# Сложность вычислений.
# Спецкурс 2011–2012 уч. г.

*проф. Гашков С. Б., к.ф.-м.н. Сергеев И. С.*

представлена только часть материалов лекций

# Оглавление

# Программа спецкурса

**Тема 1. Аддитивные цепочки**

Возведение в степень и аддитивные цепочки. Линейные аддитивные цепочки. Методы построения аддитивных цепочек: бинарный метод, метод множителей, асимптотически наилучший $2^k$-арный метод Брауэра. Построение аддитивных цепочек для чисел вида $2^n - 1$ (метод Брауэра). Быстрые методы вычисления линейных преобразований с булевыми матрицами: метод Лупанова, метод Нечипорука. Векторные аддитивные цепочки. Метод Страуса построения векторных аддитивных цепочек. Соотношение между сложностью реализации матриц $A$ и $A^T$ векторными аддитивными цепочками. [БГФЧ, К]

**Тема 2. Простейшие арифметические схемы**

Схемы из функциональных элементов и неветвящиеся программы. Сложность и глубина схем. Стандартные схемы сложения и умножения. Минимизация глубины булевых схем для сложения (метод золотого сечения, метод Храпченко) и умножения чисел (метод компрессоров). Параллельные префиксные схемы, префиксный сумматор (метод Ладнера-Фишера). [W]

**Тема 3. Быстрые алгоритмы умножения. Дискретное преобразование Фурье**

Метод Карацубы умножения чисел. Дискретное преобразование Фурье. Теоремы Кули—Тьюки и Гуда—Томаса. Алгоритм быстрого преобразования Фурье (БПФ). Быстрое умножение чисел и многочленов (методы Шёнхаге и Штрассена). Метод Фюрера умножения чисел. [АХУ, ГС, ГЧ, Г1, F, W]

**Тема 4. Элементарные арифметические операции с числами и многочленами**

Быстрый метод деления чисел.

(ВЕСЕННИЙ СЕМЕСТР)

Метод последовательных приближений. Алгоритмы приближенного деления и извлечения квадратного корня из степенных рядов. Метод Штрассена деления многочленов с остатком. Вычисление элементарных аналитических функций степенных рядов и чисел (логарифм, экспонента, тригонометрические функции): метод Брента—Саламина. [Г1, Br, BCS]

**Тема 5. Алгоритмы, основанные на быстром умножении**

Бинарный алгоритм вычисления НОД многочленов. Лемма Лехмера. Применение принципа «деления пополам»: быстрый расширенный алгоритм вычисления НОД многочленов, быстрые алгоритмы интерполяции и вычисления значений многочлена на наборе точек. Быстрый переход между системами счисления; быстрое вычисление факториала (методы Шёнхаге). [АХУ, ГЧ, Г1, BCS, GG]

**Тема 6. Быстрое умножение матриц**

Билинейные алгоритмы умножения матриц. Метод Штрассена. Метод приближенных разложений. Теорема Бини—Шёнхаге. Пример Шёнхаге, основанный на приближенном билинейном алгоритме умножения матриц размера $3 \times 3$. Тау-теорема Шёнхаге. Построение алгоритма умножения $n \times n$ матриц сложности $O(n^{2{,}55})$. [A, K, BCS]

**Тема 7. Арифметика конечных полей**

Конечные поля. Стандартные и нормальные базисы конечных полей. Умножение в конечном поле. Модулярная композиция многочленов и реализация автоморфизмов Фробениуса в стандартных базисах конечных полей (метод Брента—Кунга). Переходы между нормальными и стандартными представлениями элементов. Инвертирование в конечном поле: метод аддитивных цепочек. [БГФЧ, ГЧ, GG]

## Литература

[A] Алексеев В. Б. Сложность умножения матриц // Кибернетический сборник. Вып. 25. — М.: Мир, 1988, 189–236.

[АХУ] Ахо А., Хопкрофт Дж., Ульман Дж. Проектирование и анализ вычислительных алгоритмов. — М.: Мир, 1979.

[БГФЧ] Болотов А. А., Гашков С. Б., Фролов А. Б., Часовских А. А. Элементарное введение в эллиптическую криптографию: алгебраические и алгоритмические основы. — М.: КомКнига, 2006.

[ГС] Гашков С. Б., Сергеев И. С. Алгоритмы быстрого преобразования Фурье // Сборник «Дискретная математика и ее приложения». Часть V. — М.: Изд-во Института прикладной математики РАН, 2009, 3–23.

[ГЧ] Гашков С. Б., Чубариков В. Н. Арифметика. Алгоритмы. Сложность вычислений. — М.: Изд-во МГУ, Дрофа, 2005.

[Г1] Гашков С. Б. Занимательная компьютерная арифметика. В 2-х тт. М.: ЛИБРОКОМ, 2012.

[K] Кнут Д. Искусство программирования. Т. 2. Получисленные алгоритмы. — М.: Вильямс, 2000–2008.

[Br] Brent R. Multiple-precision zero-finding methods and the complexity of elementary function evaluation // Analytic computational complexity. — NY.: Academic Press, 1975, 151–176.

[BCS] Bürgisser P., Clausen M., Shokrollahi M. A. Algebraic complexity theory. — Berlin—Heidelberg: Springer-Verlag, 1997.

[F] Fürer M. Faster integer multiplication // SIAM J. Comput. — 2009. — Vol. 39(3), 979–1005.

[GG] von zur Gathen J., Gerhard J. Modern computer algebra. — Cambridge University Press, 1999, 2003.

[W] Wegener I. The complexity of boolean functions. — Stuttgart: Wiley, 1987.

# Глава 1

# Аддитивные цепочки

Рассмотрим следующую задачу. Пусть задано число $x$, которое требуется возвести в некоторую натуральную степень $n$. Как обойтись при этом наименьшим числом умножений? Первым шагом очевидно станет получение $x^2$. Далее, мы можем получить $x^3$, перемножив $x$ и $x^2$, либо $x^4$ возведением в квадрат $x^2$. И т.д. В итоге выписывается последовательность

$$x,\ x^{a_1},\ x^{a_2}, \ldots,\ x^n,$$

из степеней $x$, которая оканчивается на $x^n$. Можно не переписывать все время $x$, и оставить в записи только показатели степени — они образуют аддитивную цепочку.

*Аддитивной цепочкой* для числа $n$ называется любая начинающаяся с 1 последовательность натуральных чисел $a_0 = 1,\ a_1,\ \ldots,\ a_m = n$, в которой каждое число является суммой каких-то двух предыдущих чисел (возможно совпадающих), т.е. для всех $i \geq 1$ выполнено $a_i = a_j + a_k,\ j, k < i$. Под *длиной* цепочки $a_0,\ a_1,\ \ldots,\ a_m$ понимается число $m$. Через $l(n)$ обозначим длину *кратчайшей* аддитивной цепочки для $n$.

Таким образом, возведение в степень интерпретируется как построение аддитивной цепочки для показателя степени. *Удвоения* в аддитивной цепочке соответствуют возведениям в квадрат, а прочие сложения — умножениям.

Аддитивная цепочка называется *линейной*, если каждый ее элемент равен сумме предыдущего элемента и какого-то еще, т.е. для всех $i \geq 1$ выполнено $a_i = a_{i-1} + a_j,\ j < i$. Длина кратчайшей линейной цепочки обозначается через $l^*(n)$.

Очевидно следующее соотношение:

$$l^*(n) \geq l(n) \geq \lambda(n),$$

где $\lambda(n) = \lfloor \log_2 n \rfloor$, поскольку $2^k$ — это максимальное число, которое можно получить при помощи аддитивной цепочки длины $k$.

## Бинарный метод

Пусть $n = [n_{k-1}, n_{k-2}, \ldots, n_0]$. Используя схему Горнера, можно записать формулу

$$n = (\ldots(2n_{k-1} + n_{k-2})2 + \ldots + n_1)2 + n_0,$$

по которой выписывается универсальная аддитивная цепочка для числа $n$ (вычисления производятся слева направо)

$$a_0 = n_{k-1} = 1,\ a_1 = 2a_0 = 2,\ a_2 = a_1 + n_{k-2},\ a_3 = 2a_2,\ \ldots\ ,$$
$$a_{2k-4} = a_{2k-5} + n_1,\ a_{2k-3} = 2a_{2k-4},\ a_{2k-2} = a_{2k-3} + n_0.$$

Удалив из построенной цепочки повторяющиеся элементы, получим окончательно цепочку, соответствующую так называемому бинарному (перебирающему разряды слева направо) алгоритму.

Например, для числа $n = 19 = [10011]$ выписывается цепочка

$$1,\ 2,\ 2,\ 4,\ 4,\ 8,\ 9,\ 18,\ 19.$$

Удаляя из нее повторяющиеся элементы, получаем цепочку

$$1,\ 2,\ 4,\ 8,\ 9,\ 18,\ 19.$$

Чтобы сразу построить цепочку без повторений, можно воспользоваться следующим правилом. Удалим из двоичной записи числа $n$ единицу в старшем разряде, перед остальными единицами вставим двойки, а все нули заменим на двойки. В итоге получится слово из символов 1 и 2, в котором единицы означают прибавление 1, а двойки — удвоение. Например, для числа $n = 19 = [10011]$ получается слово 222121, которому соответствует приведенная выше аддитивная цепочка.

Пусть $\nu(n)$ обозначает вес числа $n$, т.е. количество единиц в двоичной записи.

**Лемма 1.1.** *Длина бинарной аддитивной цепочки для числа $n$ равна*

$$\lambda(n) + \nu(n) - 1.$$

*Доказательство.* Заметим, что в описанном выше бинарном методе используется $k - 1 = \lambda(n)$ удвоений и столько прибавлений единицы, сколько ненулевых разрядов в двоичной записи числа $n$, не считая старшего, а именно $\nu(n) - 1$. Лемма доказана.

Следовательно, доказано

$$l(n) \le \lambda(n) + \nu(n) - 1.$$

Бинарный метод не является оптимальным, что видно из следующего примера. Пусть $n = s\,t$, и построены аддитивные цепочки для $s$ и $t$

$$1,\ a_1,\ \ldots,\ a_i = s; \qquad 1,\ b_1,\ \ldots,\ b_j = t.$$

Тогда для $n$ выписывается следующая аддитивная цепочка

$$1,\ a_1,\ \ldots,\ a_i = s,\ sb_1,\ sb_2, \ldots,\ sb_j = n.$$

Длина ее равна сумме длин аддитивных цепочек для сомножителей, т.е.

$$l(st) \le l(s) + l(t).$$

Если для вычисления $s$ и $t$ используются бинарные цепочки, то длина цепочки для $n$ составит $\lambda(s) + \lambda(t) + \nu(s) + \nu(t) - 2$. При $n = 15$ этим способом впервые улучшается результат бинарного метода (с 6 до 5).

Можно однако доказать, что при $n \le 14$ и вообще для всех $n$ с весом $\nu(n) \le 3$ бинарный метод все же дает кратчайшую цепочку.

## Асимптотически наилучший метод

Следующий метод еще называется $2^k$-арным методом Брауэра.

**Теорема 1.1** (Брауэр, 1939)**.**

$$l(n) \le \lambda(n) + (1 + o(1))\frac{\lambda(n)}{\lambda(\lambda(n))}.$$

*Доказательство.* Пусть $k \in \mathbb{N}$, $t = \lfloor \log_{2^k} n \rfloor$. Запишем число $n$ в системе счисления с основанием $2^k$:

$$n = n_{t-1}2^{k(t-1)} + n_{t-2}2^{k(t-2)} + \ldots + n_0.$$

Перепишем, используя схему Горнера:

$$n = (\ldots(2^k n_{t-1} + n_{t-2})2^k + \ldots)2^k + n_0.$$

Рассмотрим следующую аддитивную цепочку:

$$1,\ 2,\ 3,\ \ldots,\ 2^k - 1,\ 2n_{t-1},\ 4n_{t-1},\ \ldots,\ 2^k n_{t-1},\ 2^k n_{t-1} + n_{t-2},\ \ldots\ldots,\ n.$$

Эта цепочка имеет длину $2^k - 2 + (k+1)(t-1)$. При $k = \lambda(\lambda(n)) - 2\lambda(\lambda(\lambda(n)))$ (полагая, что $n \ge 4$) имеем:

$$\begin{aligned} 2^k - 2 + (k+1)(t-1) &\le c_1\frac{\lambda(n)}{\lambda^2(\lambda(n))} + (k+1)\left(\frac{\lambda(n)}{k} + c_2\right) \le \\ &\le \lambda(n) + \frac{\lambda(n)}{k} + c_3\frac{\lambda(n)}{\lambda^2(\lambda(n))} = \lambda(n) + (1+o(1))\frac{\lambda(n)}{\lambda(\lambda(n))}. \end{aligned}$$

Теорема доказана.

Метод Брауэра является асимптотически наилучшим в силу доказанной П. Эрдошем нижней оценки длины аддитивной цепочки

$$l(n) \geq \lambda(n) + (1 - o(1))\frac{\lambda(n)}{\lambda(\lambda(n))},$$

справедливой для почти всех $n$ при $n \to \infty$.

В отношении абсолютной нижней границы длины аддитивной цепочки известна гипотеза

$$l(n) \geq \lambda(n) + \lceil \log_2 \nu(n) \rceil.$$

В 1975 г. А. Шёнхаге доказал лишь чуть-чуть более слабую оценку

$$l(n) \geq \log_2 n + \log_2 \nu(n) - 2,13.$$

## Аддитивные цепочки для чисел вида $2^n - 1$

Представляет интерес построение аддитивных цепочек для чисел специального вида. Известная недоказанная гипотеза Шольца—Брауэра гласит:

$$l(2^n - 1) \leq n - 1 + l(n).$$

Задача построения коротких аддитивных цепочек для чисел вида $2^n - 1$ является актуальной в наше время — к ней, например, сводится задача инвертирования в конечном поле характеристики 2. Гипотеза Брауэра сравнительно просто доказывается для линейных цепочек.

**Теорема 1.2** (Брауэр, 1939)**.**

$$l^*(2^n - 1) \leq n - 1 + l^*(n).$$

*Доказательство.* По произвольной линейной цепочке

$$1,\ a_1,\ \ldots,\ a_m = n$$

строится аддитивная цепочка для $2^n - 1$ следующим образом. Выписывается последовательность

$$1 = 2^1 - 1,\ 2^{a_1} - 1,\ \ldots,\ 2^{a_m} - 1 = 2^n - 1,$$

затем в промежутки между числами вставляются последовательности удвоений. Так, между соседними числами $2^{a_k} - 1$ и $2^{a_{k+1}} - 1$, где $a_{k+1} = a_k + a_j$, помещается последовательность из $a_j$ элементов

$$2(2^{a_k} - 1),\ 2^2(2^{a_k} - 1),\ \ldots,\ 2^{a_{k+1} - a_k}(2^{a_k} - 1) = 2^{a_{k+1}} - 2^{a_j}.$$

Поскольку $2^{a_{k+1}} - 1 = (2^{a_{k+1}} - 2^{a_j}) + (2^{a_j} - 1)$, то итоговая последовательность является линейной аддитивной цепочкой.

Покажем, что суммарное количество элементов во всех вставках составляет $n - 1$. Для этого докажем по индукции, что число вставленных перед $2^{a_i} - 1$ элементов равно $a_i - 1$.

Для $i = 0$ проверяемое утверждение очевидно выполнено: перед элементом $2^{a_0} - 1 = 1$ мы ничего не вставляем. Предположим, что оно выполнено для всех $i \le k$ и пусть $a_{k+1} = a_k + a_j$. Тогда число вставленных перед $2^{a_k} - 1$ элементов равно $a_k - 1$. Складывая это число с числом $a_j$ вставляемых между $2^{a_k} - 1$ и $2^{a_{k+1}} - 1$ элементов, получаем, что всего $a_k - 1 + a_j = a_{k+1} - 1$ элементов вставляется перед числом $2^{a_{k+1}} - 1$.

Таким образом, построенная цепочка для $2^n - 1$ имеет длину $n + m - 1$. Выбирая $m = l^*(n)$, приходим к утверждению теоремы.

Анализируя доказательство, можно заметить, что оно проходит для цепочек более общего вида, чем линейные. Такие цепочки называются цепочками Ханзена.

Известно, что существуют $n$ такие, что $l(n) < l^*(n)$ (наименьшее такое число — 12509), т.е. не всегда удается найти среди кратчайших цепочек линейную. Этот факт доказан Ханзеном, причем в доказательстве используются цепочки Ханзена. В 2005 г. было обнаружено число $n = 5784689$, для которого среди кратчайших цепочек нет даже цепочки Ханзена.

