p_i(x) = x^3 - 4x^2 + 3x + 1$ и $\mathbf{M}_i(x) = (x+1)/x$ мы получили, полагая $x = 1/x$ в $p_{\mathbf{M}}(x)$ и $\mathbf{M}(x)$ соответственно.]

Шаг 8. На этом шаге мы устанавливаем ti-flag $:= 2$ и вычисляем значение $v_{\mathbf{M}1}$, которое оказывается равным 2. В этом случае мы полагаем $T := \{[x^3 - x^2 - 2x + 1, (x+2)/(x+1), 2]\}$.

Шаг 9. Поскольку ti-flag $= 2$, мы устанавливаем ti-flag $:= 0$ и переходим к шагу 3.

Шаг 3. $T \neq \emptyset$, и мы удаляем из него первую тройку $p_{\mathbf{M}}(x) = x^3 - x^2 - 2x + 1, \mathbf{M}(x) = (x+2)/(x+1), v_{\mathbf{M}} = 2$ и переходим к шагу 4.

Шаг 4. Снова мы сначала полагаем $x := 1/x$ в $p_{\mathbf{M}}(x) = 0$ и получаем $x^3 + 2x^2 - x + 1 = 0$; затем мы применяем **BPR** к последнему уравнению и получаем $1/b = 4$. Поэтому $b = 1/4 < 1$, и мы переходим к шагу 6.

Шаг 6. Полагаем $v' := 2$, модифицируем $p_{\mathbf{M}1}(x) := p_{\mathbf{M}}(1+x) = x^3 + 2x^2 - x - 1$ и $\mathbf{M}1(x) := \mathbf{M}(1+x) = (x+3)/(x+2)$ и переходим к шагу 8.

Шаг 8. Здесь мы устанавливаем ti-flag $:= 1$ и вычисляем значение $v_{\mathbf{M}1}$, которое оказывается равным 1. В этом случае мы выводим первый изолирующий интервал $(1, 3/2)$, полученный из $\mathbf{M}1(x) = (x+3)/(x+2)$.

Шаг 9. Поскольку ti-flag $= 1$, мы переходим к шагу 7.

Шаг 7. $v' \neq v_{\mathbf{M}1}$, и мы вычисляем новые значения $p_{\mathbf{M}1}(x) = x^3 + x^2 - 2x - 1$ и $\mathbf{M}1(x) = (2x+3)/(x+2)$.

Шаг 8. Здесь мы устанавливаем ti-flag $:= 2$ и вычисляем значение $v_{\mathbf{M}1}$, которое оказывается равным 1. В этом случае мы выводим второй изолирующий интервал $(3/2, 2)$, полученный из $\mathbf{M}1(x) = (2x+3)/(x+2)$.

Шаг 9. Поскольку ti-flag $= 2$, мы устанавливаем ti-flag $:= 0$ и переходим к шагу 3.

Шаг 3. $T = \emptyset$ и pn-flag $= 0$; поэтому мы переходим к шагу 10.

Шаг 10. Мы теперь отделяем отрицательные корни. Поскольку $p(x) \neq p(-x)$, мы полагаем pn-flag $:= 1$, ti-flag $:= 0$, $p_w(x) := p(-x) = x^3 - 7x - 7$, $T := \emptyset$, вычисляем значение v, которое оказывается равным 1, и переходим к шагу 2.

Шаг 2. Здесь $v = 1$, откуда следует, что изолирующим интервалом для отрицательного корня является $(-\infty, 0)$, и, поскольку

pn-flag $= 1$, мы выходим. (Пользуясь правилом Коши, мы можем получить для этого корня лучшую границу, чем $-\infty$.)

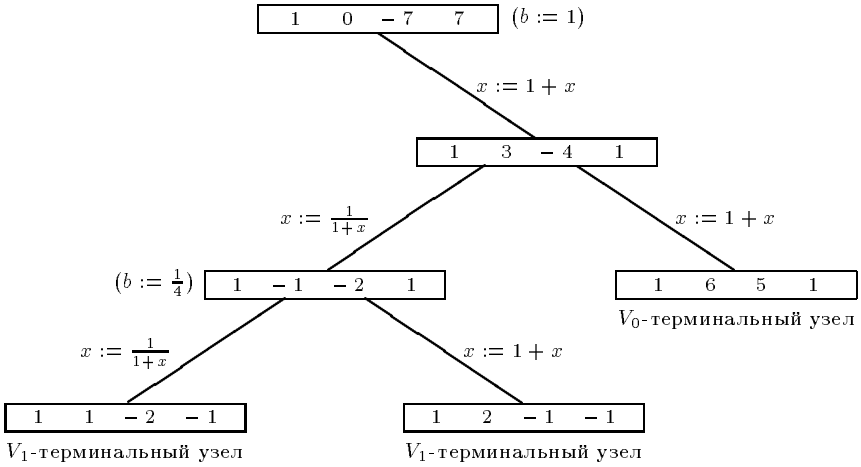


Рис. 7.3.2.

Двоичное дерево, полученное для отделения положительных корней уравнения $x^3 - 7x + 7 = 0$.

Полностью процесс проиллюстрирован на рис. 7.3.2.

7.4

Эмпирическое сравнение двух методов отделения вещественных корней

В разд. 7.2 и 7.3 мы подробно обсуждали два классических способа, метод бисекций Штурма и метод цепных дробей 1978 г., отделения вещественных корней полиномиальных уравнений с целыми коэффициентами. Ниже мы приводим две таблицы, показывающие наблюденные времена вычислений для двух классов полиномов при использовании этих методов. Все времена даны в секундах и были получены с использованием системы компьютерной алгебры **sac-1** на IBM S/370, модель 165. Каждый из полиномов в табл. 7.4.1 был получен путем перемножения соответствующего числа линейных сомножителей. Все коэффициенты полиномов в табл. 7.4.2 были ненулевыми, каждый коэффициент состоял из 10 десятичных цифр, коэффициенты были заданы случайным образом.

Таблица 7.4.1

Полиномы со случайно заданными в интервале $(0, 10^3)$ *корнями*

Степень	Метод цепных дробей 1978 г.	Метод бисекций Штурма
5	0.71	0.73
10	23.22	22.50
15	96.35	151.42
20	288.49	> 600

Таблица 7.4.2

Полиномы со случайно заданными коэффициентами

Степень	Метод цепных дробей 1978 г.	Метод бисекций Штурма
5	0.26	2.05
10	0.46	33.28
15	0.94	156.40
20	2.36	524.42

Таблица 7.4.3

Сравнение различных методов для полиномов из табл. 7.4.2

Степень	ACF1978/ Штурм	Коллинз–Акритас/ Штурм	Дифференцирование/ Штурм
5	0.13	0.28	1.31
10	0.014	0.10	0.55
15	0.004	0.05	0.28
20	0.0045	0.03	0.21

Полиномы в табл. 7.4.2 — те же, что были использованы Коллинзом [в (Rice, 1979)] и Коллинзом и Лоосом (Collins, Loos, 1976) для тестирования метода Коллинза–Акритаса и методов, основанных на

дифференцировании; см. также исторические замечания 1 и 7. Поэтому нетрудно убедиться в преимуществе метода цепных дробей 1978 г. (**ACF1978**) простым сравнением отношений времен различных методов к соответствующим временам, полученным при использовании метода Штурма. Мы приводим эти отношения в табл. 7.4.3. [См. также (Mignotte, 1976).]

7.5

Аппроксимация вещественных корней полиномиального уравнения

В предыдущих разделах мы подробно исследовали два различных подхода к отделению вещественных корней полиномиального уравнения с целыми коэффициентами, т.е. мы может теперь найти вещественные непересекающиеся интервалы, такие, что каждый интервал содержит ровно один вещественный корень и каждый вещественный корень содержится в некотором интервале. Однако, согласно Фурье, это только первый шаг (из двух) вычисления вещественного корня полиномиального уравнения; второй шаг состоит в аппроксимации этих корней с любой желаемой степенью точности ε. Другими словами, этот второй шаг состоит из уменьшения длин изолирующих интервалов, пока они не станут меньше или равны ε.

Ниже мы рассматриваем два метода аппроксимации, первый из которых основан на бисекции, а второй использует цепные дроби вместе с теоремой Винсента; см. (Akritas et al., 1983; Cajori, 1910; Ng, 1980; Nordgaard, 1922; Verbaeten, 1975).

7.5.1. Аппроксимация вещественного корня с использованием бисекции

В этом разделе мы аппроксимируем вещественный корень полиномиального уравнения путем постоянного деления пополам его (данного) изолирующего интервала до тех пор, пока расстояние между концевыми точками интервала не станет меньшим или равным некоторому ε.

Алгоритм бисекции. Дано свободное от квадратов полиномиальное уравнение $p(x) = 0$ с целыми коэффициентами и изолирующий интервал (a, b) корня, который мы хотим аппроксимировать, где a, b — рациональные числа.

Найдем знак полинома $p(x)$ в точке $(a+b)/2$; если значение равно нулю, то заменим (a,b) на $[(a+b)/2,(a+b)/2]$ и завершим работу. Если $p[(a+b)/2]$ имеет тот же знак, что и $p(b)$, то корень находится в интервале $[a,(a+b)/2]$, в то время как если $p[(a+b)/2]$ имеет тот же знак, что и $p(a)$, то корень находится в интервале $[(a+b)/2,b]$. Этот процесс повторяется, пока длина текущего интервала не станет меньше или равной ε.

Из разд. 3.1.2 и 7.2.2 мы можем легко увидеть, что метод бисекций аппроксимирует корень с точностью ε за время

$$O\left\{n^2 L\left(\frac{h}{\varepsilon}\right) L\left(\frac{eh}{\varepsilon}\right) L\left(\frac{eh|p(x)|_\infty}{\varepsilon}\right)\right\}, \tag{B1}$$

где n — степень свободного от квадратов полинома $p(x)$, h — начальная длина изолирующего интервала, ε — точность (предел аппроксимации) и $e=\max\{|a_1|,|a_2|,b_1,b_2\}$, где $(a_1/b_1,a_2/b_2)$ — исходный интервал. Эмпирические результаты показывают, что деление пополам — очень медленный метод; его эффективность значительно повышается, когда он комбинируется с методом Ньютона (см. также табл. 7.6.1 и 7.6.2 разд. 7.6).

Пример. Дано уравнение $p(x)=x^3-7x+7$; аппроксимируем с точностью $\varepsilon=0.01$ его положительный корень, изолирующий интервал которого — это $(1,3/2)$. Заметим, что $p(1)=1$, в то время как $p(3/2)=-1/8$. (Здесь нас фактически интересуют знаки получающихся чисел, а не сами эти числа.)

