ЦИФРОВЫЕ ТЕХНОЛОГИИ

Ю.В. Нестеренко

Полугодовой спецкурс для студентов 1-6 курсов мехмата МГУ
Весенний семестр 2020-2021 учебного года, дистанционно.
Номер конференции ZOOM: 3229725298. Код доступа 314159.

Оглавление

Глава 1

Введение

Лекция 1.

Основное содержание курса составляют так называемые цифровые технологии, возникшие сравнительно недавно, в связи с образовавшимися потребностями, и переживающие сейчас период становления, они широко обсуждаются, некоторые уже используются, а другие в ближайшее время получат широкое распространение. Деятельность в этой области направлена на создание информации и механизмов её обработки, безопасного хранения и передачи, а в роли инструментов выступают математические алгоритмы, основанные на методах теории чисел, алгебры, дискретной математики. Так что сначала мы освоимся с такими понятиями, как вычислительная сложность алгоритма, познакомимся с конкретными алгебраическими и теоретико-числовыми алгоритмами, имеющими различную сложность, узнаем, что такое хэш-функция и эллиптическая кривая, состоящая из конечного числа точек, научимся строить очень большие простые числа, вычислять дискретные логарифмы и делать многое другое. Всё это составляет основу криптографической науки - важной части цифровых технологий.

Криптография, как область научной деятельности, стала широко известной примерно 40 лет назад, см. [?]. Это было вызвано необходимостью обмена большими массивами конфиденциальной информации (не только государственной и военной, но и банковской, экономической, медицинской, юридической и т.п.), а также возможностью такого обмена, в связи с появлением доступных и эффективных компьютерных средств обработки этой информации. Любая информация может быть закодирована последовательностью чисел. Например, букве "а"можно сопоставить число 1, букве "б" — число 2 и так далее, букве "я"— число 32. Можно сопоставить числа пробелам, точке, другим знакам препинания. После этого процессы зашифрования и

расшифрования информации представятся как некоторые алгоритмы, перерабатывающие одни массивы целых чисел в другие.

Стойкость криптографических алгоритмов зависит от невозможности найти быстрые алгоритмы для решения некоторых задач, другими словами, от того, что некоторые математические задачи сложны в вычислительном отношении.

Сложность алгоритмов теории чисел обычно принято измерять количеством арифметических операций (сложений, вычитаний, умножений и делений с остатком), необходимых для выполнения всех действий, предписанных алгоритмом. Впрочем, это определение не учитывает величины чисел, участвующих в вычислениях. Ясно, что перемножить два стозначных числа значительно сложнее, чем два однозначных, хотя и в том, и в другом случае выполняется лишь одна арифметическая операция. Поэтому иногда учитывают еще и величину чисел, сводя дело к так называемым битовым операциям, т.е. оценивая количество необходимых операций с цифрами 0 и 1 в двоичной записи чисел. Это зависит от рассматриваемой задачи, от целей автора и т.д.

На первый взгляд кажется странным, что операции умножения и деления приравниваются по сложности к операциям сложения и вычитания. Житейский опыт подсказывает, что умножать числа значительно сложнее, чем складывать их. В действительности же, вычисления можно организовать так, что на умножение или деление больших чисел понадобится не намного больше битовых операций, чем на сложение, см. [**?**], гл.8, теорема 8.5. Примерно таким же количеством битовых операций можно обойтись при выполнении деления с остатком двух двоичных чисел. Говоря далее о сложности алгоритмов, мы будем иметь в виду количество арифметических операций, необходимых для их выполнения. Если же будет необходима битовая сложность алгоритма — это будет специально оговариваться.

Рассмотрим далее два примера, показывающие, как используется сложность алгоритмов в некоторых криптографических применениях. Далее будут появляться как общеизвестные, так и известные только тем, кто их определил — секретные величины. В первом случае эти величины будут обозначаться строчными (маленькими) буквами, например "a", во втором же случае прописными (крупными) буквами, скажем "A". Кроме того, условимся, что обмен информацией между участниками алгоритма или протокола[1] идёт только через Интернет, причём все сообщения пересылаются в зашифрованном виде. Участников обмена информацией, в ближайших примерах их будет двое, принято считать одушевлёнными лицами и присваивать им имена.

[1] так называется последовательность действий, в которой участвуют двое или более лиц

Обычно легальных действующих лиц именуют Алиса и Боб [2], злоумышленник получает имя Ева и так далее. Заметим, что, несмотря на происхождение, действующие лица могут быть и компьютерами, которые иногда для удобства описания наделяются чувствами и желаниями.

Пример 1. Алгоритм шифрования RSA.

В 1977г. был предложен действующий до настоящего времени и наиболее известный алгоритм шифрования RSA. Это название образовано первыми буквами фамилий авторов работы [**?**], в которой он был описан.