## Векторные аддитивные цепочки

Рассмотрим два многомерных обобщения задачи о возведении в степень за минимальное число умножений.

1) Требуется вычислить $x_1^{n_1} x_2^{n_2} \cdot \ldots \cdot x_k^{n_k}$, исходя из $x_1, \ldots, x_k$.

2) Требуется вычислить $x^{n_1}, x^{n_2}, \ldots, x^{n_k}$, зная $x$.

(В обоих случаях предполагается, что все $n_i \ne 0$.)

Для работы с первой задачей вводится концепция *векторной $k$-мерной аддитивной цепочки*, состоящей из векторов, в которых $i$-я компонента соответствует показателю степени при $x_i$. Такая цепочка определяется по аналогии с обычной: она начинается с $k$ базисных единичных векторов, а каждый следующий вектор равен сумме каких-либо двух предыдущих. Длина кратчайшей цепочки для вектора $(n_1, \ldots, n_k)$ обозначается через $l([n_1, \ldots, n_k])$, базисные вектора при подсчете длины не учитываются.

**Теорема 1.3** (Страус). *Пусть $n = \max n_i$. Тогда*

$$l([n_1, \ldots, n_k]) \le \lambda(n) + (1 + o(1)) \frac{k\lambda(n)}{\lambda(\lambda(n))}.$$

*Доказательство.* Для доказательства воспользуемся обобщением $2^k$-арного метода на многомерный случай.

Обозначим $\vec{n} = (n_1, \ldots, n_k)$, и запишем этот вектор в системе счисления с основанием $2^s$:

$$\vec{n} = \vec{d}_{t-1}2^{s(t-1)} + \vec{d}_{t-2}2^{s(t-2)} + \ldots + \vec{d}_0 = (\ldots(2^s\vec{d}_{t-1} + \vec{d}_{t-2})2^s + \ldots)2^s + \vec{d}_0,$$

где $t = \lfloor \log_{2^s} n \rfloor$, а компоненты всех векторов $\vec{d}_i$ не превосходят $2^s - 1$.

Аддитивная цепочка для $\vec{n}$ начинается с выписывания всевозможных векторов $d\vec{e}_i$, где $\vec{e}_i$ — базисные вектора, а $2 \le d \le 2^s - 1$ (всего $k(2^s - 2)$ штук). Затем из них образуются вектора $\vec{d}_i$ (для чего требуется не более $t(k-1)$ шагов). Еще $(s+1)(t-1)$ шагов требуется, чтобы завершить вычисление $\vec{n}$ по схеме Горнера. Итого, построенная цепочка имеет длину не более

$$t(k+s) + 2^s k - (2k + s + 1) \le \lambda(n) + \frac{\lambda(n)}{s}k + 2^s k.$$

Утверждение теоремы получается при выборе $s = \lambda(\lambda(n)) - 2\lambda(\lambda(\lambda(n)))$.
*Замечание.* Остаточный член в формулировке теоремы является не лучшим из возможных. Неулучшаемый остаточный член выглядит как $O(k) + (1 + o(1))\frac{R}{\log_2 R}$, где $R = \log_2(n_1 \cdot \ldots \cdot n_k)$.

Рассмотрим вторую задачу, которая сводится к построению (одномерной) аддитивной цепочки, содержащей числа $n_1, \ldots, n_k$. Длину кратчайшей из таких аддитивных цепочек обозначим через $l(n_1, \ldots, n_k)$.

Задачи 1 и 2 являются двойственными друг другу в смысле, который будет разъяснен ниже, а величины $l([n_1, \ldots, n_k])$ и $l(n_1, \ldots, n_k)$ связаны следующим соотношением.

**Теорема 1.4** (Пиппенджер, Оливос)**.**

$$l([n_1, \ldots, n_k]) = l(n_1, \ldots, n_k) + k - 1.$$

Для доказательства нам удобно рассмотреть задачу 3, обобщающую задачи 1 и 2:

3) Требуется вычислить набор мономов $x_1^{a_{i,1}} x_2^{a_{i,2}} \cdot \ldots \cdot x_p^{a_{i,p}}$, $i = 1, \ldots, q$, исходя из $x_1, \ldots, x_p$.

О решении этой задачи будем говорить как о реализации матрицы $A = (a_{i,j})$ векторными аддитивными цепочками.

## Лемма о сложности транспонированного отображения

Предварительно напомним определение схемы из функциональных элементов (СФЭ). Пусть задано множество функций $B$, аргументы и значения которых принадлежат множеству $M$. СФЭ над базисом $B$ — это ориентированный граф без ориентированных циклов с вершинами-входами, которым приписаны символы переменных или константы, и функциональными элементами в других вершинах;

некоторые вершины отмечены как выходы. Входы и выходы элементов схемы принимают значения в $M$, а сами функциональные элементы реализуют функции из базиса $B$. Более подробное определение и доказательство корректности приводится в соответствующих курсах. *Сложностью* схемы $S$ называется число функциональных элементов в ней и обозначается $L(S)$, а *глубиной* — максимальное число элементов в цепочке, ведущей от входа к выходу схемы, которое обозначается $D(S)$. Сложность (глубина) функции $f$ определяется как минимальная сложность (глубина) схемы, реализующей данную функцию. Обозначения: $L(f)$ и $D(f)$.

Рассмотрим оператор $AX$ линейного отображения с целочисленной матрицей $A$ размера $p \times q$ ($p$ столбцов, $q$ строк) над базисом $\{+\}$ (здесь "+" — ассоциативная и коммутативная операция).

**Лемма 1.2** (Митягин, Садовский, 1965)**.** $L(AX) = L(A^T X) + p - q$.

*Доказательство.* Пусть схема $S$ реализует $AX$, где $X = (x_1, \ldots, x_p)$ — вектор входов схемы. Через $Y = (y_1, \ldots, y_q)$ обозначим выходы схемы.

Можно проверить, что число (ориентированных) путей в схеме, соединяющих вход $x_i$ с выходом $y_j$, равно соответствующему элементу $a_{i,j}$ матрицы $A$. Для этого достаточно доказать по индукции, что $i$-я компонента вектора $f(e)$, вычисляемого в произвольной вершине $e$ схемы, равна числу путей $\rho(e, x_i)$, соединяющих эту вершину с $i$-м входом. Если $e = x_j$ (основание индукции), то утверждение очевидно. Если утверждение верно для вершин $e_1$ и $e_2$, которые являются входами для вершины $e$, то оно верно и для $e$, т.к. $f(e) = f(e_1) + f(e_2)$ и $\rho(e) = \rho(e_1) + \rho(e_2)$.

Пусть $r$ и $v$ — соответственно число ребер и вершин в схеме $S$. Тогда $L(S) = r - v + p$ (поскольку $r = 2L(S)$ и $L(S) = v - p$). Преобразуем схему $S$ к схеме $S'$, вычисляющей $A^T Y$.

Сначала устраним случаи использования выходов схемы в качестве входов для других ее элементов. Для этого выпустим из таких выходов висячие ребра, и перенесем выходы на свободные концы этих ребер.

После этого обратим ориентацию ребер схемы. Вершины $Y$ становятся входами, а $X$ — выходами в новой схеме. В этой схеме могут оказаться вершины, в которые входит только одно ребро. Удалим такие ребра, совместив концы каждого из них (если одним из концов ребра был вход (выход), то входом (выходом) становится совмещенная вершина). Также в схеме могут оказаться вершины, в которые входит пучок из более чем двух ребер. Такие пучки заменим эквивалентными бинарными деревьями на тех же входах. Окончательно получим схему $S'$, в которой в каждую вершину кроме входов ведут по два ребра.

Обратим внимание, что во-первых, число путей, соединяющих вершины $x_i$ и $y_j$, не изменяется при всех преобразованиях. Следовательно, схема $S'$ реализует матрицу $A^T$. Во-вторых, разница между числом ребер и вершин остается постоянной (по существу, достаточно убедиться, что при замене пучка с $t$ входами

бинарным деревом мы добавляем в схему $t-2$ новых ребра и столько же новых вершин). Как следствие, $L(S') = r - v + q = L(S) + q - p$. Лемма доказана.

### Схемная интерпретация аддитивной цепочки

Рассмотрим $k$-мерную аддитивную цепочку с $q$ выходами. Она вычисляет некоторую матрицу $A$ размера $q \times k$. Цепочку можно изобразить графически в виде ориентированного графа, вершинам которого для удобства приписаны символы элементов цепочки $a_i$. В вершину, которой приписан символ $a_i$, идут ребра от вершин с символами $a_j$, $a_k$, где $a_i = a_j + a_k$ (в случае неоднозначности разложения выбирается произвольное из возможных представлений).

Построенный граф можно интерпретировать как схему. Для этого тем вершинам графа, которым приписаны символы базисных единичных векторов, припишем символы переменных $x_i$. Вершины графа интерпретируются как функциональные элементы, реализующие операцию сложения.

Таким образом, построена схема над базисом $\{+\}$, реализующая линейное целочисленное преобразование $AX$, причем сложность схемы совпадает с длиной цепочки.

Обращая проведенное рассуждение, замечаем обратное: некоторой схеме, вычисляющей преобразование $AX$, соответствует цепочка, вычисляющая матрицу $A$, и имеющая длину, равную сложности схемы.

В силу установленной выше двойственности задач реализации матриц $A$ и $A^T$ линейными схемами, двойственность имеет место и для аддитивных цепочек. Таким образом, получаем

**Следствие 1.3.** *Пусть $A$ — матрица размера $p \times q$. Тогда*

$$l(A) = l(A^T) + p - q.$$

Из этого следствия вытекает теорема 4.

## Дополнительные вопросы

1. Дать конструктивное определение аддитивной цепочке Ханзена.
2. Доказать теорему 1 для линейных цепочек.
3. Доказать, что предполагаемая нижняя граница длины аддитивной цепочки $\lambda(n) + \lceil \log_2 \nu(n) \rceil$ не может быть повышена. Для любого $v \in \mathbb{N}$ предъявить число $n$ веса $v$ и аддитивную цепочку для $n$ длины $\lambda(n) + \lceil \log_2 v \rceil$.

# Глава 2

# Сумматоры

Предварительно напомним определение схемы из функциональных элементов (СФЭ). Пусть задано множество функций $B$, аргументы и значения которых принадлежат множеству $M$. СФЭ над базисом $B$ — это ориентированный граф без ориентированных циклов с вершинами-входами, которым приписаны символы переменных или константы, и функциональными элементами в других вершинах; некоторые вершины отмечены как выходы. Входы и выходы элементов схемы принимают значения в $M$, а сами функциональные элементы реализуют функции из базиса $B$. Более подробное определение и доказательство корректности приводится в соответствующих курсах. *Сложностью* схемы $S$ называется число функциональных элементов в ней и обозначается $L(S)$, а *глубиной* — максимальное число элементов в цепочке, ведущей от входа к выходу схемы, которое обозначается $D(S)$. Сложность (глубина) функции $f$ определяется как минимальная сложность (глубина) схемы, реализующей данную функцию. Обозначения: $L(f)$ и $D(f)$.

Рассмотрим задачу построения $n$-разрядного сумматора — схемы сложения двоичных $n$-разрядных чисел

$$A = [a_{n-1}, a_{n-2}, \ldots, a_0] \text{ и } B = [b_{n-1}, b_{n-2}, \ldots, b_0]$$

(схема строится над базисом двуместных булевых функций). Сумму чисел $A$ и $B$ обозначим через $Z = [z_n, z_{n-1}, \ldots, z_0]$.

**Лемма 2.1.** *Можно построить $n$-разрядный сумматор $S_n$, такой, что*

$$L(S_n) = 5n - 3; \qquad D(S_n) = 2n - 1.$$

*Доказательство.* Построим схему, реализующую известный метод сложения «столбиком». Введем обозначения $x_i = a_i + b_i$ («+» в булевых выражениях будет означать сумму по модулю 2) и $y_i = a_i b_i$. Метод заключается в том, что на каждом шаге вычисляется очередной разряд числа $Z$ по формуле $z_i = x_i + c_i$, где $c_i$ — перенос из младших разрядов, и следующий перенос $c_{i+1} = y_i + x_i c_i$.

Все $x_i$ и $y_i$ вычисляются со сложностью $2n$ и глубиной 1. По 3 операции необходимо для вычисления каждой пары $z_i$ и $c_{i+1}$, за исключением самой первой, т.к. $z_0 = x_0$ и $c_1 = y_0$, и старшего разряда $z_n = c_n$. Легко видеть, что $c_1$ вычисляется на глубине 1, $c_2$ — на глубине 3 и т.д., и окончательно $c_n = z_n$ вычисляется на глубине $2n-1$. Следовательно,

$$L(S_n) = 2n + 3(n-1) = 5n - 3; \qquad D(S_n) = 2n - 1.$$

Н. П. Редькин в 1981 г. показал, что построенная схема — минимальная по сложности (доказательство относится к числу наиболее сложных в классе нижних оценок сложности конкретных функций).

Естественным образом возникает задача минимизации глубины схемы сумматора. Для ее решения достаточно уметь реализовывать функции переноса $c_i$ с небольшой глубиной. Для всех $i$ справедлива формула

$$c_i = y_{i-1} + x_{i-1}(y_{i-2} + x_{i-2}(\ldots(y_1 + x_1 y_0)\ldots)).$$

Введем обозначение:

$$F_i(y_{i-1}, x_{i-1}, \ldots, x_1, y_0) = y_{i-1} + x_{i-1}(y_{i-2} + x_{i-2}(\ldots(y_1 + x_1 y_0)\ldots)).$$

Переменные $x_i$ и $y_i$ здесь являются независимыми.

Введем также сокращения:

$$F_i(k) = F_i(y_{k+i-1}, x_{k+i-1}, \ldots, x_{k+1}, y_k), \qquad F_i = F_i(0).$$

# Метод золотого сечения

**Лемма 2.2.**

$$F_n = F_k(n-k) + x_{n-1} \cdot \ldots \cdot x_{n-k} F_{n-k}.$$

Для доказательства достаточно раскрыть внешние $k-1$ скобок в определении $F_i$.

Пусть $\{\Phi_i\} = 1, 1, 2, 3, 5, 8, \ldots$ — последовательность Фибоначчи, в которой $\Phi_i = \lfloor \varphi^i/\sqrt{5} \rceil$ и $\varphi = (\sqrt{5}+1)/2$ — пропорция золотого сечения ($\lfloor x \rceil$ обозначает ближайшее целое к числу $x$). Справедлива лемма

**Лемма 2.3.** $D(F_{\Phi_m}) \le m - 1$.

*Доказательство.* Воспользуемся индукцией. Непосредственно проверяется, что $D(F_{\Phi_2}) = 0$ и $D(F_{\Phi_3}) = 2$.

Пусть выполнено $D(F_{\Phi_m}) \le m-1$ и $D(F_{\Phi_{m-1}}) \le m-2$. Заметим дальше, что конъюнкция $n$ переменных реализуется с глубиной $\lceil \log_2 n \rceil$. В частности, конъюнкция $\Phi_m$ переменных реализуется с глубиной не более $m-2$, т.к. $\Phi_i \le 2^{i-2}$ для всех $i \ge 2$.

Окончательно, соотношение $D(F_{\Phi_{m+1}}) \le m$ следует из предыдущей леммы при выборе параметров $n = \Phi_{m+1}$ и $k = \Phi_m$, т.к. $\Phi_{m+1} = \Phi_m + \Phi_{m-1}$.

**Следствие 2.4.**

$$D(F_n) \leq \lceil \log_\varphi(\sqrt{5}n) \rceil - 1 < \log_\varphi n + 1,68.$$

Оценка следует из того, что $n$ не превосходит числа Фибоначчи с индексом $\lceil \log_\varphi(\sqrt{5}n) \rceil$.

Оценим сложность метода. Обозначим $L(n,d)$ сложность реализации всех функций $F_i(y_{i-1},\ldots,y_0)$ для $i \leq n$ одной схемой с глубиной $d$.

**Лемма 2.5.**

$$L(\Phi_m, m-1) \leq (m+1)\Phi_{m+1}.$$

Будем строить схему, вычисляющую не только все $F_i$, но и все конъюнкции $x_{i-1}\cdot\ldots\cdot x_0$, $i=1,\ldots,n$ (ее сложность обозначим через $L_n$). Докажем, что $L_{\Phi_m} \leq (m+1)\Phi_{m+1}$. Очевидно, это верно при $m=1,2$.