Заметим прежде всего, что $3/2-1=0.5>\varepsilon$, так что вычисляем знак $p(x)$ в средней точке; а именно, получаем $p(5/4)=13/64$, откуда следует, что меньший изолирующий интервал для этого корня — $(5/4,3/2)$. Продолжая в том же духе, получаем следующую таблицу:

Текущий изолирующий интервал	Расстояние между концевыми точками	Средняя точка	Знак значения $p(x)$ в средней точке	Следующий изолирующий интервал
$(5/4,3/2)$	0.25	11/8	$-13/512$	$(5/4,11/8)$
$(5/4,11/8)$	0.125	21/16	$+301/4096$	$(21/16,11/8)$
$(21/16,11/8)$	0.0625	43/32	$+659/32768$	$(43/32,11/8)$
$(43/32,11/8)$	0.031	87/64	$-953/262144$	$(43/32,87/64)$
$(43/32,87/64)$	0.0156	173/128	$+16757/2097152$	$(173/128,87/64)$
$(173/128,87/64)$	$0.0078<\varepsilon$			

и останавливаемся.

Выражая в десятичном виде концевые точки последнего текущего изолирующего интервала, $(173/128, 87/64)$, мы получаем $173/128 = 1.3515625$ и $87/64 = 1.359375$. Поэтому с точностью $\varepsilon = 0.01$ приближенное значение корня равно 1.35.

7.5.2. Аппроксимация вещественного корня с использованием цепных дробей

Идея аппроксимировать вещественные корни полиномиальных уравнений, пользуясь цепными дробями, принадлежит Лагранжу, который задался целью разработать процедуру, свободную от дефектов, свойственных хорошо известному методу Ньютона. Идея Лагранжа может быть сформулирована следующим образом:

Предположим, что корень полиномиального уравнения $p(x) = 0$ находится между *последовательными целыми числами* a_1 и $a_1 + 1$; уменьшим корни уравнения на a_1, т.е. выполним замену $p(x) := p(a_1 + x)$, и возьмем возвратное уравнение, т.е. выполним подстановку $p(x) := p(1/x)$. Найдем *методом проб* корень последнего уравнения, лежащий между a_2 и $a_2 + 1$, уменьшим корни на a_2 и возьмем возвратное уравнение. Продолжаем таким же образом. Тогда цепная дробь

$$a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ddots}}}$$

аппроксимирует корень уравнения с требуемой точностью.

Ясно, что подход Лагранжа имеет определенные недостатки. Отметим, что он четко направлен, если имеется один и только один корень между последовательными целыми числами a_i и $a_i + 1$; однако процесс не работает, если в интервале $(a_i, a_i + 1)$ имеется два или несколько корней (см. историческое замечание 8). (В этом случае Лагранж фактически использует неопубликованную еще теорему Винсента, чтобы сначала отделить корни.) Более того, так же, как в методе цепных дробей Винсента 1836 г., целая часть корня вычисляется методом проб, т.е., чтобы определить, где находится единственный корень > 1, он подставляет значения $1, 2, 3, \ldots$ в полиномиальное уравнение, выполняя ряд подстановок вида $p(x) := p(1 + x)$, до тех пор, пока не наблюдается изменение знака. Ясно, что это процедура экспоненциальной сложности.

Продолжая исследования в направлении, обозначенном в разд. 7.3, следует заметить, что теорема 7.3.9 может также использоваться для аппроксимации вещественных корней полиномиального уравнения с любой требуемой точностью (см. историческое замечание 9).

Ясно, что это легко можно достигнуть *расширением* (вычислением большего числа неполных частных) цепной дроби (V3), что преобразует исходное полиномиальное уравнение в уравнение с ровно одной переменой знаков в последовательности его коэффициентов (см. историческое замечание 10).

Замечание. Заметим, что описываемый нами метод аппроксимации в значительной степени зависит от процесса отделения, т.е. он *не может* работать, если ему подаются просто изолирующие интервалы корней и полином $p(x) = 0$.

Предположим, что предел аппроксимации равен ε и что в процессе аппроксимации вещественного корня ρ полиномиального уравнения мы вычислили k неполных частных разложения корня в цепную дробь. Тогда из рассуждений разд. 7.3 становится очевидным, что ρ находится между последовательными подходящими дробями p_{k-1}/q_{k-1} и p_k/q_k, абсолютное значение разности которых равно $1/q_{k-1}q_k$. Следовательно,

$$\left|\frac{p_k}{q_k} - \rho\right| \leq \frac{1}{q_{k-1}q_k} \leq \frac{1}{q_{k-1}^2},$$

и метод завершает работу, когда

$$\frac{1}{q_{k-1}^2} \leq \varepsilon \qquad \text{(CF1)}$$

для некоторого k.

Ниже мы описываем два способа вычисления цепной дроби (V3), чтобы аппроксимировать с точностью ε вещественный корень ρ уравнения. Эти два метода имеют одну и ту же теоретическую оценку времени вычислений, но различную эмпирическую эффективность.

Первый способ вычисления цепной дроби (V3) состоит в вычислении каждого дополнительного неполного частного с помощью правила Коши, как описано в разд. 7.3.4. Однако, в основном из-за правила Коши, этот подход неэффективен, в чем можно убедиться, взглянув на табл. 7.6.2 в разд. 7.6. Фактически он даже медленнее метода бисекций, метода, хорошо известного своей неэффективностью.

Второй способ вычисления цепной дроби (V3) — воспользоваться преимуществами специальных свойств полиномов, с которыми мы имеем дело. Эти полиномы специфичны в том смысле, что они имеют одну перемену знаков (и, следовательно, только один положительный корень) и, значит пересекают ось x только один раз. Поэтому целая часть положительного корня (являющаяся следующим неполным частным a_i) может быть вычислена последовательным делением пополам (и вычислением значения в средних точках) интервала $(0, b)$, где b — легко вычислимая верхняя граница значения единственного корня; см. табл. 7.6.2 в разд. 7.6. Вычислить эту верхнюю границу b нам поможет следующая

Теорема 7.5.1. Пусть $p(x) = c_n x^n + \cdots + c_{r+1}x^{r+1} - c_r x^r - \cdots - c_0 = 0$ — полиномиальное уравнение степени n с только одной переменой знаков в последовательности своих целых коэффициентов. Тогда

$$b = \left\{ \frac{\max_{0 \le j \le r} |c_j|}{\sum_{r+1 \le i \le n} c_i} \right\} + 1 \qquad \text{(CF2)}$$

— верхняя граница значения его (единственного) положительного корня.

Доказательство. Для любой степени m переменной x мы имеем

$$x^m = (x-1)(x^{m-1} + x^{m-2} + \cdots + x + 1) + 1.$$

В выражении полинома $p(x)$ мы подставляем это соотношение в каждый член с *положительным* коэффициентом и получаем

$$\begin{aligned} p(x) = c_n(x-1)x^{n-1} + c_n(x-1)x^{n-2} \quad &+ c_n(x-1)\ x^{n-3} + \cdots + c_n \\ + c_{n-1}(x-1)x^{n-2} + c_{n-1}(x-1)\ &x^{n-3} + \cdots + c_{n-1} \\ &\cdots \\ &- c_r x^r - \cdots - c_0. \end{aligned}$$

Рассмотрим последовательные вертикальные столбцы, в которых нет отрицательных коэффициентов. Значение каждого такого столбца положительно при условии, что $x > 1$. Чтобы гарантировать положительные значения последних столбцов, в которых встречается отрицательный коэффициент, мы должны иметь

$$(c_n + c_{n-1} + \cdots + c_{r+1})(x-1) \ge c_i$$

для $0 \le i \le r$, и это доказывает теорему. $\square$

Реализация этой теоремы значительно проще, чем правила Коши, и мы предлагаем читателю в качестве упражнения показать, что вычисления могут быть выполнены за время

$$O\{nL[|p(x)|_\infty] + L^2[|p(x)|_\infty]\}. \tag{CF3}$$

Ниже мы предполагаем существование процедуры **A7.5.1**, которая, пользуясь теоремой 7.5.1 и методом бисекций, вычисляет целую часть a_i единственного положительного корня уравнения $p(x) = 0$ за время

$$t_{\mathbf{A7.5.1}}[p(x)] = O\{n^2 L^3[|p(x)|_\infty]\} \tag{CF4}$$

(см. упражнения по программированию для этого раздела).

APPROX-CF. Аппроксимация вещественного корня уравнения с использованием цепных дробей (**Approx**imate a Real Root of an Equation Using **C**ontinued **F**ractions)

Вход: Предел аппроксимации ε, являющийся рациональным числом, и уравнение $p_{\mathbf{M}}(x) = 0$ с полиномом $\mathbf{M}(x)$, из которого получаем изолирующий интервал для корня; $p_{\mathbf{M}}(x) = 0$ — полиномиальное уравнение степени n с *одной* переменой знаков в последовательности его целых коэффициентов, полученное из исходного уравнения $p(x) = 0$ после подстановки $x := \mathbf{M}(x) = (a_1 x + a_0)/(b_1 x + b_0)$ для некоторых определенных значений a_0, a_1, b_0, b_1. Мы хотим аппроксимировать с точностью ε корень уравнения $p(x) = 0$, расположенный в открытом интервале с неупорядоченными концевыми точками a_1/b_1 и a_0/b_0. Очевидно, что $|a_1/b_1 - a_0/b_0| \geq \varepsilon$, потому что иначе нам нечего делать.

Выход: Изолирующий интервал того корня уравнения $p(x) = 0$, который находится в открытом интервале с концевыми точками e_1 и e_2, где теперь $|e_1 - e_2| \leq \varepsilon$, или точное значение корня уравнения $p(x) = 0$; e_1 и e_2 — два рациональных числа.

1a. [Инициализация] Положить $p_w(x) := p_{\mathbf{M}}(x)$, $\mathbf{M}_w(x) := \mathbf{M}(x)$.

1b. [Уточнение цепной дроби] Пользуясь **A7.5.1**, вычислить a_i — целую часть положительного корня уравнения $p_w(x) = 0$. Если $a_i = 0$, то перейти к шагу 5.

2. [$x := a_i + x$, $a_i > 0$] Модифицировать $p_w(x) := p_w(a_i + x)$ и $\mathbf{M}_w(x) := \mathbf{M}_w(a_i + x)$.

3. [Корень?] Если $p_w(0) = 0$, то положить $e_1 := e_2 := a_0/b_0$, где a_0 и b_0 получены из $\mathbf{M}_w(x)$, вернуть замкнутый интервал $[e_1, e_2]$ и

закончить работу. (Положительный корень, который мы пытались аппроксимировать, — рациональное число.)

4. [Готово?] Если $|a_1/b_1 - a_0/b_0| \leq \varepsilon$, то вернуть открытый интервал с неупорядоченными концевыми точками $e_1 := a_1/b_1$ и $e_2 := a_0/b_0$, где a_0, a_1, b_0 и b_1 получены из $\mathbf{M}_w(x)$, и закончить работу. [Этот интервал аппроксимирует корень полинома $p(x)$ с заданной точностью.]

5. [$x := 1/x$] Положить $p_w(x) := p_w(1/x)$, $\mathbf{M}_w(x) := \mathbf{M}_w(1/x)$ и перейти к шагу 1b.

Анализ времени работы алгоритма ***APPROX-CF***. Отметим, что время выполнения всего алгоритма доминируется временами выполнения шагов 1 и 2.

Для первой итерации алгоритма мы имеем следующее:

Шаг 1 выполняется за время $O\{n^2L^3[|p_{\mathbf{M}}(x)|_\infty]\}$ (см. упражнения по программированию для этого раздела).

Шаг 2 выполняется за время $O\{n^3L^2(a_i) + n^2L(a_i)L[|p_{\mathbf{M}}(x)|_\infty]\}$, что является временем выполнения подстановки $p_w(x) := p_w(a_i + x)$ [и включает в себя подстановку $\mathbf{M}_w(x) := \mathbf{M}_w(a_i + x)$]. Комбинируя это выражение с тем, что $a_i = O[|p_{\mathbf{M}}(x)|_\infty]$ для всех i [см. (A2)], мы получаем $O\{n^3L^2[|p_{\mathbf{M}}(x)|_\infty]\}$, что ограничивается величиной

$$O\{n^3L^3[|p_{\mathbf{M}}(x)|_\infty]\}. \tag{CF5}$$

Сравнивая время вычислений для правила Коши (разд. 7.2.3), (CF3) и (CF4) с (CF5), мы видим, что (CF5) доминирует над временем вычислений шагов 1 и 2.

Для последующих итераций алгоритма (CF5) снова доминирует над временем вычисления шагов 1 и 2, поскольку для любого последующего $p_w(x)$ мы имеем $|p_w(x)|_\infty \sim |p_{\mathbf{M}}(x)|_\infty$ (теорема 7.3.13).

Чтобы найти число итераций i, необходимых для аппроксимации корня с точностью ε, воспользуемся соотношением (CF1), т.е. соотношением $1/q_i^2 \leq \varepsilon$, и следующими фактами:

i. $q_i \leq F_i$, где F_i есть i-й элемент последовательности Фибоначчи.

ii. $F_i = \phi^i/\sqrt{5}$ (округленное до ближайшего целого), где $\phi = 1.618\ldots$.

Очевидно, из (CF1) следует $5/\phi^{2i} \leq \varepsilon$, откуда мы получаем

$$i = O\left(\log_\phi \frac{1}{\varepsilon}\right) = O\left[L\left(\frac{1}{\varepsilon}\right)\right]. \tag{CF6}$$

Перемножая (CF5) и (CF6), получаем

$$t_{\textbf{APPROX-CF}}[p_{\mathbf{M}}(x), \mathbf{M}(x), \varepsilon] = O\{n^3 L(1/\varepsilon) L^3[|p_{\mathbf{M}}(x)|_\infty]\}.$$

Пример. Аппроксимируем с точностью $\varepsilon = 0.01$ положительный корень уравнения $p(x) = x^3 - 7x + 7$, изолирующий интервал которого есть $(1, 3/2)$. Нам даны полином $p_{\mathbf{M}}(x) = x^3 + 2x^2 - x - 1$ и рациональная функция $\mathbf{M}(x) = (x+3)/(x+2)$, с помощью которой мы получили изолирующий интервал; как $p_{\mathbf{M}}(x)$, так и $\mathbf{M}(x)$ были вычислены в примере разд. 7.3.4.

Применяя **APPROX-CF**, получаем следующие результаты:

Шаг 1a. Полагаем $p_w(x) := x^3 + 2x^2 - x - 1$ и $\mathbf{M}_w(x) := (x+3)/(x+2)$.

Шаг 1b. Применение алгоритма **A7.5.1** дает $a_i = 0$, и мы переходим к шагу 5.

Шаг 5. Здесь мы вычисляем новые полиномы $p_w(x) = x^3 + x^2 - 2x - 1$ и $\mathbf{M}_w(x) = (3x + 1)/(2x + 1)$ и переходим к шагу 1b.

Шаг 1b. Применяя **A7.5.1**, получаем $a_i = 1$.

Шаг 2. На этом шаге мы модифицируем полиномы $p_w(x) := p_w(1 + x) = x^3 + 4x^2 + 3x - 1$ и $\mathbf{M}_w(x) := \mathbf{M}_w(1 + x) = (3x + 4)(2x + 3)$.

Шаг 3. Очевидно, что $p_w(0) \neq 0$.

Шаг 4. $|3/2 - 4/3| = 0.169$.

Шаг 5. Здесь мы вычисляем новые значения $p_w(x) = x^3 - 3x^2 - 4x - 1$ и $\mathbf{M}_w(x) = (4x + 3)/(3x + 2)$ и переходим к шагу 1b.

Шаг 1b. Применяя **A7.5.1**, получаем $a_i = 4$.

Шаг 2. На этом шаге мы модифицируем $p_w(x) = p_w(4 + x) = x^3 + 9x^2 + 20x - 1$ и $\mathbf{M}_w(x) := \mathbf{M}_w(4 + x) = (4x + 19)/(3x + 14)$.

Шаг 3. Очевидно, что $p_w(0) \neq 0$.

Шаг 4. $|19/14 - 4/3| = 0.0238$.

Шаг 5. Здесь мы вычисляем новые значения $p_w(x) = x^3 - 20x^2 - 9x - 1$ и $\mathbf{M}_w(x) = (19x + 4)/(14x + 3)$ и переходим к шагу 1b.

Шаг 1b. Применяя **A7.5.1**, получаем $a_i = 20$.

Шаг 2. На этом шаге мы модифицируем $p_w(x) := p_w(20 + x) = x^3 + 40x^2 + 391x - 181$ и $\mathbf{M}_w(x) := \mathbf{M}_w(20 + x) = (19x + 384)/(14x + 283)$.

Шаг 3. Очевидно, что $p_w(0) \neq 0$.

Шаг 4. $|19/14 - 384/283| = 0.00025 < \varepsilon$,

и мы завершаем вычисления.

Выражая в десятичном виде концевые точки последнего изолирующего интервала, мы получаем $384/283 = 1.3568904$ и $19/14 = 1.3571428$ (см. также упражнения по программированию для разд. 7.5.1). Поэтому с точностью $\varepsilon = 0.01$ приближенное значение корня равно 1.35.

7.6

Эмпирическое сравнение двух методов аппроксимации вещественных корней

В этом разделе мы даем несколько таблиц, сравнивающих методы бисекций и цепных дробей для аппроксимации вещественных корней полиномиального уравнения. Мы сравниваем как теоретические аспекты, так и фактические времена вычисления для полиномов Чебышёва.

Таблица 7.6.1

Сравнение числа неполных частных и бисекций, необходимых для получения требуемой точности

m	Бисекции $1/2^m$	Цепные дроби $1/F_m^2$
1	0.5	1
5	0.03125	0
10	9.7×10^{-4}	3.3×10^{-4}
15	3.1×10^{-5}	2.7×10^{-6}
20	9.5×10^{-7}	2.2×10^{-8}
25	1.9×10^{-8}	1.7×10^{-10}
30	9.3×10^{-10}	1.4×10^{-12}
35	1.5×10^{-11}	3.1×10^{-14}
40	2.3×10^{-13}	9.5×10^{-17}

Прежде всего определим число бисекций и неполных частных соответственно, необходимых для каждого рассматриваемого метода, чтобы аппроксимировать корень с заданной точностью. В предположении, что исходный полином имеет только один положительный

корень, мы можем взять $(0, \infty)$ как исходный интервал для метода бисекций. Кроме того, для метода цепных дробей мы предполагаем худший из возможных случаев, т.е. каждое неполное частное равно 1; в этом случае мы имеем $q_m = F_m$, где F_m есть m-й элемент последовательности Фибоначчи. Из табл. 7.6.1 видно, что при сделанных предположениях для аппроксимации корня с одной и той же точностью бисекций требуется больше, чем вычислений неполных частных.

Таблица 7.6.2

Времена вычислений (в секундах) аппроксимации всех вещественных корней полиномов Чебышёва ($\varepsilon = 10^{-15}$)

Степень	Бисекции	Метод цепных дробей		
		Правило Коши	Теорема 7.5.1 с бисекциями	Предобработка
2	17.2	11.5	6.7	5.4
3	17.9	10.3	4.9	3.8
4	42.3	38.7	15.7	10.3
5	45.8	40.0	16.4	10.8
6	83.1	99.8	46.2	29.2
7	90.9	105.1	44.6	27.0
8	146.3	277.8	93.0	50.2
9	170.6	257.6	106.2	60.2
10	243.2	524.3	202.8	116.2

В табл. 7.6.2 показаны времена вычислений, требующиеся для аппроксимации ($\varepsilon = 10^{-15}$) вещественных корней полиномов Чебышёва с использованием методов бисекций и цепных дробей. Все времена даны в секундах. Мы сравниваем три версии метода цепных дробей: версии, которые используют (1) правило Коши, (2) теорему 7.5.1 с бисекциями и (3) предварительную обработку. В версии (3) мы предполагаем, что список неполных частных задается как входной. Таким образом, мы не тратим время на вычисление функций floor. (Результаты версии 3 отражают оптимальное время аппроксимации вещественного корня с использованием метода цепных дробей.) Различие между версиями, использующими теорему 7.5.1 с бисекциями и предварительную обработку, отражает время, затрачиваемое на вычисление функций floor. Мы напоминаем читателю, что для того,

Таблица 7.6.3

Аппроксимации вещественных корней полиномов Чебышёва степеней 2–10 ($\varepsilon = 10^{-15}$)

Степень	Список неполных частных для отделения	Список неполных частных для аппроксимации	Изолирующие интервалы
2	( )	(0; 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2)	0.70710678118654683338 0.70710678118654764296
3	( )	(0; 1, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2)	0.86602540378443847925 0.86602540378443865879
4	(0; 1, 1)	(11, 7, 3, 2, 1, 1, 1, 1, 20, 5, 3, 11, 1, 7)	0.92387953251128634045 0.92387953251128676332
	(0; 2)	(0, 1, 1, 1, 1, 2, 2, 4, 3, 1, 19, 6, 8, 3, 2, 9)	0.38268343236508971655 0.38268343236509064218
5	(0; 1, 1, 1)	(1, 2, 1, 6, 1, 56, 1, 54, 1, 1, 1, 10, 1, 16)	0.58778525229247311679 0.58778525229247338805
	(0; 1, 2)	(17, 2, 3, 6, 5, 1, 1, 1, 3, 2, 1, 25, 2, 2, 1, 1)	0.95105651629515309991 0.95105651629515359558
6	(0; 1, 2, 1)	(1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2)	0.70710678118654683338 0.70710678118654764296
	(0; 1, 3)	(25, 2, 1, 7, 21, 1, 8, 1, 3, 10, 1, 2, 3)	0.96592582628906811211 0.96592582628906894040
	(0; 2)	(1, 1, 6, 2, 1, 30, 5, 2, 9, 3, 1, 10, 5, 5)	0.25881904510252075804 0.25881904510252080914
7	(0; 1, 3, 1)	(0, 1, 2, 2, 24, 2, 3, 2, 2, 2, 2, 2, 8, 1, 8, 1)	0.78183148246802977984 0.78183148246803000835
	(0; 1, 4)	(34, 1, 7, 1, 2, 3, 1, 1, 2, 3, 1, 132, 2, 3)	0.97492791218182315610 0.97492791218182365484
	(0; 2)	(0, 3, 3, 1, 1, 3, 1, 29, 1, 3, 1, 18, 16, 1, 1, 2)	0.43388373911755805748 0.43388373911755880719
8	(0; 1, 1, 3, 1)	(0, 14, 14, 17, 1, 1, 10, 2, 1, 2, 18, 1)	0.83146961230254518379 0.83146961230254537502
	(0; 1, 1, 4)	(46, 23, 43, 8, 1, 2, 1, 3, 1, 15)	0.98078528040322959629 0.98078528040323045751
	(0; 1, 1, 1)	(2, 1, 840, 2, 1, 4, 1, 3, 21, 1, 17)	0.55557023301960177859 0.55557023301960224274
	(0; 2)	(3, 7, 1, 17, 1, 13, 3, 2, 7, 1, 1, 8, 1, 1, 1, 40)	0.19509032201612826749 0.19509032201612828909