Рассмотрим индуктивный переход. При $n=\Phi_{m+1}$ воспользуемся схемами, реализующими

$$F_i, \quad i=1,\ldots,\Phi_{m-1},$$

$$x_{i-1}\cdot\ldots\cdot x_0, \quad i=1,\ldots,\Phi_{m-1},$$

$$F_i(\Phi_{m-1}), \quad i=1,\ldots,\Phi_m,$$

$$x_{i+\Phi_{m-1}-1}\cdot\ldots\cdot x_{\Phi_{m-1}}, \quad i=1,\ldots,\Phi_m,$$

Недостающие функции $F_{\Phi_{m-1}+1},\ldots,F_{\Phi_{m+1}}$ вычисляются при помощи леммы 2, для чего требуется дополнительно $2\Phi_m$ функциональных элементов. Еще $\Phi_m$ функциональных элементов требуется для вычисления оставшихся конъюнкций. Имеем,

$$L_{\Phi_{m+1}} \leq L_{\Phi_m} + L_{\Phi_{m-1}} + 3\Phi_m \leq (m+1)\Phi_{m+1} + m\Phi_m + 3\Phi_m \leq \\ \leq (m+2)\Phi_{m+2} - \Phi_{m+1} + \Phi_m < (m+2)\Phi_{m+2}.$$

Лемма доказана.

Как следствие, получаем, что сложность реализации набора функций $F_i$, $i \leq n$ в методе золотого сечения составляет $O(n\log n)$. Значит, справедливо

**Следствие 2.6.** *Можно реализовать $n$-разрядный сумматор схемой сложности $O(n\log n)$ и глубины $\log_\varphi n + O(1)$.*

# Метод Храпченко

Метод В. М. Храпченко показывает, что на самом деле глубина сложения $n$-разрядных чисел составляет асимптотически $\log_2 n$.

Пусть $n = k_1 + \ldots + k_r$, а $m_i = k_{i+1} + \ldots + k_r$. Итерируя формулу леммы 2, получаем формулу

$$F_n = F_{k_1}(m_1) + x_{n-1} \cdot \ldots \cdot x_{m_1} \cdot \\ \cdot (\ldots (F_{k_{r-1}}(m_{r-1}) + x_{m_{r-2}-1} \cdot \ldots \cdot x_{m_{r-1}} F_{k_r}) \ldots), \quad (\star)$$

и далее, раскрывая скобки:

$$F_n = F_{k_1}(m_1) + x_{n-1} \cdot \ldots \cdot x_{m_1} F_{k_2}(m_2) + \ldots \ldots \\ \ldots \ldots + x_{n-1} \cdot \ldots \cdot x_{m_{r-2}} F_{k_{r-1}}(m_{r-1}) + x_{n-1} \cdot \ldots \cdot x_{m_{r-1}} F_{k_r}. \quad (*)$$

**Лемма 2.7.** *Пусть* $C_l^2 < m \leq C_{l+1}^2$. *Тогда*

$$D(F_{2^m}) \leq m + l + 1.$$

*Доказательство.* Докажем неравенство $D(F_{2^{C_l^2}}) \leq C_{l+1}^2$, очевидно справедливое при $l = 2$. Для индуктивного перехода применяется формула $(*)$ с параметрами $n = 2^{C_{l+1}^2}$, $r = 2^l$ и $k_1 = \ldots = k_r = 2^{C_l^2}$. Слагаемые формулы $(*)$ реализуются с глубиной $C_{l+1}^2 + 1$, т.к. $D(F_{k_j}) \leq C_{l+1}^2$ по предположению, а для оценки глубины конъюнкции $k_1 + \ldots + k_j$ переменных используется неравенство $\log_2(k_1 + \ldots + k_j) < C_l^2 + l = C_{l+1}^2$, где $j = 1, \ldots, r$. Следовательно, функция $F_n$, представленная в виде суммы $2^l$ слагаемых, вычисляется на глубине $C_{l+1}^2 + 1 + l = C_{l+2}^2$.

Теперь, для $n = 2^m$, где $C_l^2 < m \leq C_{l+1}^2$, используем $(*)$ с параметрами $r = 2^{m - C_l^2}$ и $k_1 = \ldots = k_r = 2^{C_l^2}$. Лемма доказана.

**Следствие 2.8.**

$$D(F_n) < \log_2 n + \sqrt{2 \log_2 n} + 3.$$

*Доказательство.* Заметим, что из условия $C_l^2 < m \leq C_{l+1}^2$ следует, что $l = \lceil (\sqrt{1 + 8m} - 1)/2 \rceil$. Тогда

$$D(F_n) \leq \lceil \log_2 n \rceil + \left\lceil \frac{\sqrt{1 + 8\lceil \log_2 n \rceil} - 1}{2} \right\rceil + 1 \leq \\ \leq \lceil \log_2 n \rceil + \left\lceil \sqrt{2 \log_2 n} \right\rceil + 1 < \log_2 n + \sqrt{2 \log_2 n} + 3.$$

При переходе используется справедливое при $x \geq 2$ неравенство:

$$\sqrt{1 + 8\lceil x \rceil} - 1 < \sqrt{9 + 8x} - 1 \leq 2\sqrt{2x},$$

а при $n = 2$, 3 соотношение проверяется непосредственно.

Таким образом, можно построить $n$-разрядный сумматор глубины $\log_2 n + \sqrt{2 \log_2 n} + O(1)$. Можно показать, что сложность такого сумматора составляет $O\left(c^{\sqrt{\log_2 n}} n\right)$.

## Линеаризация сложности

Пусть $n \le rk$. Положим в формуле $(\star)$ все $k_i = k$. Для $i = 0, \ldots, r-1$ введем обозначения:

$$Y_i = F_k(y_{(i+1)k-1}, \ldots, y_{ik}), \qquad X_i = x_{(i+1)k-1} \cdot \ldots \cdot x_{ik}.$$

Тогда $(\star)$ переписывается как

$$F_n(y_{n-1}, \ldots, y_0) = Y_{r-1} + X_{r-1}(Y_{r-2} + X_{r-2}(\ldots(Y_1 + X_1 Y_0)\ldots)) = F_r(Y_{r-1}, \ldots, Y_0).$$

(Если $i \ge n$, то полагается $x_i = 1$ и $y_i = 0$.)

Теперь допустим, что имеются два метода (условно назовем их методом $A$ и методом $B$) реализации набора функций $F_1, \ldots, F_s$ вместе с соответствующими конъюнкциями $x_{i-1} \cdot \ldots \cdot x_0$, $i = 1, \ldots, s$. Построим новый метод, назовем его методом $C$. Обозначим $L_I(s), D_I(s)$ сложность и глубину реализации указанного набора функций методом $I$, где $I \in \{A, B, C\}$. Основная идея состоит в том, что при надлежащем выборе параметров сложность метода $C$ будет «близка» к сложности метода $A$, а глубина — к глубине метода $B$.

Метод $C$ заключается в следующем.

**1**. Для всех $j = 0, \ldots, r-1$ реализуются наборы функций $F_l(y_{jk+l-1}, \ldots, y_{jk})$ и конъюнкции $x_{jk+l-1} \cdot \ldots \cdot x_{jk}$, $l = 1, \ldots, k$, методом $A$.

**2**. Для всех $j = 1, \ldots, r$ вычисляются функции

$$F_{jk}(y_{jk-1}, \ldots, y_0) = F_j(Y_{j-1}, \ldots, Y_0)$$

и вместе с ними конъюнкции $K_j = X_{j-1} \cdot \ldots \cdot X_0$ методом $B$.

**3**. Для каждого $j = 1, \ldots, r-1$ вычисляются все функции $F_{jk+l}(y_{jk+l-1}, \ldots, y_0)$, $l = 1, \ldots, k-1$, по формулам

$$F_{jk+l}(y_{jk+l-1}, \ldots, y_0) = F_l(y_{jk+l-1}, \ldots, y_{jk}) + x_{jk+l-1} \cdot \ldots \cdot x_{jk} F_{jk}(y_{jk-1}, \ldots, y_0)$$

и необходимые конъюнкции

$$x_{jk+l-1} \cdot \ldots \cdot x_0 = (x_{jk+l-1} \cdot \ldots \cdot x_{jk}) \cdot K_j.$$

**Лемма 2.9.** *Если* $(r-1)k < n \le rk$, *то*

$$L_C(n) \le rL_A(k) + L_B(r) + 3n, \qquad D_C(n) \le D_A(k) + D_B(r) + 2.$$

*Доказательство.* Оценки сложности и глубины построенной методом $C$ схемы складываются из суммы сложностей и глубин реализации шагов 1–3.

Шаг 1 реализуется со сложностью $rL_A(k)$ и глубиной $D_A(k)$. Шаг 2 — со сложностью $L_B(r)$ и глубиной $D_B(r)$. Шаг 3 — со сложностью $3(r-1)(k-1) \le 3n$ и глубиной 2. Складывая оценки на всех шагах, приходим к утверждению леммы.

Применим описанный способ к построению сбалансированных по сложности и глубине сумматоров. Примем за основу стандартный метод, имеющий сложность $3(n-1)$ и глубину $2(n-1)$.

А) Выберем $k, r \sim \sqrt{n}$ и стандартный метод в качестве как метода $A$, так и метода $B$. Получим метод сложности $O(n)$ и глубины $O(\sqrt{n})$.

Б) Выберем $k \sim \log n$, только что полученный метод в качестве метода $A$ и метод золотого сечения — в качестве метода $B$. Получим метод линейной сложности $O(n)$ и логарифмической глубины $\log_\varphi n + O(\sqrt{\log n})$.

В) Выберем $k \sim \sqrt{\log n}$, стандартный метод в качестве метода $A$ и только что полученный метод — в качестве метода $B$. Получим метод сложности $6n + o(n)$ и глубины $\log_\varphi n + O(\sqrt{\log n})$.

Заметим, что если не вычислять конъюнкции, то оценка сложности последнего метода понизится до $5n + o(n)$. Так как для построения $n$-разрядного сумматора схему, вычисляющую набор функций $F_i$, достаточно дополнить $3n$ функциональными элементами, получаем

**Следствие 2.10.** *Можно реализовать $n$-разрядный сумматор схемой сложности $8n + o(n)$ и глубины $\log_\varphi n + o(\log n)$.*

Аналогичный результат можно получить и для метода Храпченко. Наилучшая известная верхняя оценка глубины схемы сложения $n$-разрядных чисел получена М. И. Гринчуком и составляет $\log_2 n + \log_2 \log_2 n + O(1)$.

# Дополнительные вопросы

1. Показать, что глубина $n$-разрядного сумматора не превосходит $D(F_n) + 1$.

# Глава 3

# Дискретное преобразование Фурье

Пусть $\mathbf{K}$ — коммутативное кольцо с единицей.

Определение: элемент $\zeta \in \mathbf{K}$ является *примитивным корнем степени* $N \in \mathbb{N}$, если $\zeta^N = 1$, и никакой из элементов $\zeta^{N/p} - 1$, где $p$ — простой делитель $N$, не является делителем нуля в $\mathbf{K}$. (Напомним, что элемент $a$ называется делителем нуля, если существует ненулевой элемент $b$, такой, что $ab = 0$.)

Определение: *дискретным преобразованием Фурье* (*ДПФ*) *порядка* $N$ называется $(\mathbf{K}^N \to \mathbf{K}^N)$-преобразование

$$\text{ДПФ}_{N,\zeta}(\gamma_0, \ldots, \gamma_{N-1}) = (\gamma_0^*, \ldots, \gamma_{N-1}^*), \qquad \gamma_j^* = \sum_{i=0}^{N-1} \gamma_i \zeta^{ij}. \qquad (*)$$

где $\zeta$ — примитивный корень степени $N$.

## Основное свойство ДПФ

Фундаментальное свойство ДПФ формулируется следующим образом:

**Лемма 3.1.** *Пусть элементы* $\gamma_j^*$ *определяются из* $(*)$. *Тогда*

$$\text{ДПФ}_{N,\zeta^{-1}}(\gamma_0^*, \ldots, \gamma_{N-1}^*) = (N\gamma_0, \ldots, N\gamma_{N-1}),$$

*где под* $N$ *в правой части формулы понимается сумма* $N$ *единиц кольца.*

Перед тем, как перейти к доказательству леммы, установим несколько вспомогательных фактов.

Заметим, что если элемент $a \in \mathbf{K}$ не является делителем нуля, и $a = cd$, то множители $c$ и $d$ также не являются делителями нуля. Действительно, если, скажем, $ce = 0$ и $e \neq 0$, то $ae = (ce)d = 0$, откуда следует, что $a$ — делитель нуля.

**Лемма 3.2.** *Если* $\zeta$ *— примитивный корень степени* $N$, *то при любом* $l = 1, \ldots, N-1$

$$\sum_{i=0}^{N-1} \zeta^{il} = 0.$$

*Доказательство.* Рассмотрим разложение

$$0 = \zeta^{lN} - 1 = (\zeta^l - 1) \sum_{i=0}^{N-1} \zeta^{il}.$$

Из определения примитивного корня следует, что $N$ — это минимальный натуральный показатель степени $n$, при котором $\zeta^n = 1$, поэтому $\zeta^l - 1 \neq 0$. Следовательно, либо $\zeta^l - 1$ является делителем нуля, либо $\sum_{i=0}^{N-1} \zeta^{il} = 0$. Покажем, что первое невозможно.

Пусть $m = \text{НОД}(l, N)$. По свойству наибольшего общего делителя, существуют целые $q, s$, такие, что $m = ql + sN$, при этом можно считать, что $q$ — положительно. В таком случае $\zeta^m - 1 = \zeta^{ql} - 1$ делится на $\zeta^l - 1$. С другой стороны, поскольку $m < N$, найдется простое $p$, такое, что $m \mid (N/p)$. Тогда $(\zeta^m - 1) \mid (\zeta^{N/p} - 1)$. Окончательно, имеем $(\zeta^l - 1) \mid (\zeta^{N/p} - 1)$. Поскольку элемент $\zeta^{N/p} - 1$ не является делителем нуля, то и $\zeta^l - 1$ не может быть делителем нуля. Следовательно, $\sum_{i=0}^{N-1} \zeta^{il} = 0$. Лемма доказана.

*Доказательство леммы* 1. В векторе $\text{ДПФ}_{N,\zeta^{-1}}(\gamma_0^*, \ldots, \gamma_{N-1}^*)$ рассмотрим произвольную $j$-ю компоненту:

$$\sum_{i=0}^{N-1} \gamma_i^* \zeta^{-ij} = \sum_{i=0}^{N-1} \sum_{k=0}^{N-1} \gamma_k \zeta^{ki} \zeta^{-ij} = \sum_{k=0}^{N-1} \sum_{i=0}^{N-1} \gamma_k \zeta^{i(k-j)} = \sum_{k=0}^{N-1} \gamma_k \sum_{i=0}^{N-1} (\zeta^{k-j})^i.$$

Внутренняя сумма, как следует из леммы 2, равна нулю во всех случаях, за исключением случая $k - j = 0$, в котором эта сумма равна $N$. Поэтому, продолжая выкладку, получаем $N\gamma_j$, что и требовалось. Лемма 1 доказана.

Как следствие, получаем, что если элемент $N = 1 + \ldots + 1 \in \mathbf{K}$ обратим, то определено обратное к ДПФ преобразование

$$\text{ДПФ}_{N,\zeta}^{-1} = N^{-1} \text{ДПФ}_{N,\zeta^{-1}}.$$

# Полиномиальная интерпретация ДПФ

Рассмотрим многочлен $\Gamma(x) = \gamma_0 + \ldots + \gamma_{N-1} x^{N-1}$. Тогда, по определению,

$$\text{ДПФ}_{N,\zeta}(\gamma_0, \ldots, \gamma_{N-1}) = \left(\Gamma(\zeta^0), \ldots, \Gamma(\zeta^{N-1})\right).$$

Смысл обратного преобразования $\text{ДПФ}_{N,\zeta}^{-1}$ заключается в восстановлении коэффициентов единственного многочлена степени, меньшей $N$, имеющего заданный набор значений в точках $\zeta^0, \ldots, \zeta^{N-1}$.

Формально, связь между ДПФ и интерполяцией описывается следующей леммой:

**Лемма 3.3.** *Преобразование* $\text{ДПФ}_{N,\zeta}$ *задает изоморфизм:* $\mathbf{K}[x]/(x^N - 1) \to \mathbf{K}^N$.

*Доказательство.* Проверим, что ДПФ сохраняет операции сложения и умножения: в кольце $\mathbf{K}[x]/(x^N-1)$ эти операции выполняются как с обычными многочленами, только с последующим приведением по модулю $x^N-1$, в кольце $\mathbf{K}^N$ операции выполняются покомпонентно.