Таблица 7.6.3 (продолжение)

Степень	Список неполных частных для		Изолирующие интервалы
	отделения	аппроксимации	
9	(0; 1, 1, 5, 1)	(1, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2, 6, 2)	0.86602540378443847925 0.86602540378443865879
	(0; 1, 1, 6)	(57, 1, 4, 1, 1, 1, 6, 1, 1, 6, 10, 1, 1, 3, 1, 1, 2, 6)	0.98480775301220799222 0.98480775301220806644
	(0; 1, 1, 1)	(0, 3, 1, 72, 1, 1, 2, 3, 1, 1, 3, 14, 1, 1, 1, 16, 1)	0.64278760968653925445 0.64278760968653933737
	(0; 2)	(0, 1, 12, 8, 17, 1, 1, 2, 1, 1, 1, 49, 1, 20, 3)	0.34203014332566870745 0.34203014332566873472
10	(0; 1, 2, 6, 1, 1)	(3, 1, 14, 13, 1, 1, 10, 1, 72, 23, 1)	0.45399049973954674160 0.45399049973954687273
	(0; 1, 2, 6, 2)	(3, 2, 1, 1, 4, 1, 2, 2, 1, 35, 1, 2, 4, 1, 3, 4, 65)	0.15643446504023080945 0.15643446504023086913
	(0; 1, 2, 6, 1)	(4, 1, 2, 1, 1, 3, 1, 12, 5, 19, 11, 1, 1)	0.89100652418836719434 0.89100652418836801487
	(0; 1, 2, 7)	(71, 4, 2, 7, 3, 1, 1, 42, 1, 12, 18)	0.98768834059513772582 0.98768834059513783641
	(0; 1, 2, 1)	(1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2)	0.70710678118654683338 0.70710678118654764296

чтобы получить функцию floor для некоторого корня, нужно несколько раз применить правило Коши.

Эти результаты показывают, что версия, использующая теорему 7.5.1 с бисекциями, существенно лучше версии, использующей правило Коши. С другой стороны, сравнивая ее с результатами версии, использующей предварительную обработку, мы замечаем, что еще имеется место для усовершенствования.

В качестве иллюстрации алгоритма аппроксимации в табл. 7.6.3 мы приводим список неполных частных с аппроксимирующими интервалами для каждого корня полиномов Чебышёва степеней 2–10 ($\varepsilon = 10^{-15}$). Поскольку полиномы Чебышёва симметричны, мы рассматриваем только положительные корни. Для полиномов нечетной степени мы опускаем корень $x = 0$.

Отметим, что последняя цифра в списке отделений (неполных частных) и первая цифра в списке аппроксимаций (неполных частных)

составляют одно и то же неполное частное.

Упражнения

Раздел 7.2.1

1. Докажите лемму 7.2.2.
2. Вычислите верхнюю границу числа вещественных корней следующих полиномиальных уравнений в интервале $(-2, 4)$:
   - **a.** $p(x) = x^3 - 3x + 1 = 0$;
   - **b.** $p(x) = x^4 - 4x^3 + x^2 + 6x + 2 = 0$;
   - **c.** $p(x) = 20x^5 + 5x^4 - 20x^2 - 10x + 2 = 0$.

Раздел 7.2.2

1. Закончите пример, предшествующий алгоритму **STURM**.
2. Пользуясь методом Штурма, отделите вещественные корни полиномиальных уравнений в интервале $(-2, 4)$:
   - **a.** $p(x) = x^3 - 3x + 1 = 0$;
   - **b.** $p(x) = x^4 - 4x^3 + x^2 + 6x + 2 = 0$;
   - **c.** $p(x) = 20x^5 + 5x^4 - 20x^2 - 10x + 2 = 0$.

Раздел 7.2.3

1. Закончите пример этого раздела.
2. Докажите, что верхняя граница абсолютных значений корней уравнения $p(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_0 = 0,\ c_i \in \mathbb{Z}$, равна

$$b = 2 \max_{1 \le k \le n} |c_{n-k}/c_n|^{1/k}. \tag{C4}$$

3. Используя (C), (C4) и алгоритм **BPR**, вычислите и сравните верхние границы положительных, отрицательных и абсолютных значений корней полиномиальных уравнений:
   - **a.** $p(x) = x^3 - 3x + 1 = 0$;
   - **b.** $p(x) = x^4 - 4x^3 + x^2 + 6x + 2 = 0$;
   - **c.** $p(x) = 20x^5 + 5x^4 - 20x^2 - 10x + 2 = 0$.

Раздел 7.2.4

1. Вычислите наименьшее расстояние между любыми двумя корнями полиномиальных уравнений:
   a. $p(x) = x^3 - 3x + 1 = 0$;
   b. $p(x) = x^4 - 4x^3 + x^2 + 6x + 2 = 0$;
   c. $p(x) = 20x^5 + 5x^4 - 20x^2 - 10x + 2 = 0$.

Раздел 7.3.2

1. Покажите, что подстановка $x := \alpha + 1/x$ эквивалентна *обобщенному* сдвигу длины α, сопровождаемому инверсией.

Раздел 7.3.4

1. Сможете ли вы объяснить на основании теоремы 7.3.9, почему мы *должны* использовать правило Коши? Что станет с методом цепных дробей 1978 г., если мы будем использовать нижнюю границу *абсолютных* значений корней?
2. Что происходит с комплексными корнями полинома $p(x)$, когда мы применяем метод цепных дробей 1978 г.? Сформулируйте предложение, аналогичное лемме 7.3.11.
3. Пользуясь методом цепных дробей 1978 г., отделите вещественные корни полиномиальных уравнений:
   a. $p(x) = x^3 - 3x + 1 = 0$;
   b. $p(x) = x^4 - 4x^3 + x^2 + 6x + 2 = 0$;
   c. $p(x) = 20x^5 + 5x^4 - 20x^2 - 10x + 2 = 0$.
4. Перепишите алгоритм **ACF1978** в «структурном» виде на языке программирования по своему выбору.

Раздел 7.5.1

1. Используя описанный выше метод бисекций, аппроксимируйте с точностью $\varepsilon = 0.01$ второй положительный корень уравнения $x^3 - 7x + 7 = 0$, изолирующий интервал которого есть $(3/2, 2)$.
2. Используя описанный выше метод бисекций, аппроксимируйте с точностью $\varepsilon = 0.01$ вещественные корни следующих полиномиальных уравнений (их изолирующие интервалы были получены в упражнениях к разд. 7.2.2 и/или 7.3.4.):
   a. $p(x) = x^3 - 3x + 1 = 0$;
   b. $p(x) = x^4 - 4x^3 + x^2 + 6x + 2 = 0$;
   c. $p(x) = 20x^5 + 5x^4 - 20x^2 - 10x + 2 = 0$.

Раздел 7.5.2

1. Используя описанный выше метод цепных дробей, аппроксимируйте с точностью $\varepsilon = 0.01$ второй положительный корень уравнения $x^3 - 7x + 7 = 0$, изолирующий интервал которого есть $(3/2, 2)$. Как мы видели в примере к разд. 7.3.4, этот изолирующий интервал был получен из $p_{\mathrm{M}}(x) = x^3 + x^2 - 2x - 1$ и $\mathbf{M}(x) = (2x + 3)/(x + 2)$.
2. Используя описанный выше метод цепных дробей, аппроксимируйте с точностью $\varepsilon = 0.01$ вещественные корни следующих полиномиальных уравнений (их изолирующие интервалы были получены в упражнениях к разд. 7.3.4.):
   - **a.** $p(x) = x^3 - 3x + 1 = 0$;
   - **b.** $p(x) = x^4 - 4x^3 + x^2 + 6x + 2 = 0$;
   - **c.** $p(x) = 20x^5 + 5x^4 - 20x^2 - 10x + 2 = 0$.

Упражнения по программированию

Раздел 7.2.3

1. Реализуйте процедуры для
   - **a.** вычисления $\lfloor \log_2 |i| \rfloor$ для любого целого $i \neq 0$ за время $O[L(|i|)]$;
   - **b.** вычисления 2^k для неотрицательного целого k за время $O(|k| + 1)$;
   - **c.** вычисления $[|i|/2k]$ для любых целых i и k, положительных или отрицательных, за время $O[L(|i|) + |k|]$.

Раздел 7.5.1

1. Реализуйте процедуру перевода рационального числа в десятичное представление. Число цифр, которые требуется получить в десятичном представлении, должно быть параметром; цифры должны получаться *по одной*. Эта процедура может тогда использоваться, чтобы выражать в десятичном виде рациональные концевые точки изолирующего интервала корня.

Раздел 7.5.2

1.

a. Реализуйте процедуру для реализации теоремы 7.5.1 за время

$$O\{nL[|p9x)|_\infty] + L^2[|p(x)|_\infty]\}.$$

b. Реализуйте процедуру **A7.5.1**, которая, пользуясь теоремой 7.5.1 и делением пополам, вычисляет целую часть единственного положительного корня уравнения $p(x) = 0$ за время

$$t_{\mathbf{A7.5.1}}[p(x)] = O\{n^2 L^3[|p(x)|_\infty]\}.$$

Исторические замечания и литература

1a. Мы подробно рассмотрели только методы, разработанные Штурмом и автором, потому что только они могут рассматриваться как классические, поскольку базируются на двух очень старых и связанных между собой теоремах. Существуют еще два метода отделения вещественных корней: (i) при помощи дифференцирования, метод, основанный на теореме Ролля [см. статьи Коллинза и Лооса (Collins, Loos, 1976, 1982)], и (ii) метод Коллинза–Акритаса, который базируется на полной модификации теоремы Винсента. Однако оба этих метода не представляют сколько-нибудь значительного интереса, поскольку они основаны на процедуре *бисекции* (так же, как метод Штурма), теоретическая и практическая эффективность которых ниже, чем у авторского метода цепных дробей 1978 г.

1b. Авторский метод цепных дробей 1978 г. получил *первую премию* на конкурсе студенческих работ, когда он был впервые представлен на 16-ю ежегодную юго-восточную региональную конференцию ACM в Атланте, Джорждия (апрель 1978).

1c. Метод Коллинза–Акритаса был разработан немедленно после того, как Акритасом была обнаружена в 1975–1976 гг. теорема Винсента, и описан в статьях Коллинза и Акритаса (Collins, Akritas, 1976), Коллинза и Лооса (Collins, Loos, 1982) и Коллинза [в (Rice, 1977)] под названием «*модифицированный* метод Успенского». Относительно этого пункта см. статью Акритаса 1986 г. и историческое замечание 7.

2. В книге (Obreschkoff, 1963), с. 61–87, имеются другие доказательства теоремы Фурье, принадлежащие Гурвицу и Лагерру, а также ее обобщение, принадлежащее Обрешкову.

3. Хайндель получил другую оценку для метода Штурма; а именно, он показал, что если $p(x) = 0$ — полиномиальное уравнение с целыми коэффициентами от одной неизвестной степени $n > 0$ без кратных корней, то время вычислений в методе Штурма — это

$$O\{n^{13}L^3[|p(x)|_\infty]\}.$$

Кроме того, вместо отделения сначала положительных, а затем отрицательных корней, как первоначально предложил Штурм, Хайндель вычисляет верхнюю границу b абсолютных значений корней, так что все они находятся в интервале $(-b, b)$; этот интервал затем делится на части.

4. Правило Коши, несмотря на его значимость, не слишком хорошо известно; мы нашли его формулировку только в книге (Obreschkoff, 1963), с. 50–51. Кроме того, Ван дер Слюс (van der Sluis) доказал теорему (теорема 2.7), из которой можно заключить, что правило Коши — наилучшее.

5. Теорема Винсента не упоминалась какими-либо авторами, за исключением Обрешкова (Obreschkoff, 1963) и Успенского (Uspensky, 1948). Автор обнаружил ее в 1975–1976 гг., пересматривая методы отделения вещественных корней уравнений, представленные Успенским. Имеется также следующее обобщение теоремы Винсента [см. статью (Chen, 1987)]. *Теорема Ванга* (1960 г.): Пусть $p(x) = 0$ — полиномиальное уравнение с целыми коэффициентами степени $n \geq 3$, и предположим, что оно имеет не менее двух перемен знаков в последовательности своих коэффициентов; кроме того, пусть $\Delta > 0$ — наименьшее расстояние между любыми двумя из его корней. Пусть m' — наименьший положительный индекс, такой, что $F^2_{m'-1} > 1/\Delta$, где F_k есть k-й элемент последовательности Фибоначчи $1, 1, 2, 3, 5, 8, 13, 21, \ldots$, и пусть m'' — наименьшее положительное целое, такое, что $m'' > 1 + \lceil \log_\phi n \rceil / 2$. Если положить $m = m' + m''$, то произвольная подстановка цепной дроби

$$x := a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \cfrac{1}{a_m + \cfrac{1}{y}}}}$$

с неотрицательным целым a_1 и положительными целыми $a_2, \ldots, a_m$ преобразует $p(x) = 0$ в уравнение $p_{ti}(y) = 0$, имеющее r перемен знаков в последовательности своих коэффициентов. Если $r = 0$, то в интервале I_m с (неупорядоченными) концевыми точками p_m/q_m, p_{m-1}/q_{m-1} [полученном из (V13)] нет корней полинома $p(x)$. Если $r > 0$, то уравнение $p(x) = 0$ имеет единственный положительный вещественный корень кратности r в интервале I_m.

Несмотря на теоретический интерес, эмпирические результаты показывают, что реализация теоремы Ванга работает плохо, и, следовательно, она не представляет практического интереса для отделения кратных корней полиномиальных уравнений; вместо этого следует пользоваться теоремой Винсента в комбинации с разложением на свободные от квадратов множители.

6. Успенский (Uspensky, 1948, p. 298–304) расширил теорему Винсента, чтобы получать верхнюю границу числа подстановок, которые нужно выполнить. Его работа, однако, содержит определенные ошибки в формулировке и доказательстве, которые были исправлены автором (Akritas, 1978b). В этом тексте мы даем правильную версию расширения теоремы Винсента; для полноты мы также добавляем ее доказательство, которое гораздо короче доказательства Успенского благодаря тому, что мы используем лемму 7.3.7; см. также статьи (Akritas, 1981a) и (Akritas, Danielopoulos, 1985).

7a. В статьях Коллинза и Акритаса (Collins, Akritas, 1976b), Коллинза и Лооса (Collins, Loos, 1982) и Коллинза [в (Rice, 1977)] (экспоненциальный) метод Винсента цепных дробей 1836 г. ошибочно приписывался Успенскому, вследствие чего метод Коллинза–Акритаса появился под названием «*модифицированный* метод Успенского»; это случилось, вероятно, из-за утверждения Успенского (в предисловии к его книге), что он сам изобрел этот метод. Однако, как было подчеркнуто в работах Акритаса (Akritas, 1978a, p. 85–86, 1981a, и 1986), Успенский, собственно, взял метод Винсента и *удвоил* его время вычислений, потому что не был знаком с теоремой Бюдана; а именно, после подстановки $x := 1 + x$, примененной к некоторому уравнению $p_{ti}(x) = 0$, Успенский должен был выполнять подстановку $x := 1/(1 + x)$, чтобы удостовериться, что $p_{ti}(x) = 0$ не имеет корней в интервале $(0, 1)$. (Винсент, естественно, получает эту информацию, пользуясь теоремой Бюдана.)

Что можно рассматривать как вклад Успенского — это только то, что для выполнения подстановки вида $x := a_i + x$ он использовал метод Руффини–Горнера. Винсент, напротив, пользовался теоремой о разложении Тейлора.

7b. Теорема о разложении Тейлора довольно утомительна для вычислений вручную, но она давала доступную в то время «технологию». Статья Горнера появилась в 1819 г., но Винсент, очевидно, не знал об этом; также и Горнер не сознавал того, что его опередил Руффини в 1804 г. (Cajori, 1911). В любом случае Акритас и Даниелопулос (Akritas, Danielopoulos, 1980) показали, что и теорема о разложении Тейлора, и метод Руффини–Горнера имеют примерно равные времена вычислений.

8. Штурм в своей статье 1835 г. представил решение проблемы, связанной с методом Лагранжа, а именно, когда число корней между двумя последовательными целыми числами больше одного. Штурм фактически использует как последовательность Штурма, так и теорему Винсента, чтобы отделить и аппроксимировать корни с любой точностью. Подробнее об этом написано на с. 292–297 и 299–305 статьи Штурма 1835 г.

9. Это находится в полном соответствии с замечанием Винсента в его статье 1836 г., с. 352–353: «Pour approcher davantage de la valeur de cette racine, nous pourrions continuer le calcul en suivant toujours la meme marche; et nous serions surs de n'avoir, dans toutes les transformées subsequentes, qu'une seule variation, et par conséquent une seule racine positive, laquelle, de plus, serait toujours necessarement plus grande que l'unité.» («Чтобы лучше аппроксимировать значение этого корня, мы можем продолжать вычисления, придерживаясь все время той же процедуры; и мы можем быть уверены, что во всех последующих преобразованных уравнениях имеется только одна перемена знаков, а следовательно, только один положительный корень, который, более того, обязательно всегда будет больше единицы».)

10. Сам Винсент не следует этому подходу, потому что в своей статье (с. 353) он утверждает: «la réduction en fraction continue ne croissant que tres lentement, changeons maintenant notre marche» («поскольку разложение в цепную дробь осуществляется очень медленно, давайте изменим теперь нашу процедуру»).

Abel N.H. Beweis der Unmoeglichkeit algebraische Gleichungen von hoeheren Graden als den vierten allgemein aufzuloesen. *Journal fur die reine und angewandte Mathematik* **1**, 65–84, 1826.

Akritas A.G. *Vincent's theorem in algebraic manipulation.* Ph.D.Thesis, Operations Research Program, North Carolina State University, Raleigh, North Carolina, 1978a.

Akritas A.G. A correction on a theorem by Uspensky. *Bulletin of the Gkeek Mathematical Society* **19**, 278–285, 1978b.

Akritas A.G. The fastest exact algorithms for the isolation of the real roots of a polynomial equation. *Computing* **24**, 219–313, 1980a.

Akritas A.G. An implementation of Vincent's theorem. *Numerische Mathematik* **36**, 53–62, 1980.

Akritas A.G. Vincent's forgotten theorem, its extension and application. *International Journal of Computers and Mathematics with Applications* **7**, 309–317, 1981.

Akritas A.G. Exact algorithms for the implementation of Cauchy's rule. *International Journal of Computer Mathematics* **9**, 323–333, 1981.

Akritas A.G. Reflections on a pair of theorems by Budan and Fourier. *Mathematics Magazine* **55**, 292–298, 1982.

Akritas A.G. There is no «Uspensky's method». Extended Abstract. *Proceedings of the 1986 Symposium on Symbolic and Algebraic Computation*, (Waterloo, Ontario, Canada), 1986, pp. 88-90.

Akritas A.G., Danielopoulos S.D. On the forgotten theorem of Mr. Vincent. *Historia Mathematica* **5**, 427–435, 1978.

Akritas A.G., Danielopoulos S.D. On the complexity of algorithms for the translation of polynomials. *Computing* **24**, 51–60, 1980.

Akritas A.G., Danielopoulos S.D. A converse rule of signs for polynomials. *Computing* **34**, 283–286, 1985.

Akritas A.G., Ng K.H. Exact algorithms for polynomial real root approximation using continued fractions. *Computing* **30**, 63–76, 1983.

Bocher M. The published and unpublished work of Charles Sturm on algebraic and differential equations. *Bulletin of the American Mathematical Society* **18**, 1–18, 1911-12.

Burnside W.S., Panton A.W. *The theory of equations*, Vol. 1, 2nd ed., Dover, New York, 1960.

Cajori F. Horner's method of approximation anticipated by Ruffini. *American Mathematical Society Bulletin* **17**, 409–414, 1911.

Cajori F. A history of the arithmetical methods of approximation to the roots of numerical equations of one unknown quantity. *Colorado College Publications, General Series No. 51*, Science Series Vol. XII, No. 7, pp. 171–287, Colorado Springs, Colorado, 1910.

Cantor D.G., Galyean P.H., Zimmer H.G. A continued fraction algorithm for real algebraic numbers. *Mathematics of Computation* **26**, 785–791, 1972.

Chen J. A new algorithm for the isolation of the real roots of polynomial equations. *Second International Conference on Computers and Applications* (Beijing, People's Republic of China), 714–719, 1987.

Collins G.E., Akritas A.G. Polynomial real root isolation using Descartes' rule of signs. *Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation*, Yorktown Heights, NY, 1976, pp. 272–275.

Collins G.E., Loos R. Polynomial real root isolation by differentiation. *Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation*, Yorktown Heights, NY, 1976, pp. 15–20.