Действительно, значение суммы многочленов $\Gamma_1(x)+\Gamma_2(x)$ в некоторой точке совпадает с суммой значений каждого из многочленов в данной точке. Представляя произведение многочленов в форме $Q(x)(x^N-1)+R(x)$, где $R(x)$ — остаток от деления на $x^N-1$, убеждаемся, что произведение переходит в произведение в силу:

$$\Gamma_1(\zeta^j)\Gamma_2(\zeta^j) = Q(\zeta^j)(\zeta^{jN}-1)+R(\zeta^j) = R(\zeta^j) = (\Gamma_1\Gamma_2 \bmod (x^N-1))(\zeta^j).$$

Лемма доказана.

## Вычисление ДПФ

Независимое вычисление компонент вектора ДПФ по формулам $(*)$ может быть выполнено за $O(N^2)$ операций в кольце. Для составного числа $N$ можно предложить более эффективный способ.

Прежде заметим, что если $\zeta$ — примитивный корень степени $PQ$, то $\zeta^P$ и $\zeta^Q$ — примитивные корни степени $Q$ и $P$ соответственно (это легко проверить непосредственно из определения).

Справедлива

**Лемма 3.4** (Кули, Тьюки). *ДПФ порядка $PQ$ реализуется при помощи $P$ ДПФ порядка $Q$, $Q$ ДПФ порядка $P$ и $PQ$ операций умножения на степени $\zeta$ — примитивного корня степени $PQ$.*

*Доказательство.* Для $p=0,\ldots,P-1$ и $q=0,\ldots,Q-1$ запишем

$$\gamma^*_{pQ+q} = \sum_{I=0}^{PQ-1}\gamma_I\zeta^{I(pQ+q)} = \sum_{i=0}^{Q-1}\sum_{j=0}^{P-1}\gamma_{iP+j}\zeta^{(iP+j)(pQ+q)} =$$

$$\sum_{i=0}^{Q-1}\sum_{j=0}^{P-1}\gamma_{iP+j}\zeta^{iqP+jpQ+jq} = \sum_{j=0}^{P-1}(\zeta^Q)^{jp}\cdot\zeta^{jq}\cdot\gamma^*_{(j),q},$$

где

$$\gamma^*_{(j),q} = \sum_{i=0}^{Q-1}\gamma_{iP+j}(\zeta^P)^{iq}.$$

Полученная формула позволяет произвести вычисления в следующем порядке:
*а)* Для $j=0,\ldots,P-1$ вычисляются вектора

$$\left(\gamma^*_{(j),0},\gamma^*_{(j),1},\ldots,\gamma^*_{(j),Q-1}\right) = \text{ДПФ}_{Q,\zeta^P}(\gamma_j,\gamma_{P+j},\ldots,\gamma_{(Q-1)P+j}).$$

*б*) Вычисляются произведения $\omega_{(q),j} = \zeta^{jq} \cdot \gamma^*_{(j),q}$, $j = 0, \ldots, P-1$, $q = 0, \ldots, Q-1$.

*в*) Заметим, что

$$\gamma^*_{pQ+q} = \sum_{j=0}^{P-1} \omega_{(q),j} (\zeta^Q)^{jp}.$$

Это позволяет окончательно найти компоненты вектора ДПФ по формулам

$$\left(\gamma^*_q, \gamma^*_{Q+q}, \ldots, \gamma^*_{(P-1)Q+q}\right) = \text{ДПФ}_{P,\zeta^Q}(\omega_{(q),0}, \omega_{(q),1}, \ldots, \omega_{(q),P-1}),$$

где $q = 0, \ldots, Q-1$.

Утверждение леммы немедленно следует из вида действий, выполненных на шагах *а*–*в*.

Обозначим сложность ДПФ порядка $N$ через $F(N)$. По индукции несложно проверяется

**Следствие 3.5.**

$$F(N_1 \cdot \ldots \cdot N_r) \leq N_1 \cdot \ldots \cdot N_r \left( \frac{F(N_1)}{N_1} + \ldots + \frac{F(N_r)}{N_r} + (r-1) \right).$$

В указанной оценке сложности слагаемое $(r-1)N_1 \cdot \ldots \cdot N_r$ отвечает операциям умножения на степени примитивного корня.

В случае, когда $N$ — гладкое число, т.е. раскладывается в произведение относительно небольших сомножителей, метод леммы 3.4 также называется алгоритмом быстрого преобразования Фурье (БПФ). В наиболее важном случае $N = 2^k$ получаем

$$F(2^k) \leq N \left( \frac{k}{2} F(2) + k - 1 \right).$$

Очевидно, $F(2) \leq 3$ в силу соотношений

$$\gamma^*_0 = \gamma_0 + \gamma_1, \qquad \gamma^*_1 = \gamma_0 + \zeta\gamma_1.$$

Учитывая, что фактически $\zeta = -1$, при наличии операции вычитания указанные формулы переписываются как

$$\gamma^*_0 = \gamma_0 + \gamma_1, \qquad \gamma^*_1 = \gamma_0 - \gamma_1,$$

откуда вытекает $F(2) = 2$.

Окончательно получаем, что ДПФ порядка $2^k$ может быть вычислено за $2,5k2^k$ (или $2k2^k$ с использованием вычитаний) операций, из которых $k2^k$ — сложения (или вычитания), остальные — умножения на степени примитивного корня.

## Умножение многочленов над кольцом K

Алгоритм БПФ подходящего порядка позволяет быстро выполнять умножение многочленов из $\mathbf{K}[x]$.

**Теорема 3.1.** *Пусть для любого $k \in \mathbb{N}$ существует $\zeta_k$ — примитивный корень степени $2^k$ в кольце $\mathbf{K}$, и элемент 2 обратим в $\mathbf{K}$. Тогда сложность $M(n)$ умножения многочленов степени $n-1$ над $\mathbf{K}$ не превосходит $O(n \log n)$.*

*Доказательство.* Обозначим перемножаемые многочлены через $A(x) = \sum_{i=0}^{n-1} a_i x^i$ и $B(x) = \sum_{i=0}^{n-1} b_i x^i$. Выберем такое $k$, что $2n-1 \le 2^k < 4n-1$. Согласно условиям теоремы, в кольце $\mathbf{K}$ определено ДПФ порядка $2^k$ и обратное к нему.

Быстрый способ умножения, основанный на БПФ, состоит в следующем: вычисляются вектора

$$(a_0^*, \ldots, a_{2^k-1}^*) = \text{ДПФ}_{2^k,\zeta_k}(a_0, \ldots, a_{n-1}, 0, \ldots, 0),$$

$$(b_0^*, \ldots, b_{2^k-1}^*) = \text{ДПФ}_{2^k,\zeta_k}(b_0, \ldots, b_{n-1}, 0, \ldots, 0).$$

Затем коэффициенты многочлена $C(x) = \sum c_i x^i = A(x)B(x)$ в силу $C(x) = C(x) \bmod (x^{2^k} - 1)$ могут быть найдены как

$$(c_0, \ldots, c_{2^k-1}) = 2^{-k}\text{ДПФ}_{2^k,\zeta_k^{-1}}(a_0^* b_0^*, \ldots, a_{2^k-1}^* b_{2^k-1}^*).$$

Таким образом, для умножения используется три ДПФ порядка $2^k$, $2^k$ умножений на $2^{-k}$ и еще $2^k$ нетривиальных умножений, откуда получаем

$$M(n) \le 3F(2^k) + 2^{k+1} = O(n \log n).$$

## Дополнительные вопросы

1. Показать, что любая степень $\zeta^m$ примитивного корня $\zeta$ степени $N$ является примитивным корнем степени $N/\text{НОД}(m, N)$.
2. Уточнить оценку леммы Кули—Тьюки и, используя операцию вычитания, показать, что $F(2^k) \le 1,5k2^k$.
3. Пусть числа $P$ и $Q$ взаимно просты. Показать, что $F(PQ) \le PF(Q) + QF(P)$.

# Глава 4

# Умножение чисел. Метод Шёнхаге—Штрассена

Стандартный (машинный) метод умножения основан на известном школьном приеме: один из множителей последовательно умножается на разряды другого, соответствующим образом сдвинутые результаты записываются друг под другом и затем складываются. Умножение чисел 26 и 21 в этой интерпретации будет выглядеть так:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | 1 | 1 | 0 | 1 | 0 |
| | | | | | 1 | 0 | 1 | 0 | 1 |
| | | | | | 1 | 1 | 0 | 1 | 0 |
| | | | | 0 | 0 | 0 | 0 | 0 | |
| | | | 1 | 1 | 0 | 1 | 0 | | |
| | | 0 | 0 | 0 | 0 | 0 | | | |
| | 1 | 1 | 0 | 1 | 0 | | | | |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |

Сложность умножения $n$-разрядных чисел этим способом, как легко проверить, составляет по порядку $n^2$ операций. Квадратичный порядок сложности казался естественным для умножения до 1961 г., когда А. А. Карацуба предложил метод, в основе которого лежит идея «деления пополам»: представим $2n$-разрядные числа $a$ и $b$ в виде $a_1 2^n + a_0$ и $b_1 2^n + b_0$ соответственно, где $a_i, b_i < 2^n$. Тогда произведение $ab$ согласно формуле

$$ab = a_1 b_1 2^{2n} + ((a_1 + a_0)(b_1 + b_0) - a_1 b_1 - a_0 b_0)2^n + a_0 b_0$$

можно свести к трем умножениям чисел вдвое меньшей длины. Рекурсивное применение этого приема в конечном счете приводит к оценке сложности $O(n^{\log_2 3})$, где $\log_2 3 < 1,585$.

В 1963 г. А. Л. Тоом обобщил прием Карацубы, сведя умножение $kn$-разрядных чисел к $2k-1$ умножениям $(n+O(1))$-разрядных, что позволило получить новую

оценку сложности умножения в виде $c^{\sqrt{\log n}}n$. В основе метода Тоома лежит идея интерполяции: перехода от кодирования многочлена набором его коэффициентов к кодированию набором значений в определенных точках. Развитие идеи интерполяции привело в конце 60-х гг. к появлению быстрых алгоритмов умножения при помощи ДПФ. Подробно рассмотрим метод Шёнхаге—Штрассена, опубликованный в 1971 г. и остававшийся асимптотически наилучшим известным способом умножения вплоть до 2007 г.

## Арифметика в кольце вычетов по модулю числа Ферма

Рассмотрим кольцо $\mathbb{Z}_{2^{2^n}+1}$. Пусть элементы этого кольца представляются $2^{n+1}$-разрядными числами, а операции с ними производятся по модулю $2^{2^{n+1}} - 1 = (2^{2^n}-1)(2^{2^n}+1)$. Вычету $\overline{a} \in \mathbb{Z}_{2^{2^n}+1}$ в таком представлении может соответствовать любое из чисел

$$\{a + b(2^{2^n} + 1) \mid b \in \mathbb{Z}\} \cap \{0, \ldots, 2^{2^{n+1}} - 1\}.$$

Далее для простоты изложения в качестве элементов кольца вычетов будем использовать числа-представители эквивалентных классов (по модулю $2^{2^n} + 1$).

Заметим, что двойка является примитивным корнем степени $2^{n+1}$ в кольце $\mathbb{Z}_{2^{2^n}+1}$. Действительно, $2^{2^{n+1}} \equiv 1$, а $2^{2^n} - 1$ не является делителем нуля в силу взаимной простоты чисел $2^{2^n} \pm 1$.

**Лемма 4.1.** *Любая из операций сложения, вычитания и умножения на $2^k$ в кольце $\mathbb{Z}_{2^{2^n}+1}$ выполняется со сложностью $O(2^n)$.*

*Доказательство.* Действительно, умножение на степень двойки реализуется простым циклическим сдвигом разрядов числа-элемента кольца. Сложность циклического сдвига можно принять равной нулю, если исходить из схемной модели вычислений, либо $O(2^n)$ в случае программной реализации.

Рассмотрим операцию сложения чисел $x, y \in [0, \ldots, 2^{2^{n+1}} - 1]$. Положим $z' = x+y \leq 2(2^{2^{n+1}}-1)$; обозначим через $z'_1$ старший $2^{n+1}$-й разряд числа $z'$ (нумерация с нуля), а через $z'_0$ — число, получаемое из $z'$ удалением этого разряда. Тогда число

$$z = z'_0 + z'_1 = z' - z'_1(2^{2^{n+1}} - 1) = (x + y) \bmod \left(2^{2^{n+1}} - 1\right)$$

в кольце $\mathbb{Z}_{2^{2^n}+1}$ представляет сумму $x + y$. Ясно, что сложность вычисления $z$ по порядку совпадает со сложностью обычного сложения $2^{n+1}$-разрядных чисел и составляет, следовательно, $O(2^n)$.

Вычитание может быть сведено к умножению на $2^{2^n}$ и сложению в силу тождества $x - y \equiv x + 2^{2^n} y \bmod (2^{2^n} + 1)$. Лемма доказана.

**Следствие 4.2.** *Пусть $k \le n+1$. Сложность ДПФ порядка $2^k$ над кольцом $\mathbb{Z}_{2^{2^n}+1}$ составляет $O(k2^{n+k})$.*

Помимо арифметических операций, следует остановиться еще на двух. Первая состоит в вычислении канонического представителя класса вычетов — числа из $[0, \ldots, 2^{2^n}]$ — по некоторому представлению, что эквивалентно операции приведения по модулю $2^{2^n}+1$. Пусть $x = u + v2^{2^n}$, где $0 \le u, v < 2^{2^n}$. Тогда

$$x \bmod (2^{2^n}+1) = \begin{cases} u - v, & u \ge v, \\ 2^{2^n} + 1 + u - v, & u < v. \end{cases}$$

Ясно, что вычисления по этой формуле сводятся к вычитаниям и сложениям и могут быть выполнены со сложностью $O(2^n)$.

**Лемма 4.3.** *Пусть $0 \le \xi \le 2^{2^n}$, $0 \le \eta < 2^k$ и $k \le 2^n$. Тогда нахождение числа $z \in [0, \ldots, 2^k(2^{2^n}+1) - 1]$ по остаткам $\xi$ и $\eta$ от деления соответственно на $2^{2^n}+1$ и $2^k$ может быть реализовано со сложностью $O(2^n)$.*

*Доказательство.* Число $z$ можно вычислить по формуле

$$z = \xi + \big((\eta - \xi) \bmod 2^k\big)\big(2^{2^n}+1\big)$$

со сложностью $O(2^n)$. Несложно проверить, что $z$ находится в условленных пределах и имеет заданные остатки от деления на $2^{2^n}+1$ и $2^k$. Лемма доказана.

# Метод Шёнхаге—Штрассена

Умножение $N$-разрядных чисел можно выполнить при помощи алгоритма умножения в кольце $\mathbb{Z}_{2^{2^m}+1}$, где $2^m \ge 2N$. В методе Шёнхаге—Штрассена умножение в кольце $\mathbb{Z}_{2^{2^m}+1}$ сводится к умножениям в кольце $\mathbb{Z}_{2^{2^n}+1}$, где $n = \lceil (m+1)/2 \rceil$. Выделяются случаи нечетного и четного $m$. Рассмотрим более подробно первый случай.

Итак, пусть $m = 2n-1$. Перемножаемые $2^{m+1}$-разрядные числа (элементы кольца $\mathbb{Z}_{2^{2^m}+1}$) обозначим через $a$ и $b$. Разобьем их на блоки длины $2^{n-1}$:

$$a = \sum_{i=0}^{2^{n+1}-1} a_i 2^{i2^{n-1}}, \quad b = \sum_{i=0}^{2^{n+1}-1} b_i 2^{i2^{n-1}}, \quad 0 \le a_i, b_i < 2^{2^{n-1}}.$$

Тогда

$$ab = \sum_{i=0}^{2^{n+1}-1} c_i 2^{i2^{n-1}} \bmod (2^{2^m}+1), \quad c_i = \sum_{\substack{\rho+\sigma \equiv i \bmod 2^{n+1} \\ 0 \le \rho, \sigma < 2^{n+1}}} a_\rho b_\sigma, \tag{1}$$

поскольку

$$2^{(\rho+\sigma)2^{n-1}} \equiv 2^{(\rho+\sigma \bmod 2^{n+1})2^{n-1}} \bmod (2^{2^m}+1).$$

Очевидно, $0 \le c_i < 2^{n+1+2^n}$.