Collins G.E., Loos R. Real zeros of polynomials. In *Computer Algebra symbolic and algebraic computations.* B. Buchberger, G.E. Collins, and R. Loos, eds, Springer Verlag, New York, *Computing Supplement* **4**, 83–94, 1982.

Dickson L.E. *First course in the theory of equations.* Wiley, New York, 1922.

Heindel L.E. Integer arithmetic algorithms for polynomial real zero determination. *Journal of the ACM* **18**, 533–548, 1971.

Hurwitz A. Ueber den Satz von Budan-Fourier. *Mathematische Annalen* **71**, 584–591, 1912.

Lagrange J.L. *Traité de la résolution des équations numériques.* Paris, (n.p.) 1798.

Lloyd E.K. On the forgotten Mr. Vincent. *Historia Mathematica* **6**, 448–450, 1979.

Mahler K. An inequality for the discriminant of a polynomial. *Michigan Mathematical Journal* **11**, 257–262, 1964.

Marcus M., Minc H. *Introduction to linear algebra.* MacMillan, New York, 1965.

Mignotte M. Sur la complexité de certains algorithms ou intervient la séparation des racines d'un polynome. *Revue Française d'Automatique, Informatique et Recherce Opérationnelle (RAIRO)* **10**, 51–55, 1976.

Mignotte M., Payafar M. Distance entre les racines d'un polynome. *RAIRO–Analyse Numérique* **13**, 181–192, 1979.

Ng K.H. *Polynomial real root approximation using continued fractions.* M.S. Research Report, University of Kansas, Department of Computer Science, Lawrence, KS, 1980.

Nordgaard M.A. *A historical survey of algebraic methods of approximating the roots of numerical higher equations up to the year 1819.* Teachers College, Columbia University, New York, 1922.

Obreschkoff N. *Verteilung und Berechnung der Nullstellen reeller Polynome.* VEB Deutscher Verlag der Wissenschaften, Berlin, 1963.

Poggendorff J.C. *Biographisch–Literarisches Handwoerterbuch zur Geschichte der exacten Wissenschaften.* J.A. Barth, Leipzig, 1863.

Rice J. *Mathematical software III.* Academic Press, New York, 1977, pp. 35–68.

Ruffini P. Sopra la determinazione delle radici nelle equazioni numeriche di qualunque grado.... *Societa Italiana delle Scienze*, 1804.

Rump S.M. Polynomial minimum root separation. *Mathematics of Computation* **33**, 327–336, 1979.

Specht W. Abschaetzungen der Wurzeln algebraischer Gleichungen. *Mathematische Zeitschrift* **52**, 310–321, 1949.

Sturm C. Mémoire sur la résolution des équations numériques. *Mémoires des Savants Etrangers* **6**, 271–318, 1835.

Todhunter I. *Theory of equations.* MacMillan, London, 1882.

Turnbull H.W. *Theory of equations.* 5th ed. Oliver & Boyd, Edinburgh and London, 1957.

Uspensky J.V. *Theory of equations.* McGraw-Hill, New York, 1948.

van der Sluis A. Upperbounds for roots of polynomials. *Numerische Mathematik* **15**, 250–262, 1970.

van der Waerden B.L. *Erwachende Wissenschaft.* Birkhaeuser Verlag, Basel and Stuttgart, 1956.

Verbaeten P. Computing real zeros of polynomials with SAC-1. *ACM–SIGSAM Bulletin* **9**, 8–10, 1975.

Vincent A.J.H. Sur la résolution des équations numériques. *Journal de Mathématiques Pures et Appliquées* **1**, 341–372, 1836.

Weisner L. *Introduction to the theory of equations.* MacMillan, New York, 1938.

ЛИНЕЙНАЯ АЛГЕБРА

В этом разделе дан краткий обзор тех аспектов линейной алгебры, которые существенны для данной книги. Ниже мы считаем, что R — коммутативное кольцо типа $\mathbb{Q}, \mathbb{R}, \mathbb{Z}$ или $\mathbb{Z}_m$.

Матричное умножение

Вектор-строка, или просто вектор, — это строка элементов из кольца R, т.е. $\mathbf{v} = (a_1, \ldots, a_n)$. Вектор-столбец, или *транспонированная* вектор-строка, — это столбец элементов из R, т.е.

$$\mathbf{v}^T = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}.$$

Матрица $\mathbf{A}$ размера $m \times n$ — это прямоугольный массив из mn элементов кольца R, где m — количество строк, а n — количество столбцов, т.е.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{11} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix},$$

который можно представлять как набор вектор-строк, расположенных в столбец, или как набор вектор-столбцов, расположенных в строку. Транспонированная $m \times n$-матрица $\mathbf{A}^T$ — это $n \times m$-матрица, полученная из исходной превращением строк в столбцы (или столбцов в строки).

Данные вектор-строку из n элементов (расположенную слева) и вектор-столбец с таким же количеством элементов (расположенный справа) можно перемножить и получить элемент из R, т.е.

$$(a_1, \ldots, a_n) \begin{bmatrix} b_1 \\ . \\ . \\ . \\ b_n \end{bmatrix} = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n.$$

В качестве примера рассмотрим случай $R = \mathbb{Z}$; тогда

$$(1,2,3)\begin{bmatrix}4\\5\\6\end{bmatrix} = 1\cdot 4 + 2\cdot 5 + 3\cdot 6 = 32.$$

По данным $m\times n$-матрице $\mathbf{A}$ (расположенной слева) и n-элементному вектор-столбцу $\mathbf{v}^T$ (расположенному справа) можно сформировать произведение $\mathbf{A}\mathbf{v}^T$, рассматривая матрицу как набор из m n-элементных вектор-строк и выполняя m умножений вектор-строк матрицы $\mathbf{A}$ на $\mathbf{v}^T$. Результат, $\mathbf{A}\mathbf{v}^T$, — это вектор из m элементов. Например,

$$\begin{bmatrix}1 & 2 & 3 & 4\\5 & 6 & 7 & 8\end{bmatrix}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} = \begin{bmatrix}10\\26\end{bmatrix}.$$

Если даны $m \times n$-матрица $\mathbf{A}$ (слева) и $n \times p$-матрица $\mathbf{B}$ (справа), то мы можем перемножить их, рассматривая $\mathbf{A}$ как набор n-элементных строк, а $\mathbf{B}$ — как набор n-элементных столбцов. Результат — это $m \times p$-матрица $\mathbf{AB}$, где элемент в i-й строке и j-м столбце получен перемножением i-й строки матрицы $\mathbf{A}$ и j-го столбца матрицы $\mathbf{B}$. Например,

$$\begin{bmatrix}1 & 2 & 3 & 4\\5 & 6 & 7 & 8\end{bmatrix}\begin{bmatrix}1 & 1\\1 & 2\\1 & 2\\1 & 1\end{bmatrix} = \begin{bmatrix}10 & 15\\26 & 39\end{bmatrix}.$$

Следует отметить, что порядок, в котором перемножаются матрицы (т.е. которая из матриц стоит слева, а которая — справа), очень важен. Существуют случаи, когда невозможно перемножить две матрицы, если поменять их порядок (читателю нужно построить пример); даже когда перемножение может быть выполнено в любом порядке, результат может быть различным; так, например,

$$\begin{bmatrix}1\\3\end{bmatrix}(2,1) = \begin{bmatrix}2 & 1\\6 & 3\end{bmatrix}, \quad \text{тогда как} \quad (2,1)\begin{bmatrix}1\\3\end{bmatrix} = 5.$$

Если $\mathbf{A}$ — матрица произвольного размера (в частности, вектор-строка или вектор-столбец) и s — элемент из кольца R, т.е. скаляр, то мы определяем матрицу $s\mathbf{A}$ как матрицу, в которой каждый элемент

матрицы $\mathbf{A}$ умножается на s. Например,

$$s\mathbf{A} = s\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} = \begin{bmatrix} sa_{11} & sa_{12} & \dots & sa_{1n} \\ sa_{21} & sa_{22} & \dots & sa_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ sa_{m1} & sa_{m2} & \dots & sa_{mn} \end{bmatrix}.$$

В частности,

$$x\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} xa_1 \\ xa_2 \\ xa_3 \end{bmatrix}.$$

Для двух данных векторов $\mathbf{v}_1 = (a_1, \dots, a_n)$ и $\mathbf{v}_2 = (b_1, \dots, b_n)$ положим $\mathbf{v}_1 + \mathbf{v}_2 = (a_1 + b_1, \dots, a_n + b_n)$. Аналогично можно складывать две матрицы $\mathbf{A}$ и $\mathbf{B}$ одинакового размера, понимая их как последовательность вектор-строк, т.е.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mn} \end{bmatrix} =$$
$$= \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} & \dots & a_{1n} + b_{1n} \\ a_{21} + b_{21} & a_{22} + b_{22} & \dots & a_{2n} + b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & a_{m2} + b_{m2} & \dots & a_{mn} + b_{mn} \end{bmatrix}.$$

Отметим, что сумма $\mathbf{A} + \mathbf{B}$ имеет смысл, только если $\mathbf{A}$ и $\mathbf{B}$ имеют одинаковую «форму». В частности, сумма $\mathbf{A} + \mathbf{B}$ определена, если $\mathbf{A}, \mathbf{B}$ — квадратные $n \times n$-матрицы.

Пусть $\mathbf{A}_n(R)$ — множество (квадратных) $n \times n$-матриц с элементами из множества R. Пусть $\mathbf{0}$ есть $n \times n$-матрица, полностью состоящая из нулей. Тогда очевидно, что $\mathbf{0} + \mathbf{A} = \mathbf{A} + \mathbf{0} = \mathbf{A}$ для любой $n \times n$-матрицы $\mathbf{A}$. Кроме того, пусть

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix};$$

тогда $\mathbf{AI} = \mathbf{IA} = \mathbf{A}$ для любой $n \times n$-матрицы $\mathbf{A}$. Имеет место следующая

Теорема А.1. Если R — коммутативное кольцо с единицей, то $\mathbf{A}_n(R)$ — кольцо с единицей.

Доказательство. Доказательство заключается в проверке набора аксиом и оставляется читателю в качестве упражнения. □

Заметим, что кольцо $\mathbf{A}_n(R)$ имеет делители нуля при любых $n \geq 2$. Например, при $n = 2$ рассмотрим

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

Кроме того, как будет видно ниже, для любого $n \geq 2$ существуют ненулевые $n \times n$-матрицы, не имеющие обратных.

Линейные уравнения

Матрицы и векторы — это удобный способ описывать системы линейных уравнений. Рассмотрим следующую систему из m уравнений с n неизвестными:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2, \\ \ldots\ldots\ldots & \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &= b_m. \end{aligned} \qquad \text{(ЛУ)}$$

Эта система уравнений называется *однородной*, если $b_1 = b_2 = \cdots = b_m = 0$, и *неоднородной* в противном случае.

Мы можем также рассматривать последнюю систему как равенство вектор-столбцов, так как два вектор-столбца равны в точности тогда, когда равны их соответствующие компоненты. Таким образом, мы можем переписать (ЛУ) любым из следующих двух способов.

(1) Используя определение сложения и умножения на скаляр вектор-столбцов, (ЛУ) можно записывать как

$$x_1 \begin{bmatrix} a_{11} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ \vdots \\ a_{mn} \end{bmatrix} = \begin{bmatrix} b_{1n} \\ \vdots \\ b_{mn} \end{bmatrix}.$$

Это означает, что решить первоначальную систему — то же самое, что записать вектор $(b_1, \ldots, b_m)^T$ в виде линейной комбинации (т.е. суммы произведений на скаляр) векторов $(a_{11}, \ldots, a_{m1})^T, \ldots, (a_{1n}, \ldots, a_{mn})^T$.

(2) Заметив, что левая часть (ЛУ) — это произведение матрицы $\mathbf{A}$ коэффициентов исходной системы на вектор $(x_1, \ldots, x_n)^T$, (ЛУ) можно записывать как $\mathbf{A}\mathbf{x}^T = \mathbf{b}^T$, где $\mathbf{x} = (x_1, \ldots, x_n)$, $\mathbf{b} = (b_1, \ldots, b_m)$ и

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix}.$$

Пример. Система уравнений

$$\begin{aligned} x_1 + \quad x_2 + x_3 &= 6, \\ 2x_1 + 3x_2 - x_3 &= 5, \\ 4x_1 \qquad\quad - x_3 &= 1 \end{aligned}$$

может быть записана как

$$x_1 \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix} + x_2 \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix} + x_3 \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 6 \\ 5 \\ 1 \end{bmatrix}$$

или как

$$\begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & -1 \\ 4 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 5 \\ 1 \end{bmatrix}.$$

Предположим теперь, что для системы уравнений $\mathbf{A}\mathbf{x}^T = \mathbf{b}^T$ существует $n \times m$-матрица $\mathbf{A}^{-1}$, обратная к $\mathbf{A}$, такая, что $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$. Если мы найдем такую матрицу $\mathbf{A}^{-1}$, то $\mathbf{A}^{-1}\mathbf{b}^T = \mathbf{A}^{-1}\mathbf{A}\mathbf{x}^T = \mathbf{I}\mathbf{x}^T = \mathbf{x}^T$ будет решением системы уравнений. Таким образом решение системы уравнений тесно связано с нахождением обратной матрицы. Например, матрица

$$\begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & -1 \\ 4 & 0 & -1 \end{bmatrix},$$

оказывается, имеет обратную

$$\begin{bmatrix} 3/17 & -1/17 & 4/17 \\ 2/17 & 5/17 & -3/17 \\ 12/17 & -4/17 & -1/17 \end{bmatrix},$$

так что

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3/17 & -1/17 & 4/17 \\ 2/17 & 5/17 & -3/17 \\ 12/17 & -4/17 & -1/17 \end{bmatrix} \begin{bmatrix} 6 \\ 5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}.$$

Определители и обратные матрицы

Рассмотрим квадратную $n \times n$-матрицу

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix},$$

где $\mathbf{A}$ — просто таблица чисел и как таковая не имеет численного значения. Мы намерены присвоить матрице $\mathbf{A}$ численное значение следующим образом:

(1) Выберем какой-нибудь элемент в первой строке, какой-нибудь элемент во второй строке, лежащий *не в том же столбце*, какой-нибудь элемент в третьей строке в столбце, *отличном от первых двух*, и так далее, пока не получим n чисел, никакие два из которых не лежат в одной строке или столбце; образуем все возможные такие группы чисел. Например, для данной 3×3-матрицы

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

мы имеем следующие возможности:

Варианты	Перестановки
a_{11}, a_{22}, a_{33}	1 2 3
a_{11}, a_{23}, a_{32}	1 3 2
a_{12}, a_{21}, a_{33}	2 1 3
a_{12}, a_{23}, a_{31}	2 3 1
a_{13}, a_{21}, a_{32}	3 1 2
a_{13}, a_{22}, a_{31}	3 2 1

Перестановка указывает на порядок вторых индексов, когда первые индексы расположены в естественном порядке $1, 2, \dots$.

(2) Вычислим количество *инверсий* для каждой перестановки, т.е. вычислим количество перемен местами двух соседних чисел,

необходимых для приведения перестановки к естественному порядку. Это может быть легко сделано следующим образом: для каждой перестановки, двигаясь слева направо, считаем, сколько чисел *справа* меньше, чем рассматриваемое. Например, для перестановки 3 1 2 мы имеем $2+0+0=2$, потому что 3 имеет два меньших числа справа, а 1, 2 таких не имеют. Таким образом, число инверсий для 3 1 2 равно 2; аналогично для 3 2 1 оно равно $3=2+1+0$.

(3) Теперь численное значение, соответствующее $\mathbf{A}$, — это сумма

$$\sum (-1)^m a_{1i_1} a_{2i_2} \dots a_{ni_n},$$

где суммирование ведется по всем i от 1 до n, каждое слагаемое в сумме есть произведение элементов, выбранных выше на шаге 1, и m — число инверсий в соответствующей перестановке. (*Мы имеем* $n!$ *слагаемых.*)

Пример. Дано $a_{22}a_{13}a_{31}$. Найдем i_1, i_2, i_3, а также выясним, прибавляется этот член или вычитается. Для нахождения i_1, i_2, i_3 мы просто переставим элементы так, чтобы первые индексы были расположены в естественном порядке: $a_{13}a_{22}a_{31}$. Теперь ясно, что $i_1=3$, $i_2=2$, $i_3=1$. Член вычитается, потому что $m=3$.

Определение A.2. Дана квадратная $n\times n$-матрица $\mathbf{A}$. Ее *определитель* — это

$$\det \mathbf{A} = \sum (-1)^m a_{1i_1} a_{2i_2} \dots a_{ni_n},$$

где суммирование ведется по всем перестановкам из n элементов, а m — число инверсий в соответствующей перестановке.

Иногда определитель также записывается как $|\mathbf{A}|$ или $|a_{ij}|$, $i,j=1,\dots,n$; кроме того, $\det(\mathbf{A})$ можно рассматривать как сумму произведений элементов из одной строки или столбца матрицы $\mathbf{A}$ на соответствующие *алгебраические дополнения*. Алгебраическое дополнение элемента a_{ij} — это определитель матрицы, полученной из $\mathbf{A}$ вычеркиванием i-й строки и j-го столбца, умноженный на $(-1)^{i+j}$ (см. также определение A.8). Для случая 3×3, который рассматривался выше, мы имеем (см. также теорему A.9).

$$\begin{aligned}|\mathbf{A}| = \det \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \\ = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - \\ a_{12}a_{21}a_{33} - a_{13}a_{22}a_{31} - a_{11}a_{23}a_{32}.\end{aligned}$$

Для случая 2×2 мы имеем

$$|\mathbf{A}| = \det \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = a_{11}a_{22} - a_{12}a_{21}.$$

Если $\mathbf{A}$ — треугольная матрица, т.е. матрица вида

$$\mathbf{A} = \begin{bmatrix} a_{11} & 0 & 0 & \dots & 0 \\ a_{21} & a_{22} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix},$$

то $\det(\mathbf{A}) = a_{11}a_{22} \dots a_{nn}$.

Если $\mathbf{A}$ есть $n \times n$-матрица с элементами из кольца R, то иногда она имеет обратную, $n \times n$-матрицу $\mathbf{A}^{-1}$, такую, что $\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$. Хорошо известная теорема линейной алгебры утверждает, что $\mathbf{A}^{-1}$ существует тогда и только тогда, когда $\det(\mathbf{A})$ — обратимый элемент кольца R. В частности, для 2×2-матриц, если определитель $\mathbf{A}$ обратим, то обратная к $\mathbf{A}$ матрица может быть найдена следующим образом. Если

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

и $1/(ad - bc)$ принадлежит R, то

$$\mathbf{A}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & d \end{bmatrix},$$

что легко проверяется. Для матриц $\mathbf{A}$ порядка 3×3 или больше формула для нахождения обратной к $\mathbf{A}$ матрицы через определители слишком сложна, чтобы приводить ее здесь. Однако мы можем найти обратную к $n \times n$-матрице $\mathbf{A}$, если она существует, используя операции над строками.

Элементарные операции над строками

Метод исключения Гаусса — наиболее полезная техника, когда используются только элементарные операции над строками. Существуют три такие операции:

(1) Перестановка двух строк матрицы.
(2) Умножение строки на обратимый скаляр.
(3) Прибавление к строке другой строки, умноженной на скаляр.

Эти операции над строками соответствуют действиям, которые можно выполнять над системой уравнений, не изменяя решений этих уравнений. Если $\mathbf{A}$ есть $m \times n$-матрица с элементами из поля R, то, используя операцию (3), можно получить в матрице нулевые элементы и преобразовать ее к *ступенчатой форме по строкам*. Следующие примеры — это матрицы такого вида:

$$\begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\begin{bmatrix} 1 & 2 & 0 & 0 & 4 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Мы можем найти обратную к $n \times n$-матрице $\mathbf{A}$, если она существует, решая относительно $\mathbf{x}^T$ уравнение

$$\mathbf{A}\mathbf{x}^T = \mathbf{y}^T = \mathbf{I}\mathbf{y}^T \quad \text{или} \quad (\mathbf{A}, \mathbf{I})(\mathbf{x}^T, -\mathbf{y}^T)^T = \mathbf{0}^T,$$

где $(\mathbf{A}, \mathbf{I})$ — это $n \times 2n$-матрица, левая часть которой есть $\mathbf{A}$, а правая часть — единичная $n \times n$-матрица $\mathbf{I}$, и где $\mathbf{x}, \mathbf{y}$ — векторы неизвестных. Если ступенчатая форма $n \times 2n$-матрицы $(\mathbf{A}, \mathbf{I})$ имеет вид $(\mathbf{I}, \mathbf{B})$, т.е. n самых левых столбцов ступенчатой формы $n \times 2n$-матрицы $(\mathbf{A}, \mathbf{I})$ образуют единичную $n \times n$-матрицу $\mathbf{I}$, то

$$(\mathbf{A}, \mathbf{I}) \begin{bmatrix} \mathbf{x}^T \\ -\mathbf{y}^T \end{bmatrix} = \mathbf{0}^T \quad \Longleftrightarrow \quad (\mathbf{I}, \mathbf{B}) \begin{bmatrix} \mathbf{x}^T \\ -\mathbf{y}^T \end{bmatrix} = \mathbf{0}^T \quad \Longleftrightarrow \quad \mathbf{x}^T = \mathbf{B}\mathbf{y}^T$$

тогда и только тогда, когда $\mathbf{B}$ — матрица, обратная к $\mathbf{A}$. Поэтому, для того чтобы вычислить $\mathbf{A}^{-1}$, обратную к $\mathbf{A}$ матрицу, записываем рядом $(\mathbf{A}, \mathbf{I})$ и производим операции над строками всей этой матрицы, пытаясь преобразовать $\mathbf{A}$ в $\mathbf{I}$. Если это удается, то теми же операциями над строками матрица $\mathbf{I}$ переводится в $\mathbf{A}^{-1}$.