Если заметить, что

$$2^{2^n \cdot 2^{n-1}} = 2^{2^m} \equiv -1 \bmod (2^{2^m}+1),$$

то формулу (1) можно переписать в виде

$$ab = \sum_{i=0}^{2^n-1} (c_i - c_{i+2^n}) 2^{i2^{n-1}} \bmod (2^{2^m}+1) =$$
$$\sum_{i=0}^{2^n-1} \underbrace{(c_i - c_{i+2^n} + 2^{n+1+2^n})}_{z_i} 2^{i2^{n-1}} + \sum_{i=2^n}^{2^{n+1}-1} \underbrace{2^{n+1+2^n}}_{z_i} \cdot 2^{i2^{n-1}} \bmod (2^{2^m}+1) =$$
$$\sum_{i=0}^{2^{n+1}-1} z_i 2^{i2^{n-1}} \bmod (2^{2^m}+1).$$

Ясно, что $0 \le z_i < 2^{n+2+2^n}$. При $n \ge 2$ выполняется $n+2+2^n \le 2^{n+1}$, т.е. для записи числа $z_i$ достаточно $2^{n+1}$ двоичных разрядов.

Процесс вычислений распадается на две части: вычисление $z_i$ и восстановление $ab$. Рассмотрим эти части в обратном порядке.

## Часть II. Восстановление произведения из $z_i$

Пусть

$$ab = \sum_{i=0}^{2^{n+1}-1} w_i 2^{i2^{n-1}}, \quad 0 \le w_i < 2^{2^{n-1}}.$$

По существу, задача состоит в вычислении блоков $w_i$ по известным $z_i$. Произведение $ab$ можно представить в виде суммы четырех слагаемых $A_j$:

$$A_j = \sum_{i=0}^{2^{n-1}-1} z_{4i+j} 2^{i2^{n+1}}, \quad j = 0, \ldots, 3.$$

Блоки $z_i$ в каждом из этих слагаемых не пересекаются, поскольку имеют длину не более $2^{n+1}$. Поскольку сложение в кольце $\mathbb{Z}_{2^{2^m}+1}$ имеет линейную сложность (см. лемму 1), получаем для сложности второй части оценку $O(2^m)$.

## Часть I. Вычисление $z_i$

В силу $0 \le z_i < 2^{n+2}(2^{2^n}+1)$ для вычисления $z_i$ достаточно вычислить остатки от деления $z_i$ на числа $2^{n+2}$ и $2^{2^n}+1$ и затем воспользоваться леммой 2. Согласно лемме 2, сложность восстановления всех $z_i$ равна $2^n O(2^n) = O(2^{2n})$. Оценим сложность вычисления $z_i \bmod 2^{n+2}$ и $z_i \bmod 2^{2^n}+1$.

а) *Вычисление* $z_i \bmod 2^{n+2}$.

Обозначим

$$\alpha_i = a_i \bmod 2^{n+2}, \qquad \beta_i = b_i \bmod 2^{n+2}$$

и положим

$$u = \sum_{i=0}^{2^{n+1}-1} \alpha_i 2^{i(3n+5)}, \qquad v = \sum_{i=0}^{2^{n+1}-1} \beta_i 2^{i(3n+5)}.$$

Тогда

$$uv = \sum_{i=0}^{2^{n+2}-1} \gamma_i 2^{i(3n+5)}, \qquad \gamma_i = \sum_{\rho+\sigma=i} \alpha_\rho \beta_\sigma.$$

При этом в сумме, выражающей $uv$, слагаемые не накладываются друг на друга в силу

$$\gamma_i < 2^{n+1}(2^{n+2})^2 = 2^{3n+5}.$$

Поэтому все $\gamma_i$ могут быть восстановлены из $uv$ с нулевой сложностью.

По построению,

$$c_i \equiv \gamma_i + \gamma_{i+2^{n+1}} \mod 2^{n+2},$$

следовательно,

$$z_i \equiv \gamma_i + \gamma_{i+2^{n+1}} - \gamma_{i+2^n} - \gamma_{i+3\cdot 2^n} \mod 2^{n+2}. \tag{2}$$

Окончательно, сложность вычисления всех $z_i \bmod 2^{n+2}$ можно оценить сложностью умножения $(3n+5)2^{n+1}$-разрядных чисел $u$ и $v$ плюс $O(n2^n)$ операций для реализации сложений-вычитаний (2). Умножение можно реализовать, например, методом Карацубы за $O((n2^n)^{\log_2 3}) = o(2^{2n})$ операций.

б) *Вычисление* $z_i \bmod 2^{2^n}+1$.

При $0 \le i < 2^n$, учитывая, что $2^{2^n} \equiv 2^{-2^n} \equiv -1 \mod (2^{2^n}+1)$, запишем

$$\begin{aligned}
\hat{c}_i = 2^i(c_i - c_{i+2^n}) \equiv \\
\equiv \sum_{\substack{\rho+\sigma\equiv i \bmod 2^{n+1} \\ 0\le\rho,\sigma<2^{n+1}}} (2^\rho a_\rho)(2^\sigma b_\sigma) - 2^{-2^n} \sum_{\substack{\rho+\sigma\equiv i+2^n \bmod 2^{n+1} \\ 0\le\rho,\sigma<2^{n+1}}} (2^\rho a_\rho)(2^\sigma b_\sigma) \equiv \\
\equiv \sum_{\substack{\rho+\sigma\equiv i \bmod 2^{n} \\ 0\le\rho,\sigma<2^{n+1}}} (2^\rho a_\rho)(2^\sigma b_\sigma) \equiv \\
\equiv \sum_{\substack{\rho+\sigma\equiv i \bmod 2^{n} \\ 0\le\rho,\sigma<2^{n}}} (2^\rho a_\rho - 2^\rho a_{\rho+2^n})(2^\sigma b_\sigma - 2^\sigma b_{\sigma+2^n}) \mod (2^{2^n}+1).
\end{aligned}$$

Далее в этом пункте все вычисления выполняются по модулю $2^{2^n}+1$.

Положим

$$\hat{a}_i = 2^i(a_i - a_{i+2^n}), \qquad \hat{b}_i = 2^i(b_i - b_{i+2^n}).$$

Справедливо

$$\left(\sum_{i=0}^{2^n-1} \hat{a}_i x^i\right)\left(\sum_{i=0}^{2^n-1} \hat{b}_i x^i\right) = \sum_{i=0}^{2^{n+1}-1}\left(\sum_{\rho+\sigma=i} \hat{a}_\rho \hat{b}_\sigma\right) x^i \equiv$$

$$\equiv \sum_{i=0}^{2^n-1}\left(\sum_{\rho+\sigma\equiv i \bmod 2^n} \hat{a}_\rho \hat{b}_\sigma\right) x^i = \sum_{i=0}^{2^n-1} \hat{c}_i x^i \mod (x^{2^n}-1).$$

Приведенная выкладка показывает, что для нахождения $\hat{c}_i$ можно воспользоваться ДПФ порядка $2^n$. Вычисляем:

$$(a_0^*, \ldots, a_{2^n-1}^*) = \text{ДПФ}_{2^n,4}(\hat{a}_0, \ldots, \hat{a}_{2^n-1}),$$

$$(b_0^*, \ldots, b_{2^n-1}^*) = \text{ДПФ}_{2^n,4}(\hat{b}_0, \ldots, \hat{b}_{2^n-1}),$$

$$(\hat{c}_0, \ldots, \hat{c}_{2^n-1}) = \text{ДПФ}_{2^n,4^{-1}}(a_0^* b_0^*, \ldots, a_{2^n-1}^* b_{2^n-1}^*).$$

Общая сложность вычислений в этом пункте складывается из $O(2^{2n})$ операций (вычисление $\hat{a}_i, \hat{b}_i$), $O(n2^{2n})$ операций (реализация ДПФ) и $2^n$ умножений в $\mathbb{Z}_{2^{2^n}+1}$.

## Сложность метода умножения

Случай $m = 2n-1$ расмотрен полностью. Случай $m = 2n-2$ рассматривается аналогично, только числа $a$ и $b$ разбиваются на $2^n$ блоков длины $2^{n-1}$ и, в конечном счете, умножение в $\mathbb{Z}_{2^{2^m}+1}$ сводится к $2^{n-1}$ умножениям в $\mathbb{Z}_{2^{2^n}+1}$.

В итоге приходим к следующим рекуррентным соотношениям для сложности $\mu(m)$ умножения в $\mathbb{Z}_{2^{2^m}+1}$:

$$\mu(2n-1) \le 2^n \mu(n) + \gamma_0 (n-1) 2^{2n}, \tag{3}$$

$$\mu(2n-2) \le 2^{n-1} \mu(n) + \gamma_0 (n-1) 2^{2n-1} \tag{4}$$

при подходящей константе $\gamma_0$.

Покажем по индукции, что существует постоянная $\gamma$, такая, что

$$\mu(n) \le \gamma k 2^{n+k} \tag{5}$$

при $n \le 2^k + 1$.

Определим $\gamma'$ так, чтобы при $n \le 3$ было справедливо $\mu(n) \le \gamma' k 2^{n+k}$. Положим $\gamma = \max\{\gamma', \gamma_0\}$.

В силу определения $\gamma$ при $n \le 3$ соотношение (5) выполняется . Проверим индуктивный переход при нечетном аргументе (3); четный случай (4) проверяется аналогично:

$$\mu(2n-1) \le 2^n \gamma k 2^{n+k} + \gamma_0 2^{2n+k} \le \gamma (k+1) 2^{(2n-1)+(k+1)},$$

что и требовалось доказать, поскольку если $n \in [2^{k-1}+2,\ 2^k+1]$, то $2n-2, 2n-1 \in [2^k+2,\ 2^{k+1}+1]$.

Как следствие, получаем, что сложность умножения $n$-разрядных чисел составляет $O(n \log n \log\log n)$.

# Глава 5

# Умножение чисел. Метод Фюрера

Метод Фюрера, предложенный в 2007 г., позволяет достичь наилучшей известной сегодня оценки сложности умножения $n$-разрядных чисел $nc^{\log^* n}\log n$, где $\log^* n$ — сверхлогарифм, очень медленно растущая функция, определяемая из соотношения:

$$\lfloor \underbrace{\log \ldots \log}_{\log^* n} n \rfloor = 1.$$

## Выбор подходящего кольца

Рассмотрим кольцо $\mathbb{C}[x]/(x^{2^p}+1)$. В нем элемент $x$ является примитивным корнем степени $2^{p+1}$ из единицы (что проверяется прямо по определению: многочлены $x^{2^p}-1$ и $x^{2^p}+1$ взаимно просты). Обозначим $\zeta = e^{\mathbf{i}\pi/2^p}$. Заметим, что

$$x^{2^p}+1 = \prod_{k=0}^{2^p-1}(x-\zeta^{2k+1}).$$

Для $q \in \mathbb{N}$ положим $\xi = e^{\mathbf{i}\pi/(2^{p+q})}$. Определим многочлен $\rho_q(x)$ степени меньше $2^p$ следующим образом:

$$\rho_q(\zeta^{2k+1}) = \xi^{2k+1}, \quad k = 0, \ldots, 2^p-1.$$

**Лемма 5.1.** *Многочлен $\rho_q(x)$ является примитивным корнем степени $2^{p+q+1}$ в $\mathbb{C}[x]/(x^{2^p}+1)$.*

*Доказательство.* По построению, $\rho_q^{2^q}(x) = x$ в кольце $\mathbb{C}[x]/(x^{2^p}+1)$. Действительно, при любом $k$

$$\rho_q^{2^q}(x) \bmod (x-\zeta^{2k+1}) = \rho_q^{2^q}(\zeta^{2k+1}) = \xi^{(2k+1)2^q} = \zeta^{2k+1}.$$

Следовательно, $\rho_q^{2^q}(x) \equiv x \mod (x^{2^p}+1)$. Значит, $\rho_q(x)$ является корнем степени $2^{p+q+1}$ из единицы. Примитивность следует из сравнения

$$\rho_q^{2^{p+q}}(x) - 1 \equiv x^{2^p} - 1 \equiv -2 \mod (x^{2^p}+1).$$

**Лемма 5.2.** *Модули коэффициентов многочлена* $\rho_q^m(x) \bmod (x^{2^p}+1)$ *не превосходят* 1.

*Доказательство.* Пусть $(\rho_q^m(x) \bmod (x^{2^p}+1)) = \sum_{k=0}^{2^p-1} r_k x^k$. Подставляя в равенство $\zeta x$ вместо $x$, получим $(\rho_q^m(\zeta x) \bmod (x^{2^p}-1)) = \sum_{k=0}^{2^p-1} r_k \zeta^k x^k$. По определению многочлена $\rho_q(x)$:

$$\text{ДПФ}_{2^p,\zeta^2}\left(r_0\zeta^0, r_1\zeta^1, \ldots, r_{2^p-1}\zeta^{2^p-1}\right) = \left(\xi^m, \xi^{3m}, \ldots, \xi^{(2(2^p-1)+1)m}\right).$$

Тогда, применяя формулу для обратного ДПФ, искомые коэффициенты $r_k$ могут быть вычислены как

$$r_k = \zeta^{-k} 2^{-p} \sum_{l=0}^{2^p-1} \xi^{m(2l+1)} \zeta^{-2kl}.$$

Очевидно, выражение в правой части по модулю не превосходит 1.

# ДПФ в кольце $\mathbb{C}[x]/(x^{2^p}+1)$

Пусть $\omega = \rho_{(s-1)(p+1)}(x)$ — примитивный корень степени $2^{s(p+1)}$ в кольце $\mathbb{C}[x]/(x^{2^p}+1)$. ДПФ порядка $N = 2^{s(p+1)}$, построенное методом Кули-Тьюки с разложением порядка на множители $2^{(s-1)(p+1)}2^{p+1}$, имеет следующую структуру:

– $2^{(s-1)(p+1)}$ параллельно выполняемых преобразований $\text{ДПФ}_{2^{p+1},x}$;

– $N$ параллельных умножений результатов предыдущего шага на степени примитивного корня $\omega$;

– $2^{p+1}$ применяемых к векторам, вычисленным на предыдущем шаге, и параллельно выполняемых ДПФ порядка $2^{(s-1)(p+1)}$.

Расщепляя внешние ДПФ аналогичным образом, получаем окончательно, что при вычислении ДПФ порядка $N$ чередуются $s$ блоков, в которых параллельно выполняются $N/2^{p+1}$ ДПФ порядка $2^{p+1}$ с примитивным корнем $x$, и $s-1$ блоков, в которых выполняются покомпонентные умножения текущего вектора на степени $\omega$.

При этом реализация ДПФ порядка $2^{p+1}$ состоит в выполнении операций сложения-вычитания и умножения на степени $x$ в кольце — последние сводятся к циклическому сдвигу комплексных коэффициентов, если элементы кольца $\mathbb{C}[x]/(x^{2^p}+1)$ записываются многочленами по модулю $x^{2^{p+1}}-1$.

Более точно (опять пользуемся методом Кули—Тьюки), ДПФ порядка $2^{p+1}$ состоит из $p+1$ слоев, в которых параллельно выполняются ДПФ порядка 2, а между слоями выполняются покомпонентные умножения на степени $x$. ДПФ порядка 2 — это просто сложение и вычитание: $\text{ДПФ}_{2,x^{2^p}}(\gamma_0,\gamma_1) = (\gamma_0+\gamma_1,\ \gamma_0-\gamma_1)$.

Пусть действительные и мнимые части комплексных коэффициентов кодируются $E$ двоичными разрядами до и после запятой (плюс один разряд под знак

числа), а операции с ними производятся с погрешностью (абсолютной величиной ошибки) $2^{1-E}$ и с отбрасыванием разрядов старше $E-$го. Тогда сложность каждой из операций сложения-вычитания, умножения на степень $x$ в кольце $\mathbb{C}[x]/(x^{2^p}+1)$ составляет $O(2^pE)$.

Тогда сложность умножения многочленов из кольца $(\mathbb{C}[x]/(x^{2^p}+1))[y]$ по модулю $y^N-1$ можно оценить как

$$3s(N/2^{p+1})(p+1)2^{p+1}O(2^pE)+(3s-2)NL^*\le O(2^pEN\log N)+3sNL^*, \qquad (1)$$

где через $L^*$ обозначена сложность умножения в кольце $\mathbb{C}[x]/(x^{2^p}+1)$ с погрешностью $2^{1-E}$ (под погрешностью понимается максимальная ошибка в каждом коэффициенте многочлена).