Пример. Вычислим матрицу, обратную к

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}.$$

Конечно, у нас есть формула, чтобы получить $\mathbf{A}^{-1}$ для этой матрицы, но тот же результат получается применением операций над строками

$$\begin{bmatrix} 1 & 2 & 1 & 0 \\ 3 & 4 & 0 & 1 \end{bmatrix} \to \begin{bmatrix} 1 & 2 & 1 & 0 \\ 0 & -2 & -3 & 1 \end{bmatrix} \to$$

$$\begin{bmatrix} 1 & 2 & 1 & 0 \\ 0 & 1 & 3/2 & -1/2 \end{bmatrix} \to \begin{bmatrix} 1 & 0 & -2 & 1 \\ 0 & 1 & 3/2 & -1/2 \end{bmatrix},$$

т.е.

$$\begin{bmatrix} -2 & 1 \\ 3/2 & -1/2 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Далее мы приводим некоторые теоремы, иллюстрирующие действие операций над строками на определитель; их доказательства мало познавательны, и мы их опускаем. (Их можно найти в большинстве книг по линейной алгебре.)

Теорема А.3. Если любые две строки (или столбца) определителя поменять местами, то определитель поменяет знак.

Чтобы убедиться в справедливости этой теоремы, рассмотрим

$$|\mathbf{A}| = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \text{и} \quad |\mathbf{B}| = \det \begin{bmatrix} c & d \\ a & b \end{bmatrix}.$$

Тогда $|\mathbf{A}| = ad - bc$ и $|\mathbf{B}| = bc - ad = -(ad - bc)$. Сделайте то же для столбцов.

Теорема А.4. Если все элементы строки (или столбца) определителя умножить на k, то определитель умножится на k.

Сравните

$$|\mathbf{A}| = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \text{и} \quad |\mathbf{B}| = \det \begin{bmatrix} ka & kb \\ c & d \end{bmatrix}.$$

Очевидно, что $|\mathbf{A}| = ad - bc$, принимая во внимание, что $|\mathbf{B}| = kad - kbc = k(ad - bc) = k|\mathbf{A}|$. Аналогично проводится проверка для столбцов.

Теорема А.5. Если две строки (или столбца) определителя равны, определитель равен 0.

Рассмотрим

$$|\mathbf{A}| = \det \begin{bmatrix} a & b \\ a & b \end{bmatrix} = ab - ab = 0.$$

Теорема А.6. Если одну строку (или столбец), умноженную на число, прибавить к другой строке (или столбцу) определителя, то его значение не изменится.

Рассмотрим, например,

$$|\mathbf{A}| = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \text{и} \quad |\mathbf{B}| = \det \begin{bmatrix} a+kc & b+kd \\ c & d \end{bmatrix}.$$

Тогда $|\mathbf{A}| = ad - bc$ и $|\mathbf{B}| = (a+kc)d - (b+kd)c = ad + kcd - bc - kcd = ad - bc = |\mathbf{A}|$. Повторите то же самое для столбцов.

Наконец, справедлива следующая

Теорема A.7 (перемена местами строк и столбцов). Если строки определителя сделать столбцами, а столбцы — строками (в том же порядке), то определитель не изменится.

(Другими словами, определитель матрицы, транспонированной к $\mathbf{A}$, равен определителю матрицы $\mathbf{A}$.) То есть

$$|\mathbf{A}| = \det \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \det \begin{bmatrix} a & c \\ b & d \end{bmatrix} = ad - bc.$$

Разложение определителя по строке или столбцу

Из предыдущего примера видно, что мы можем легко вычислить значение определителя 2×2-матрицы. Но что можно сказать об определителе 3×3-матрицы или, в общем случае, $n \times n$-матрицы? Ниже мы дадим ответ на этот вопрос.

Определение A.8. Даны $n \times n$-определитель $|\mathbf{A}|$ и его элемент a_{mk}. Мы назовем *минором* элемента a_{mk} определитель размера $(n-1) \times (n-1)$, полученный вычеркиванием из матрицы $\mathbf{A}$ m-й строки и k-го столбца. Минор обозначается через $|\mathbf{A}_{mk}|$. *Алгебраическим дополнением* элемента a_{mk} мы назовем $(-1)^{m+k}|\mathbf{A}_{mk}|$,

Сформулируем основной результат.

Теорема A.9. Если дан $n \times n$-определитель $|\mathbf{A}|$, то

$$|\mathbf{A}| = (-1)^{m+1} a_{m1} |\mathbf{A}_{m1}| + (-1)^{m+2} a_{m2} |\mathbf{A}_{m2}| + \cdots + (-1)^{m+n} a_{mn} |\mathbf{A}_{mn}|.$$

Пример. Чтобы вычислить значение определителя

$$|\mathbf{A}| = \det \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix},$$

мы воспользуемся теоремой A.9 и, разлагая по первой строке, получим

$$|\mathbf{A}| = (-1)^2 a_{11} \det \begin{bmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{bmatrix} + (-1)^3 a_{12} \det \begin{bmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{bmatrix} + (-1)^4 a_{13} \det \begin{bmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix},$$

откуда легко получить шесть слагаемых, которые мы видели ранее. Мы можем также разложить определитель по столбцу.

Подпространства, базисы и размерность

Если V — поле, то мы обозначаем через $\mathbf{V}_n$ множество всех n-ок $(v_1, \ldots, v_n)$, элементы которых лежат в V. Операции сложения и скалярного умножения в $\mathbf{V}_n$ определяются следующим образом:

$$(a_1, \ldots, a_n) + (b_1, \ldots, b_n) = (a_1 + b_1, \ldots, a_n + b_n),$$
$$s(v_1, \ldots, v_n) = (sv_1, \ldots, sv_n)$$

для s из V. Эти операции превращают $\mathbf{V}_n$ в то, что называется *векторным пространством*.

Подмножество $\mathbf{S}$ векторного пространства $\mathbf{V}_n$ называется *подпространством*, если всякий раз, когда $\mathbf{x}, \mathbf{y}$ принадлежат $\mathbf{S}$, то же верно и для их линейной комбинации, т.е. $r\mathbf{x} + s\mathbf{y}$ для любых r, s из поля V. В этой книге основные примеры пространств — нуль-пространства и пространства строк $m \times n$-матриц.

Пусть $\mathbf{A}$ есть $m \times n$-матрица. *Нуль-пространство* матрицы $\mathbf{A}$ — это множество $\mathbf{S}$ всех $\mathbf{x}$ из $\mathbf{V}_n$, таких, что $\mathbf{A}\mathbf{x}^T = \mathbf{0}^T$. Если рассматривать $\mathbf{A}$ как матрицу коэффициентов системы однородных уравнений, то нуль-пространство матрицы $\mathbf{A}$ — это множество решений соответствующих уравнений. Множество $\mathbf{S}$ является подпространством пространства $\mathbf{V}_n$, потому что, если $\mathbf{x}, \mathbf{y}$ — решения, то решением будет и любая их линейная комбинация: если r, s — произвольные элементы поля V, то

$$\mathbf{A}(r\mathbf{x}^T + s\mathbf{y}^T) = r(\mathbf{A}\mathbf{x}^T) + s(\mathbf{A}\mathbf{y}^T) = r\mathbf{0}^T + s\mathbf{0}^T = \mathbf{0}^T.$$

(Заметим, что множество векторов $\mathbf{x}$, таких, что $\mathbf{A}\mathbf{x}^T = \mathbf{b}^T$ для фиксированного $\mathbf{b} \neq \mathbf{0}$, не является подпространством, потому что если $\mathbf{A}\mathbf{x}^T = \mathbf{b}^T$, то $\mathbf{A}(\mathbf{x}^T + \mathbf{x}^T) = 2\mathbf{b}^T$, а не $\mathbf{b}^T$.)

Базис векторного пространства $\mathbf{S}$ — это множество $\{\mathbf{x}_1, \ldots, \mathbf{x}_r\}$ векторов из $\mathbf{S}$, такое, что любой вектор $\mathbf{a}$ из $\mathbf{S}$ может быть записан в виде линейной комбинации векторов $\mathbf{x}_1, \ldots, \mathbf{x}_r$, притом единственным способом. Существует эквивалентный способ определения базиса: мы говорим, что пространство $\mathbf{S}$ *натянуто* на множество $\{\mathbf{x}_1, \ldots, \mathbf{x}_r\}$ векторов из $\mathbf{S}$, если любой вектор из $\mathbf{S}$ может быть записан в виде линейной комбинации векторов $\mathbf{x}_1, \ldots, \mathbf{x}_r$ (не обязательно однозначно), и мы говорим, что векторы $\mathbf{x}_1, \ldots, \mathbf{x}_r$ *линейно независимы*, если единственным решением уравнения

$$c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \cdots + c_r\mathbf{x}_r = 0$$

является решение $c_1 = 0, c_2 = 0, \ldots, c_r = 0$. Тогда базис пространства $\mathbf{S}$ — это множество линейно независимых векторов, на которые натянуто $\mathbf{S}$.

Если $\mathbf{S}$ — нуль-пространство матрицы $\mathbf{A}$, то, как мы видели в разд. 6.2.4, базис пространства $\mathbf{S}$ находится путем приведения $\mathbf{A}$ к ступенчатому виду (по строкам) с помощью алгоритма **NS** из разд. 6.2.4.

В общем случае у пространства есть много различных базисов. Однако хорошо известная теорема утверждает, что все базисы пространства $\mathbf{S}$ содержат одно и то же число векторов. *Размерность* пространства равна числу векторов в любом базисе этого пространства. Так, размерность пространства $\mathbf{V}_n$ равна n.

Пространство строк $m \times n$-матрицы $\mathbf{A}$ — это множество всех n-ок, являющихся линейными комбинациями строк матрицы $\mathbf{A}$. Легко убедиться, что если мы выполним элементарную операцию над строками матрицы $\mathbf{A}$, то пространство строк новой матрицы совпадет с пространством строк матрицы $\mathbf{A}$. Поскольку $\mathbf{A_e}$, ступенчатая форма матрицы $\mathbf{A}$, получается последовательностью элементарных операций над строками, пространство строк матрицы $\mathbf{A_e}$ совпадет с пространством строк матрицы $\mathbf{A}$.

Легко проверяется, что ненулевые строки ступенчатой формы матрицы $\mathbf{A}$ составляют базис пространства строк матрицы $\mathbf{A}$. Поэтому число ненулевых строк в ступенчатой форме матрицы $\mathbf{A}$ равно размерности пространства строк матрицы $\mathbf{A}$. Эта размерность называется *рангом* (по строкам) матрицы $\mathbf{A}$, и хорошо известная теорема утверждает, что для $m \times n$-матрицы $\mathbf{A}$ ее ранг в сумме с размерностью ее нуль-пространства дает n, число столбцов матрицы $\mathbf{A}$.

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