Подчеркнем, что все вычисления являются приближенными, в частности, константы $\omega^m$ даны с точностью $E$ знаков после запятой. В силу накопления погрешности (абсолютной величины ошибки) по ходу вычислений точность, с которой вычисляются коэффициенты произведения, вообще говоря, существенно меньше, чем точность исходных коэффициентов. Оценим сопутствующее алгоритму умножения изменение погрешности, считая, что не происходит переполнения в старших разрядах.

## Точность вычислений

Далее с каждым многочленом $a$, возникающим в процессе вычислений, мы будем ассоциировать величину $e_a$, которая является верхней оценкой абсолютной величины погрешности вычисления $a$. Эту оценку будем называть просто погрешностью.

Погрешность $e_{a\pm b}$ результата $a\pm b$ выполнения любой из операций сложения или вычитания комплексных чисел $a$ и $b$, данных с погрешностями $e_a$ и $e_b$ можно положить равной $e_a+e_b$, считая, что используются точные алгоритмы сложения и вычитания ($2E$-разрядных чисел).

Рассмотрим операцию умножения элементов $u,v\in\mathbb{C}[x]/(x^{2^p}+1)$ (каждый из соответствующих многочленов имеет степень не выше $2^{p+1}-1$). Пусть абсолютные величины коэффициентов этих многочленов ограничены числами $U$ и $V$ соответственно, а погрешности — соответственно $e_u$ и $e_v$. Будем полагать, что погрешность алгоритма умножения в кольце $\mathbb{C}[x]/(x^{2^p}+1)$ не превосходит $2^{1-E}$.

Поскольку каждый из коэффициентов произведения $uv$ является суммой $2^{p+1}$ попарных произведений коэффициентов многочленов-сомножителей, то его погрешность можно оценить как

$$e_{uv}=2^{p+1}(e_vU+e_uV+e_ve_u+2^{1-E}).$$

Если $U=V=\Omega(2^{-E})$ и $1\ge e_v=e_u=\Omega(2^{-E})$, то положим $e_{uv}=c_12^{p+1}Ue_u$ при некотором $c_1\ge 1$.

Особо рассмотрим случай, когда многочлен $v$ совпадает с $\omega^m$. Согласно лемме 5.2, $V \le 1$. Т.к. $v$ — постоянный многочлен, то положим $e_v = 2^{-E}$. При выполнении условий $e_u = \Omega(2^{-E})$ и $e_u/U \ge 2^{-E}$ можно положить $e_{uv} = c_2 2^p e_u$ при некотором $c_2 \ge 1$.

В обоих случаях для каждого отдельного коэффициента $a$ произведения $uv$ положим $e_a = e_{uv}$.

Обе указанные оценки для погрешности зависят от величины коэффициентов многочленов-сомножителей. Рассмотрим, как изменяется величина комплексных коэффициентов при вычислении ДПФ.

Пусть комплексные коэффициенты компонент $\gamma_j$ исходного вектора не превосходят $A$ по абсолютной величине. Из предыдущего раздела следует, что под действием ДПФ порядка $2^{p+1}$ максимум абсолютной величины коэффициентов увеличивается не более, чем в $2^{p+1}$ раз. Согласно лемме 5.2, при умножении на $\omega^m$ этот максимум увеличивается не более, чем в $2^p$ раз. Таким образом, для максимума абсолютной величины коэффициентов компонент $\gamma_j^*$ вектора значений ДПФ справедлива оценка $2^{s(p+1)}2^{(s-1)p}A < N^2 A$.

Оценим величину накопленной погрешности. Пусть она не превосходит $e_\gamma$ для коэффициентов компонент $\gamma_j$. Тогда коэффициент увеличения погрешности при выполнении внутреннего ДПФ порядка $2^{p+1}$ можно оценить как $2^{p+1}$, а коэффициент увеличения при умножении на $\omega^m$ — как $c_2 2^p$, если выполняется условие $e_u/U \ge 2^{-E}$. Таким образом, для максимума $e_\gamma^*$ погрешности коэффициентов компонент $\gamma_j^*$ получаем оценку $e_\gamma^* \le N(c_2 2^p)^s < c_2^s N^2$.

Условие $e_u/U \ge 2^{-E}$ в алгоритме выполняется при всех умножениях на степени $\omega$, если изначально выполнено $e_\gamma/A \ge 2^{-E}$, т.к. в случае каждой элементарной операции (сложение, вычитание, умножение на $\omega^m$) отношение $e_u/U$ не убывает.

## Умножение в $\mathbb{C}[x]/(x^{2^p}+1)$

Осталось указать как умножаются многочлены из $\mathbb{C}[x]/(x^{2^p}+1)$ с погрешностью $2^{1-E}$. Поле $\mathbb{C}$ изоморфно фактор-кольцу $\mathbb{R}[\mathbf{i}]/(\mathbf{i}^2+1)$ (элементы которого представляются вещественными многочленами степени 1). Многочлены из $\mathbb{C}[x]/(x^{2^p}+1) \cong \mathbb{R}[\mathbf{i},x]/(\mathbf{i}^2+1,\, x^{2^p}+1)$ могут быть перемножены как обычные многочлены переменных $\mathbf{i}$ и $x$ с последующим приведением по модулям.

Умножим коэффициенты перемножаемых многочленов на $2^E$ (коэффициенты, таким образом, становятся целыми числами). Запишем эти многочлены как

$$a(\mathbf{i},x) = \sum_{j=0}^{1}\sum_{l=0}^{2^{p+1}-1} a_{j,l}\mathbf{i}^j x^l, \qquad b(\mathbf{i},x) = \sum_{j=0}^{1}\sum_{l=0}^{2^{p+1}-1} b_{j,l}\mathbf{i}^j x^l.$$

Коэффициенты произведения исходных многочленов получаются делением коэф-

фициентов произведения $ab$ на $2^{2E}$. Обозначим

$$h(\mathbf{i},x)=\sum_{j=0}^{1}\sum_{l=0}^{2^{p+1}-1}2^{2E}\mathbf{i}^j x^l, \qquad \begin{array}{l} a'(\mathbf{i},x)=a(\mathbf{i},x)+h(\mathbf{i},x),\\ b'(\mathbf{i},x)=b(\mathbf{i},x)+h(\mathbf{i},x).\end{array}$$

Заметим, что коэффициенты многочленов $a', b', h$ положительны. При этом $ab = a'b' - a'h - b'h - h^2$. Тем самым задача сведена к умножению многочленов с неотрицательными целочисленными $(2E+1)-$разрядными коэффициентами.

Такое умножение сводится к умножению $2^{p+1}3(4E+p+4)-$разрядных чисел при подстановке $2^{4E+p+4} \to \mathbf{i}$ и $2^{3(4E+p+4)} \to x$. Окончательно имеем, что сложность умножения в $\mathbb{C}[x]/(x^{2^p}+1)$ по порядку не превосходит сложности умножения $O(2^p(E+p))-$разрядных чисел (в этой же оценке учитывается сложность $O(2^pE)$ дополнительных сложений и вычитаний при вычислении $a'$, $b'$, восстановлении $ab$ и приведении по модулям $\mathbf{i}^2+1$, $x^{2^p}+1$). Алгоритм умножения с погрешностью $2^{1-E}$ получается отбрасыванием разрядов коэффициентов результата младше $E-$го после запятой.

## Оценка сложности

В методе Фюрера используется выбор параметров $E=\Theta(\log N)$ и $p=\log_2\log_2 N+O(1)$. Перемножаемые числа представляются многочленами над $\mathbb{C}[x]/(x^{2^p}+1)$ степени меньше $N/2$ следующим образом: они разбиваются на блоки длины $E/3$, которые интерпретируются как целые числа, каждые $2^{p-1}$ подряд расположенных блоков интерпретируются как коэффициенты многочлена из кольца $\mathbb{C}[x]/(x^{2^p}+1)$.

Получившиеся многочлены (переменной $y$) перемножаются при помощи ДПФ порядка $N$, описанного выше. Число из многочлена восстанавливается за $O(2^pEN)$ операций при подстановке $x=2^{E/3}$, $y=2^{2^{p-1}E/3}$ и приведении подобных.

Параметр $E$ выбирается, исходя из следующих рассуждений. В обозначениях из предыдущего пункта при старте алгоритма $A \le 2^{E/3}$, поэтому положим $e_\gamma = A2^{-E}$. Коэффициенты компонент вектора значений ДПФ имеют абсолютные величины не более $N^2A$ и погрешности не более $c_2^s N^2 e_\gamma$.

Коэффициенты произведения двух элементов из $\mathbb{C}[x]/(x^{2^p}+1)$, абсолютные величины коэффициентов которых не превосходят $U$, ограничены величиной $2^{p+1}U^2$. Погрешность после их перемножения возрастает не более, чем в $c_1 2^{p+1}U$ раз (см. предыдущий пункт). Поэтому для максимума абсолютных величин коэффициентов $W$ и их погрешностей $e_W$ после выполнения покомпонентных умножений векторов значений ДПФ имеем оценки

$$W \le 2^{p+1}U^2 \le 2^{p+1}N^4A^2,$$

$$e_W = c_1c_2^s 2^{p+1}N^2Ue_\gamma \le c_1c_2^s 2^{p+1}N^4Ae_\gamma.$$

Заметим, что $e_W/W \geq c_1 c_2^s N^2 e_\gamma/U \geq c_1 c_2^s e_\gamma/A \geq 2^{-E}$.

Обратное ДПФ (с точки зрения оценок точности и сложности) отличается от прямого делением на $N$, которое выражается в сдвиге позиции запятой в представлении коэффициентов и ведет к уменьшению оценок для абсолютных значений и погрешностей в $N$ раз. Окончательно абсолютные величины и погрешности компонент вектора значений заключительного ДПФ оцениваются как $2^{p+1}N^5A^2$ и $c_1 c_2^{2s} 2^{p+1} N^5 A^2 2^{-E}$ соответственно.

Требуется, чтобы первая оценка была меньше $2^{E-1}$, а последняя оценка была меньше $1/4$ (первое условие следует из второго). Тогда, если в заключение алгоритма выполнить приведение всех коэффициентов при степенях $y$ по модулю $x^{2^p}+1$ (это сводится к параллельным вычитаниям комплексных чисел), то итоговая погрешность будет меньше $1/2$, а абсолютные значения комплексных коэффициентов — меньше $2^E$.

Приходим к условию $2^{E/3} \geq c_1 c_2^{2s} 2^{p+1} N^5 = o(N^6)$, выполнения которого можно добиться, полагая $E = c_3 \log_2 N$ при достаточно большой константе $c_3$.

**Теорема 5.1.** *Умножение $n$-разрядных целых чисел реализуется схемой сложности $L(n) = nc^{\log^* n} \log n$, где $\log^* n$ определяется из соотношения*

$$1 \leq \underbrace{\log_2 \ldots \log_2}_{\log^* n} n < 2.$$

*Доказательство.* Воспользуемся оценкой (1), в которой $L^*$, согласно предыдущему пункту, можно оценить сверху как $L(O(2^p(E+p)))$:

$$L(n) \leq O(2^p EN \log N) + 3sNL(O(2^p(E+p))),$$

а $n \leq 2^{p-2}NE/3$. Учитывая, что $N \asymp n/\log^2 n$, $E \asymp \log n$, $2^p \asymp \log n$, $s \asymp \log n/\log\log n$, получаем соотношение

$$L(n) \leq \alpha \left( n \log_2 n + \frac{n}{\log_2 n \log_2 \log_2 n} L\left(\lfloor \log_2^2 n \rfloor\right) \right).$$

Пусть $L(n) \leq nc_0^{\log^*(\sqrt{n}/4)} \log_2 n$ при $2 \leq n < 256$. Положим $c = \max\{1, c_0, 3\alpha\}$ и проверим по индукции, что $L(n) \leq nc^{\log^*(\sqrt{n}/4)} \log_2 n$. Заметим, что при $n \geq 256$ справедливо $n > \log_2^2 n$ и $\log_2(\sqrt{n}/4) = \frac{1}{2}\log_2 n - 2 \geq \frac{1}{4}\log_2 n$, откуда следует, что $\log^*(\sqrt{\log_2^2 n}/4) \leq \log^*(\sqrt{n}/4) - 1$.

Применяя индуктивное предположение для $n \geq 256$ получаем

$$L(n) \leq \alpha \left( n \log_2 n + 2n \log_2 n c^{\log^*(\sqrt{n}/4)-1} \right) = n \log_2 n \left( 2\alpha c^{\log^*(\sqrt{n}/4)-1} + \alpha \right),$$

откуда следует доказываемое соотношение. Теорема доказана.

# Глава 6

# Деление чисел. Арифметика многочленов

## Быстрое деление чисел

Поскольку операция деления сводится к инвертированию и умножению, достаточно рассмотреть операцию инвертирования. Пусть дано число $A \in [1/2, 1]$, и положим $R = 1/A$. Рассмотрим задачу вычисления числа $R_n$, такого, что $|R - R_n| \leq 2^{-n}$. Иначе говоря, $R_n$ есть приближение к $R$ с точностью $2^{-n}$. Общий случай, когда $A \notin [1/2, 1]$, сводится к рассмотренному при замене $A$ на $2^k A$ и последующем умножении результата на $2^k$ (эти операции реализуются сдвигом позиции запятой, что в схемной модели выполняется бесплатно).

Пусть $a_{..k}$ обозначает число $a$ с отсеченными разрядами после запятой младше $k$-го — это приближение к $a$ с точностью $2^{-k}$.

Пусть $A$ известно с точностью до $n + O(1)$ знаков после запятой. Определим последовательность $r_i$ следующим образом:

$$r_0 = 1, \qquad \tilde{r}_{i+1} = 2r_i - A_{..4+2^i} \cdot r_i^2, \qquad r_i = (\tilde{r}_i)_{..4+2^i}. \tag{1}$$

**Лемма 6.1.**

$$|1 - r_i A_{..4+2^i}| < 2^{-2^{i-1}-1/2}.$$

*Доказательство.* Очевидно, неравенство справедливо при $i = 0$. Докажем индуктивный переход от $i$ к $i + 1$.

а) По индуктивному предположению, справедливо:

$$0 \leq 1 - \tilde{r}_{i+1} A_{..4+2^i} = (1 - r_i A_{..4+2^i})^2 \leq 2^{-2^i-1}.$$

Очевидно $r_{i+1} > 0$. Из левого неравенства дополнительно получаем, что $r_{i+1} \leq A^{-1}_{..4+2^i} \leq 2$.

б) Получим вспомогательную оценку:

$$\begin{aligned}|r_{i+1}A_{..4+2^{i+1}} - \tilde{r}_{i+1}A_{..4+2^i}| \leq \\ \leq |r_{i+1}A_{..4+2^{i+1}} - r_{i+1}A_{..4+2^i}| + |r_{i+1}A_{..4+2^i} - \tilde{r}_{i+1}A_{..4+2^i}| \leq \\ \leq r_{i+1}\,|A_{..4+2^{i+1}} - A_{..4+2^i}| + A_{..4+2^i}\,|r_{i+1} - \tilde{r}_{i+1}| \leq \\ \leq 2 \cdot 2^{-4-2^i} + 1 \cdot 2^{-4-2^{i+1}} < 2^{-3-2^i} + 2^{-4-2^i}.\end{aligned}$$

в) Окончательно, утверждение леммы следует из выкладки:

$$\begin{aligned}|1 - r_{i+1}A_{..4+2^{i+1}}| \leq |r_{i+1}A_{..4+2^{i+1}} - \tilde{r}_{i+1}A_{..4+2^i}| + |1 - \tilde{r}_{i+1}A_{..4+2^i}| < \\ < 2^{-3-2^i} + 2^{-4-2^i} + 2^{-1-2^i} = \frac{11}{16}2^{-2^i} < 2^{-2^i-1/2}.\end{aligned}$$

**Следствие 6.2.** $|1 - r_iA| < 2^{-2^{i-1}}$.

*Доказательство.* Действительно,

$$|1 - r_iA| \leq |1 - r_iA_{..4+2^i}| + r_i\,|A - A_{..4+2^i}| < 2^{-2^{i-1}-1/2} + 2^{-3-2^i} < 2^{-2^{i-1}}.$$

Таким образом, число $r_i$ есть приближение к $R$ с точностью $2^{1-2^{i-1}}$.

## Оценка сложности

Определим специальную функцию $M(n)$, которая, во-первых, не меньше сложности умножения $n$-разрядных чисел, а во-вторых для любых $x, y \in \mathbb{N}$ при $x \leq y$ выполнено

$$M(x)/x \leq M(y)/y. \tag{2}$$

Из этого неравенства вытекает суперлинейность: $M(x + y) \geq M(x) + M(y)$. Из метода Шёнхаге—Штрассена следует, что $M(n) = O(n \log n \log\log n)$.

**Теорема 6.1.** *Сложность инвертирования (с точностью $2^{-n}$) составляет $I(n) = O(M(n))$.*

*Доказательство.* Для вычисления обратного числа с точностью $2^{-n}$ достаточно определить $r_i$ вплоть до $i = \lceil \log_2 n \rceil + 1$. Вычисление $r_{i+1}$, отталкиваясь от $r_i$, можно выполнить за 2 умножения $(5 + 2^i)$-разрядных чисел и одного вычитания со сложностью $2M(5 + 2^i) + O(2^i)$. Следовательно, для определения последнего $r_i$ будет затрачено

$$\sum_{i=0}^{\lceil \log_2 n \rceil + 1} (2M(5 + 2^i) + O(2^i)) \leq 2M\left(2^{\lceil \log_2 n \rceil + 2} + 5\log_2 n + 10\right) + O(n) = O(M(n))$$

операций. Последний переход справедлив в силу $M(cn) = O(M(n))$, где $c$ — константа.

Таким образом, фактически сложность инвертирования и, как следствие, деления, по порядку не выше сложности умножения.

Указанный метод инвертирования был предложен С. Куком около 1966 г. По существу, этот метод является переложением известного метода Ньютона—Рафсона на дискретный случай: в методе Ньютона—Рафсона для поиска решения уравнения $f(x) = 0$ предлагается итерация $x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$. В нашем случае $f(x) = A - 1/x$.

## Арифметика многочленов

От чисел перейдем к изучению основных операций с многочленами. Будем рассматривать многочлены над коммутативным кольцом $R$ с единицей без делителей нуля. Многочлен кодируется набором своих коэффициентов. При реализации арифметических операций с многочленами сложность будем измерять числом арифметических действий: сложений-вычитаний, умножений, делений, умножений в кольце $R$.

Очевидно, сложение или вычитание многочленов степени, меньшей $n$, имеет сложность $n$. Для умножения можно воспользоваться модификацией алгоритма Шёнхаге—Штрассена, который в случае многочленов имеет даже более простое строение, чем для чисел. Умножение выполняется в кольце $R[x]/(x^{2^m}+1)$, и также используется ДПФ. Правда, приходится отдельно рассматривать случай кольца $R$ характеристики 2, т.к. ДПФ порядка степени двойки не определено в кольце $R[x]/(x^{2^m}+1)$. В этом случае предлагается умножать в кольцах $R[x]/(x^{2\cdot 3^m}+x^{3^m}+1)$ и использовать ДПФ порядка степени тройки.

Сложность умножения многочленов степени $n$ методом Шёнхаге—Штрассена, как и в случае чисел, составляет $O(n \log n \log\log n)$, при этом для многочленов неизвестны методы с лучшим порядком сложности.

Роль точности в задаче инвертирования многочленов играет число известных коэффициентов. Пусть задан многочлен $f(x)$, младший коэффициент $f(0)$ которого отличен от нуля. Пусть $r_n(x)f(x) = 1 \bmod x^n$, т.е. $r_n(x) = f^{-1}(x) \bmod x^n$. Тогда очередное приближение $r_{2n}(x) = f^{-1}(x) \bmod x^{2n}$ может быть найдено из аналогичного (1) соотношения

$$r_{2n}(x) = 2r_n(x) - f(x)r_n^2(x). \tag{3}$$

Сложность $I(n)$ нахождения младших $n$ коэффициентов обратного многочлена, таким образом, составляет $O(M(n))$, где $M(n)$ — функция сложности умножения многочленов степени $n-1$, дополненная, как и для чисел, условием $M(x)/x \le M(y)/y$ при $x \le y$.

## Деление с остатком

Рассмотрим задачу нахождения частного $q(x)$ и остатка $r(x)$ от деления многочлена $a(x)$, $\deg a < 2n$, на многочлен $b(x)$, $\deg b = n$. Описываемый далее способ вычислений был предложен Ф. Штрассеном.

Представим делимый многочлен в виде $a(x) = h(x)x^n + g(x)$, $\deg g < n$. Перепишем равенство $a(x) = q(x)b(x) + r(x)$ в виде

$$h(x)x^n = q(x)b(x) + r'(x), \tag{4}$$

где $r'(x) = r(x) - g(x)$.

Введем обозначение $\widetilde{c}(x) = x^{\deg c} c\left(\frac{1}{x}\right)$, которое означает, что коэффициенты многочлена $\widetilde{c}$ являются коэффициентами многочлена $c$, переписанными в обратном порядке.

Подставим в (4) $1/x$ вместо $x$ и умножим на $x^{\deg h+n}$:

$$x^{\deg h+n} h\left(\frac{1}{x}\right) x^{-n} = x^{\deg h+n} q\left(\frac{1}{x}\right) b\left(\frac{1}{x}\right) + x^{\deg h+n} r'\left(\frac{1}{x}\right),$$

откуда, замечая, что $\deg h = \deg q$, получаем

$$\widetilde{h}(x) = \widetilde{q}(x)\widetilde{b}(x) + \widetilde{r'}(x)x^{n+\deg h-\deg r'},$$

и далее, домножением на $x^{n-1-\deg h}$:

$$\left(\widetilde{h}(x)x^{n-1-\deg h}\right) = \left(\widetilde{q}(x)x^{n-1-\deg h}\right)\widetilde{b}(x) + \left(\widetilde{r'}(x)x^{n-1-\deg r'}\right)x^n.$$

Многочлены в скобках обозначим через $\widehat{h}(x), \widehat{q}(x), \widehat{r}(x)$ соответственно. Очевидно, степени всех трех этих многочленов равны $n-1$. Приходим к равенству

$$\widehat{h}(x) = \widehat{q}(x)\widetilde{b}(x) + \widehat{r}(x)x^n, \tag{5}$$

которое приводит к следующему алгоритму.

Найдем многочлен $i(x) = \left(\widetilde{b}(x)\right)^{-1} \bmod x^n$. Из (5) следует, что $\widehat{q}(x) = i(x)\widehat{h}(x) \bmod x^n$, т.е. частное $q(x)$ определяется из произведения $i(x)\widehat{h}(x)$. Наконец, остаток находится по формуле $r(x) = (q(x)b(x) - a(x)) \bmod x^n$. Таким образом, сложность $D(n)$ рассмотренного алгоритма можно оценить как $I(n) + 2M(n) + O(n)$.

# Дополнительные вопросы

1. Обосновать соотношение $M(cn) = O(M(n))$, которое использовалось при доказательстве теоремы 1.
2. Отталкиваясь от метода Ньютона—Рафсона, построить алгоритм приближенного вычисления квадратного корня, сложность которого по порядку равна $M(n)$.
3. Доказать формулу (3).

# Глава 7

# Арифметика чисел. Логарифм и экспонента

## Арифметико-геометрическое среднее

Пусть $a, b \geq 0$. Положим $a_0 = a$, $b_0 = b$ и при любом $k \geq 1$ положим $a_k = \frac{a_{k-1}+b_{k-1}}{2}$ и $b_k = \sqrt{a_{k-1}b_{k-1}}$. *Арифметико-геометрическим средним* чисел $a$ и $b$ называется предел введенных последовательностей:

$$\mathrm{АГС}(a, b) = \lim_{k\to\infty} a_k = \lim_{k\to\infty} b_k.$$

Последовательности $\{a_k\}$ и $\{b_k\}$ сходятся к пределу очень быстро: разность $a_k - b_k$ становится величиной порядка $2^{-n}$ при $k \approx \log_2(n \log_2 |a - b|)$. Более точно скорость сходимости описывается следующей леммой. Для удобства всюду далее будем считать, что $a \geq b$.

**Лемма 7.1.** *Пусть* $1 \leq \frac{a}{b} \leq 1 + 2^{2^m}$. *Тогда*

*а)* $\frac{a_n}{b_n} \leq 1 + 2^{2^{m-n}}$;

*б) при* $n \geq m$ *справедливо* $\frac{a_n}{b_n} \leq 1 + 2^{3-2^{n+1-m}}$.

*Доказательство.* Докажем п. а) индукцией по $n$. При $n = 0$ соотношение гарантируется условием леммы. Индуктивный переход доказывает выкладка:

$$\frac{a_{n+1}}{b_{n+1}} = \frac{1}{2}\left(\sqrt{\frac{a_n}{b_n}} + \sqrt{\frac{b_n}{a_n}}\right) \leq \sqrt{\frac{a_n}{b_n}} \leq \sqrt{1 + 2^{2^{m-n}}} < 1 + 2^{2^{m-n-1}}.$$

Также индукцией докажем п. б). При $n = m$ имеем частный случай п. а). Положим $\frac{a_n}{b_n} = 1 + \varepsilon$, тогда

$$\left(\frac{a_{n+1}}{b_{n+1}}\right)^2 = \left(\frac{\sqrt{1+\varepsilon} + \frac{1}{\sqrt{1+\varepsilon}}}{2}\right)^2 = \frac{1 + \varepsilon + 2 + \frac{1}{1+\varepsilon}}{4} \leq \frac{4 + \varepsilon^2}{4} \leq \left(1 + \frac{\varepsilon^2}{8}\right)^2$$

в силу соотношения $\frac{1}{1+\varepsilon} \leq 1 - \varepsilon + \varepsilon^2$. Следовательно, $\frac{a_{n+1}}{b_{n+1}} \leq 1 + \frac{\varepsilon^2}{8}$, а из $\varepsilon \leq 2^{3-2^{n+1-m}}$ следует $\frac{\varepsilon^2}{8} \leq 2^{3-2^{n+2-m}}$. Лемма доказана.

# Эллиптические интегралы. Теорема Гаусса

При $a, b > 0$ определим несобственный интеграл

$$I(a,b) = \int_0^{+\infty} \frac{dx}{\sqrt{(x^2+a^2)(x^2+b^2)}},$$

относящийся к семейству эллиптических интегралов. Справедлива

**Теорема 7.1** (Гаусс)**.**

$$I(a,b) = \frac{\pi}{2\text{АГС}(a,b)}.$$

Теорема вытекает из следующих двух лемм.

**Лемма 7.2.** *При* $a \geq b$

$$\frac{\pi}{2a} \leq I(a,b) \leq \frac{\pi}{2b}.$$

*Доказательство.* Очевидно $I(a,a) \leq I(a,b) \leq I(b,b)$. Остается заметить, что

$$I(a,a) = \int_0^{+\infty} \frac{dx}{x^2+a^2} = \frac{1}{a}\int_0^{+\infty} \frac{dy}{y^2+1} = \frac{1}{a}\operatorname{arctg} y\Big|_0^{+\infty} = \frac{\pi}{2a}.$$

**Лемма 7.3.**

$$I(a,b) = I\left(\frac{a+b}{2}, \sqrt{ab}\right).$$

*Доказательство.* Пусть $u = \frac{1}{2}\left(x - \frac{ab}{x}\right)$. Заметим, что

$$(x^2+a^2)(x^2+b^2) = x^2(a+b)^2 + (x^2-ab)^2 = 4x^2\left(u^2 + \left(\frac{a+b}{2}\right)^2\right).$$

Кроме того,

$$x\,du = \frac{x}{2}\left(1 + \frac{ab}{x^2}\right)dx = \frac{1}{2}\left(x + \frac{ab}{x}\right)dx = \sqrt{u^2+ab}\,dx.$$

Тогда

$$I(a,b) = \frac{1}{2}\int_0^{+\infty} \frac{dx}{x\sqrt{u^2 + \left(\frac{a+b}{2}\right)^2}} = \frac{1}{2}\int_{-\infty}^{+\infty} \frac{du}{\sqrt{\left(u^2 + \left(\frac{a+b}{2}\right)^2\right)(u^2+ab)}} =$$
$$= \int_0^{+\infty} \frac{du}{\sqrt{\left(u^2 + \left(\frac{a+b}{2}\right)^2\right)(u^2+ab)}} = I\left(\frac{a+b}{2}, \sqrt{ab}\right).$$

Далее нам понадобится еще одна простая лемма:

**Лемма 7.4.**

$$I(1,b) = 2\int_0^{\sqrt{b}} \frac{dx}{\sqrt{(x^2+1)(x^2+b^2)}}.$$

*Доказательство.*

$$\int_{\sqrt{b}}^{+\infty} \frac{dx}{\sqrt{(x^2+1)(x^2+b^2)}} = \int_{\sqrt{b}}^{0} \frac{-b\,du}{u^2\sqrt{((b/u)^2+1)(b/u)^2+b^2)}} = \\ = \int_0^{\sqrt{b}} \frac{du}{\sqrt{(u^2+b^2)(u^2+1)}}.$$

Следующая лемма является ключевой — она устанавливает связь между эллиптическими интегралами и логарифмической функцией.

**Лемма 7.5.** *Если* $b \in (0,\,1]$, *то*

$$0 \le I(1,b) - (2+b^2/2)\ln\left(\sqrt{\frac{1}{b}} + \sqrt{1+\frac{1}{b}}\right) + \frac{1}{2}b\sqrt{1+b} \le \frac{1}{5}b^{3/2}.$$

*Доказательство.* А) Для начала нам понадобятся простые соотношения: при $\alpha \ge 0$

$$1 - \frac{\alpha}{2} \le \frac{1}{\sqrt{1+\alpha}} \le 1 - \frac{\alpha}{2} + \frac{\alpha^2}{2}.$$

Левое неравенство верно в силу того, что при $\alpha \le 2$:

$$(1-\alpha/2)^2(1+\alpha) = (1-\alpha+\alpha^2/4)(1+\alpha) = 1-\alpha^2+\frac{\alpha^2}{4}(1+\alpha) = 1+\frac{\alpha^2}{4}(\alpha-3) \le 1.$$

Правое неравенство следует из

$$(1-\alpha/2+\alpha^2/2)^2 = 1-\alpha+\alpha^2+\frac{1}{4}(\alpha-\alpha^2)^2 \ge 1-\alpha+\alpha^2 \ge \frac{1}{1+\alpha}.$$

Б) Выпишем производные для следующих функций:

$$\ln\left(x+\sqrt{x^2+b^2}\right)' = \frac{1}{\sqrt{x^2+b^2}}, \qquad \left(x\sqrt{x^2+b^2}\right)' = \frac{2x^2+b^2}{\sqrt{x^2+b^2}}.$$

В) Нижняя оценка леммы доказывается как

$$I(1,b) = 2\int_0^{\sqrt{b}} \frac{dx}{\sqrt{(x^2+1)(x^2+b^2)}} \ge \int_0^{\sqrt{b}} \frac{2-x^2}{\sqrt{x^2+b^2}}dx = \\ = \int_0^{\sqrt{b}} \frac{2+b^2/2}{\sqrt{x^2+b^2}}dx - \int_0^{\sqrt{b}} \frac{x^2+b^2/2}{\sqrt{x^2+b^2}}dx = \\ = (2+b^2/2)\ln\left(x+\sqrt{x^2+b^2}\right)\Big|_0^{\sqrt{b}} - \frac{x}{2}\sqrt{x^2+b^2}\Big|_0^{\sqrt{b}} = \\ = (2+b^2/2)\ln\left(\frac{\sqrt{b}+\sqrt{b+b^2}}{b}\right) - \frac{1}{2}b\sqrt{1+b}.$$

Г) Верхняя оценка следует из цепочки неравенств:

$$\int_0^{\sqrt{b}} \frac{1}{\sqrt{(x^2+b^2)}} \left( \frac{2}{\sqrt{x^2+1}} - (2-x^2) \right) dx \leq$$
$$\leq \int_0^{\sqrt{b}} \frac{1}{\sqrt{(x^2+b^2)}} x^4 dx \leq \frac{1}{b} \int_0^{\sqrt{b}} x^4 dx = \frac{1}{5} b^{3/2}.$$

Лемма доказана.

## Алгоритм Брента—Саламина

Пусть дано число $X$, и требуется вычислить $\ln X$ с точностью $2^{-2N}$. Общую схему вычислений описывает диаграмма:

$$X \longrightarrow 2^n X = X_0 \in [2^{N+1}, 2^{N+2}) \longrightarrow$$
$$\longrightarrow b = \left( \frac{2X_0}{X_0^2 - 1} \right)^2,\ X_0 = \sqrt{1/b} + \sqrt{1+1/b} \longrightarrow$$
$$\longrightarrow \text{АГС}(1,b) \longrightarrow I(1,b) = \frac{\pi}{2\text{АГС}(1,b)} \longrightarrow$$
$$\longrightarrow \ln X_0 \approx I(1,b)/2 \longrightarrow \ln X = \ln X_0 - n \ln 2.$$

Предварительно домножением на подходящую степень двойки вход алгоритма приводится к интервалу $X_0 \in [2^{N+1}, 2^{N+2})$. Это гарантирует достаточную малость параметра $b$, такого, что $X_0 = \sqrt{1/b} + \sqrt{1+1/b}$. А именно, $b \leq 2^{-2N}$. Тогда, согласно лемме 7.5, $\ln X_0 = I(1,b)/2 + b/4 + \epsilon$. При этом $\epsilon$ по абсолютной величине можно оценить сверху как $\frac{b^{3/2}}{10} + \frac{b^2}{8} + \frac{b^2}{4} \ln X_0$. Если $N \geq 2$, то $\epsilon \leq b \left( \frac{1}{5 \cdot 2^{N+1}} + \frac{1}{2^{2N+3}} + \frac{N+2}{2^{2N+2}} \right) \leq b/4$, следовательно, $|\ln X_0 - I(1,b)/2| \leq b/2$.

Помимо вычисления АГС, алгоритм содержит несколько аддитивных операций, инвертирований и умножений, в том числе умножений на константы $\pi$ и $\ln 2$, которые тоже нуждаются в вычислении. В действительности эти константы уже вычислены с точностью, по меньшей мере, до нескольких миллионов знаков — и этого достаточно для любых практических вычислений. Иначе, для вычисления указанных (и многих других) констант с точностью $2^{-n}$ известны алгоритмы сложности $O(\log n)M(n)$, их мы оставим за скобками.

Потребуем, чтобы интеграл $I(1,b)$ был вычислен с точностью $b/2$. Тогда точность, с которой вычисляется $\ln X_0$ оценивается как $3b/4$ и, при надлежащей точности заключительного вычитания точность вычисления $\ln X$ может быть оценена как $b$.

Оценим точность $\epsilon$, с которой надо вычислить $\text{АГС}(1,b)$, чтобы обеспечить точность $b/2$ для $I(1,b)$. Для этого с точностью $b$ надо вычислить отношение

$\pi/\mathrm{АГС}(1,b)$ (деление на 2 выполняется точно и при этом вдвое уменьшается погрешность). Так как некоторый запас точности (скажем, $b/2$) нужно оставить для инвертирования и умножения на $\pi$, то приближенное к АГС значение $\mathrm{АГС}^*$ должно удовлетворять соотношению $|\pi/\mathrm{АГС}^* - \pi/\mathrm{АГС}(1,b)| \le b/2$. Это позволяет выписать соотношение для $\epsilon$ вида

$$\left|\frac{\pi}{\mathrm{АГС}(1,b)-\epsilon} - \frac{\pi}{\mathrm{АГС}(1,b)}\right| \le b/2.$$

Несложно получить оценку $\epsilon \le b(\mathrm{АГС}(1,b)/\pi)^2$. Таким образом, можно положить $\epsilon = 2^{-2N-2\log_2 N - c_0}$ при подходящей константе $c_0$.

## Точность вычисления АГС

Следует учесть, что выполняемые в ходе алгоритма действия дают приближенный результат (в случае квадратного корня, это в принципе неизбежно). Оценим рост абсолютной погрешности при вычислении АГС. Пусть в действительности вместо последовательностей $\{a_k\}$ и $\{b_k\}$ вычисляются последовательности $\{a'_k = a_k + e^a_k\}$ и $\{b'_k = b_k + e^b_k\}$. Обозначим $e_k = \max\{|e^a_k|, |e^b_k|\}$. Пусть $E_M$ и $E_Q$ обозначают погрешность выполнения умножения и вычисления квадратного корня соответственно.

**Лемма 7.6.**

$$e_{k+1} \le 2\frac{a_{k+1}}{b_{k+1}}e_k + \frac{e_k^2}{b_{k+1}} + E_Q + \sqrt{E_M}.$$

*Доказательство.* Ясно, что погрешность, вообще говоря, больше прирастает при вычислении $b_k$.

$$|b'_{k+1} - \sqrt{a_k b_k}| = \left|\sqrt{a'_k b'_k + e^M_k} + e^Q_k - \sqrt{a_k b_k}\right| \le$$
$$\le \left|\sqrt{a'_k b'_k + e^M_k} - \sqrt{a'_k b'_k}\right| + \left|\sqrt{a'_k b'_k} - \sqrt{a_k b_k}\right| + E_Q. \quad (7.1)$$

Используя неравенство $|\sqrt{\alpha+\beta} - \sqrt{\alpha}| \le \sqrt{|\beta|}$, первое слагаемое в правой части (7.1) можно оценить как $\sqrt{E_M}$.

При помощи неравенства $|\sqrt{1+\epsilon} - 1| \le |\epsilon|$ второе слагаемое в правой части (7.1) оценим как

$$\left|\sqrt{(a_k + e^a_k)(b_k + e^b_k)} - \sqrt{a_k b_k}\right| \le$$
$$\le \sqrt{b'_k}\left|\sqrt{a_k + e^a_k} - \sqrt{a_k}\right| + \sqrt{a_k}\left|\sqrt{b_k + e^b_k} - \sqrt{b_k}\right| \le$$
$$\le \sqrt{\frac{b'_k}{a_k}}e_k + \sqrt{\frac{a_k}{b_k}}e_k = 2\frac{a_{k+1}}{b_{k+1}}e_k + e_k\sqrt{\frac{b_k}{a_k}}\left(\sqrt{b'_k/b_k} - 1\right) \le 2\frac{a_{k+1}}{b_{k+1}}e_k + \frac{e_k^2}{b_{k+1}}.$$

Лемма доказана.

Подбирая параметры $E_Q \le e_k/3$, $E_M \le e_k^2/9$ и, так как можно полагать (при надлежащем выборе $e_0$) $3e_k \le b_{k+1}$, то имеем $e_{k+1} \le \left(2\frac{a_{k+1}}{b_{k+1}} + 1\right) e_k$ и, как следствие,

$$e_k \le (2a_k/b_k + 1)(2a_{k-1}/b_{k-1} + 1) \cdot \ldots \cdot (2a_1/b_1 + 1)e_0.$$

В нашем случае

$$a_0/b_0 = 1/b = \left(\frac{X_0^2 - 1}{2X_0}\right)^2 \le \left(\frac{2^{2N+4} - 1}{2^{N+3}}\right)^2 < 2^{2N+2} < 1 + 2^{2^{1+\lceil \log_2(N+1)\rceil}}.$$

Пусть $m = \lceil \log_2(N+1)\rceil + 1$. Тогда при помощи леммы 1 при $k \le m-1$ можно оценить $2a_k/b_k + 1$ как $3 + 2^{2^{m-k}+1} < 2^{2^{m-k}+2}$. А при $k \ge m$ указанное выражение оценим как $2a_m/b_m + 1 \le 2(1+2) + 1 < 2^3$. Теперь при $k = L + m - 1$ получаем

$$e_k \le 2^{(2^{m-1}+2)+\ldots+(2^1+2)+3L} e_0 < 2^{2^m+2m+3L} e_0.$$

## Сложность алгоритма

Оценим число итераций для вычисления АГС с необходимой точностью. Поскольку согласно лемме 1: $a_k - b_k \le 2^{3-2^{k+1-m}} b_k$, то при $k \ge 2m + c_1$ погрешность, с которой любое из чисел $a_k$ и $b_k$ приближает АГС$(1, b)$, не превосходит $\epsilon/2 = 2^{-2N-2\log_2 N - c_0 - 1}$. Выбирая $e_0$ в виде $2^{-4N-7\log_2 N - c_2}$, получаем $e_k < \epsilon/2$ — это означает, что погрешность вычисления АГС в алгоритме не превосходит $\epsilon$.

Таким образом, АГС вычисляется за $2\log_2 N + O(1)$ итераций, на каждой из которых выполняется умножение и извлечение корня с $(4 + o(1))N$-разрядными числами. Поэтому общая сложность алгоритма составляет $O(\log N)M^*(N)$.

Заметим, что число $b$ должно быть дано с точностью $2^{-(4+o(1))N}$, следовательно число $X$ должно быть известно с точностью до $(2 + o(1))N$ знаков.

# Глава 8

# Факториал. Метод Шёнхаге

Быстрый вариант алгоритма вычисления $n!$ был предложен Шёнхаге около 1994 г. — он имеет сложность $O(M(\log n!)) = O(M(n \log n))$ и основан на идее «деления пополам».

Обозначим через $\{p_i\}$ последовательность простых натуральных чисел в порядке возрастания. Известно, что число простых чисел, не превосходящих $n$, равно $\pi(n) \sim \frac{n}{\ln n}$ (Адамар, Валле Пуссен).

Метод Шёнхаге состоит в выполнении следующих вычислений в обратном порядке:

$$n! = 2^k x_0, \quad x_0 = x_1^2 y_1, \quad x_1 = x_2^2 y_2, \quad x_2 = x_3^2 y_3, \quad \ldots, \quad x_s = 1,$$

где $y_i$ является произведением всех простых множителей, входящих в $x_{i-1}$ в нечетной степени, и поэтому $x_i$ является квадратом; $k$ — степень вхождения двойки в $n!$. Например,

$$21! = 2^{18} x_0, \quad x_0 = 3^9 \cdot 5^4 \cdot 7^3 \cdot 11 \cdot 13 \cdot 17 \cdot 19,$$

$$x_1 = 3^4 \cdot 5^2 \cdot 7, \quad y_1 = 3 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19,$$

$$x_2 = 3^2 \cdot 5, \quad y_2 = 7.$$

$$x_3 = 3, \quad y_3 = 5,$$

$$x_4 = 1, \quad y_4 = 3.$$

Пусть $e_i(N)$ — степень, в которой число $p_i$ входит в $N$. Известно (и легко проверяется), что

$$e_i(n!) = \lfloor n/p_i \rfloor + \lfloor n/p_i^2 \rfloor + \lfloor n/p_i^3 \rfloor + \ldots = \lfloor n/p_i \rfloor + e_i(\lfloor n/p_i \rfloor !). \tag{1}$$

Принципиально вычисление факториала состоит из трех этапов: (I) поиск простых чисел $p_i \le n$, (II) вычисление показателей $e_i(n!)$ и $e_i(y_j)$, (III) непосредственное вычисление факториала при помощи действий вида $x^2 y$.

# Этап III

Рассмотрим третий этап. Оценим величину чисел $x_i$ и $y_i$.

**Лемма 8.1.** *Пусть* $n! < 2^{2^t}$. *Тогда* $x_i < 2^{2^{t-i}}$ *и если простое число* $p$ *делит* $y_i$, *то* $p \le n/2^{i-2}$.

*Доказательство.* Первое неравенство следует из соотношений $x_i^2 \le x_{i-1}$ и $x_0 \le n!$. Докажем второе.

а) Если $p|y_i$, то $p|x_{i-1}$, $p^2|x_{i-2}$ и т.д. Окончательно, $p^{2^{i-1}}|x_0$.

б) Заметим, что $e_i(n!) < n/p_i + n/p_i^2 + \ldots = n/(p_i - 1)$, откуда $e_i(n!) \le n - 1$.

в) Следовательно, если $p^m|n!$, то $p \le 2n/m$. Действительно, если $p_i > 2n/m$, то

$$e_i(n!) = \lfloor n/p_i \rfloor + e_i(\lfloor n/p_i \rfloor !) \le 2\lfloor n/p_i \rfloor - 1 \le 2\lfloor m/2 \rfloor - 1 < m.$$

г) Поэтому из а) следует, что $p \le n/2^{i-2}$.

Оценим сложность вычисления $y_j$, если даны $e_i(y_j)$. Сложность перемножения $2^s$ чисел длины $b$ не превосходит

$$2^{s-1}M(b) + 2^{s-2}M(2b) + \ldots + M(2^{s-1}b) = O(sM(2^s b)).$$

По лемме число $y_j$ является произведением не более чем $\pi(n/2^{j-2})$ (простых) чисел длины $\log_2 n$, следовательно, вычисляется со сложностью $O(M(n/2^j)\log n)$.

Число $x_{j-1}$, если даны числа $x_j$ и $y_j$ вычисляется со сложностью $O(M(2^{t-j}))$, т.к. число $x_j$ согласно лемме имеет длину не более $2^{t-j}$, а $y_j$ — не более, чем $x_{j-1}$, т.е. $2^{t-j+1}$.

Суммируя сложность вычисления $y_j$ и $x_{j-1}$ по всем $j$ и учитывая, что $2^t = O(n \log n)$, получаем для сложности этапа III оценку

$$\sum_{j=0}^{\log_2 n} O(M(n/2^j)\log n + M(2^t/2^j)) = O(M(n)\log n + M(2^t)) = O(M(n \log n)).$$

# Этап II

Заметим, что достаточно вычислить только набор показателей $e_i(n!)$, т.к. для любого $j$ показатель $e_i(y_j)$ совпадает с $(j-1)$-м разрядом числа $e_i(n!)$ (нумерация с нуля). Действительно, по построению:

$$e_i(n!) = e_i(y_1) + 2e_i(y_2) + \ldots + 2^{s-1}e_i(y_s).$$

При каждом $i$ показатель $e_i(n!)$ вычисляется по формуле (1) за $O(\log n)$ делений и сложений $\log n$-разрядных чисел, т.е. со сложностью $O(M(\log n)\log n)$. Общая сложность, следовательно, не превосходит $\pi(n)O(M(\log n)\log n) = O(nM(\log n))$.

# Этап I

Если вычисления выполняются схемой из функциональных элементов, то все необходимые простые числа $p_i$ следует считать известными заранее. Однако при программной реализации целесообразно рассмотреть случай, когда эти простые числа тоже должны быть вычислены.

Далее мы без доказательства будем использовать известное соотношение

$$\ln\ln n < \sum_{i\le\pi(n)} \frac{1}{p_i} < \ln\ln n + C,$$

справедливое при любом $n \ge 2$.

Пусть нам даны простые числа, не превосходящие $\sqrt{n}$. Тогда остальные простые числа в интервале $[\sqrt{n},\, n]$ могут быть найдены методом «решета Эратосфена». Для этого последовательными сложениями вычисляются последовательности

$$p_i,\ 2p_i,\ \ldots,\ m_i p_i,$$

такие, что $m_i = \lfloor n/p_i \rfloor$. Всего эти последовательности состоят не более чем из $n\sum_{i\le\pi(\sqrt{n})} \frac{1}{p_i} = \Theta(n\log\log n)$ чисел. Таким образом, сложность их вычисления составляет $O(n\log n\log\log n)$.

Заметим, что два упорядоченных набора длины $k$ и $l$ могут быть соединены в один упорядоченный набор не более чем за $k+l-1$ операций сравнения элементов последовательностей. Как следствие, набор из $m$ упорядоченных наборов суммарной длины $N$ можно упорядочить за $O(N\log m)$ операций сравнения, если проводить попарные объединения в бинарном дереве. Операция сравнения чисел длины $b$ имеет сложность $O(b)$.

Разобьем исходные последовательности на группы: в $j$-й группе — последовательности, соответствующие числам $p_i$, $\pi(n^{2^{-1-j}}) < i \le \pi(n^{2^{-j}})$, где $1 \le j < \log_2\log_2 n$.

По построению, $j$-я группа состоит из

$$n \sum_{\pi(n^{2^{-1-j}})<i\le\pi(n^{2^{-j}})} \frac{1}{p_i} = n\Theta(\log(2^{-j}/2^{-1-j})) = O(n)$$

чисел. При этом в $j$-й группе не более $\pi(n^{2^{-j}}) < n^{2^{-j}}$ последовательностей.

Таким образом, сложность упорядочивания чисел в $j$-й группе можно оценить как $O(n\log n^{2^{-j}}) = 2^{-j}O(n\log n)$ операций сравнения $\log_2 n$-разрядных чисел, т.е. $2^{-j}O(n\log^2 n)$. Суммарная сложность упорядочиваний по всем группам следовательно оценивается как $O(n\log^2 n)$.

Полученные $\log_2\log_2 n$ упорядоченных наборов длины не более $n$ упорядочиваются за $\log_2\log_2 n$ операций объединения, при этом длина всех промежуточных упорядоченных наборов не превосходит $n$ — сложность этого шага $O(n\log n\log\log n)$.

Если обозначить через $P(n)$ сложность генерации последовательности простых чисел, не превосходящих $n$, то получено соотношение

$$P(n) \leq P(\sqrt{n}) + O(n \log^2 n),$$

откуда следует $P(n) = O(n \log^2 n)$.

Окончательно для сложности программной реализации вычисления $n!$ получаем оценку $O(M(n \log n) + n \log^2 n)$.