Рассмотрим следующий сюжет. Допустим, что Боб хотел бы получать письма от Алисы, но так, чтобы никто не смог узнать их содержание. Для этого он выбирает два простых числа p, q и вычисляет их произведение $N = pq$. Затем Боб случайным образом выбирает два натуральных числа E, d, меньших $\varphi(N) = (p-1)(q-1)$, [3] и удовлетворяющих сравнению

$$E \cdot d \equiv 1 \pmod{\varphi(N)}. \tag{1.1}$$

Для этого достаточно выбрать число E, $1 < E < \varphi(N)$, взаимно простое с $\varphi(N)$ и обозначить буквой d наименьшее положительное решение сравнения (1.1). Открытым ключом Боба будет пара чисел (N, E). Боб может, например, переслать эту пару чисел Алисе по открытой почте. Секретный ключ Боба — это число d. Числа p и q также должно держать в секрете, но рассматривать всю тройку (p, q, d) в качестве секретного ключа необязательно, потому что числа p, q используются только на стадии генерации чисел N, E, d. Поэтому числа p, q можно уничтожить. Отметим, что любые целые E и d,удовлетворяющие (1.1), взаимно просты с $\varphi(N)$. Итак, для зашифрования сообщений используется открытый ключ, а для расшифрования — другой ключ, который известен только Бобу.

Допустим, что секретное письмо x Алисы удовлетворяет условиям

$$1 < x < N, \qquad (x, N) = 1.$$

Заметим, что последнее условие означает взаимную простоту чисел x и N.

Для того, чтобы зашифровать сообщение x, Алисе достаточно вычислить

$$y \equiv x^E \pmod{N}, \qquad 0 < y < N. \tag{1.2}$$

Число y условиями (1.2) определяется однозначно. Оно представляет собой письмо Алисы в зашифрованном виде и может быть переслано Бобу через Интернет.

[2] Их роль подобна латинским буквам a и b в описаниях алгебраических преобразований.

[3] Здесь функция $\varphi(m)$ есть функция Эйлера. Её значение равно количеству целых чисел на отрезке от 1 до m, взаимно простых с m.

Согласно теореме Эйлера справедливо сравнение

$$y^d \equiv x^{Ed} \equiv x \pmod{N}. \tag{1.3}$$

Последнее сравнение выполняется в силу (1.1).

Боб, получив зашифрованное письмо Алисы y, находит оригинал письма

$$x \equiv y^d \pmod{N}, \qquad 0 < x < N. \tag{1.4}$$

Сравнение (1.3) доказывает корректность алгоритма RSA. Если письмо Алисы по длине больше N, его можно разбить на блоки меньшей длины, чем N, и шифровать каждый блок отдельно. Если же письмо не взаимно просто с N, то к нему можно дописать время отправления или небольшой набор из нулей и единиц, чтобы обеспечить это условие.

Боб может вывесить открытый ключ (N, E) на своём сайте, или поместить его в какой-нибудь справочник вроде старой телефонной книги. Это, как легко видеть, позволит большому числу людей писать ему секретные письма. Если каждый из некоторой группы людей выработает указанным способом свои открытый и секретный ключи и сделает свой открытый ключ доступным всем участникам группы, то члены этой группы смогут обмениваться конфиденциальной информацией через Интернет. Причём понять содержание письма сможет только тот, кому оно адресовано.

Возведение в степень по фиксированному модулю, решение линейных сравнений с одной переменной могут выполняться сравнительно быстро. Об этом мы поговорим немного позже. Сложность расшифрования письма с помощью алгоритма (1.4) зависит от сложности вычисления секретного ключа d. Последний может быть найден очень легко, если известно значение функции Эйлера $\varphi(N) = (p-1)(q-1)$. Но для этого нужно найти числа p, q, т.е. разложить число N на простые сомножители. Эта фундаментальная задача

Разложить заданное число N на нетривиальные множители

очень сложна. Для чисел N размером в 1024 битов при использовании самых мощных в настоящее время компьютеров требуются годы для её решения. А для сохранения очень ценных секретов рекомендуется использовать числа N, записываемые примерно 2048 битами.

Еще Ферма предложил нетривиальный алгоритм разложения чисел на множители. Различные его улучшения использовались Эйлером, Гауссом, Лежандром, Чебышевым и другими классиками теории чисел. Современные алгоритмы используют вычисления в полях алгебраических чисел, эллиптические кривые, разнообразные математические конструкции.

Пример 2. Алгоритм шифрования Эль-Гамаля.

Ещё одна очень сложная в вычислительном отношении задача носит название дискретного логарифмирования.

Для заданных простого числа P и целых чисел A, B, не делящихся на P, требуется решить сравнение

$$A^x \equiv B \pmod{P}. \tag{1.5}$$

На большой сложности этой задачи основан алгоритм шифрования Эль-Гамаля.

Как известно, для любого простого числа P ненулевые элементы поля вычетов $\mathbb{F}_P = \mathbb{Z}/P\mathbb{Z}$ образуют циклическую группу $\mathbb{F}_P^*$. Обозначим буквой G образующую этой группы. Порядок G по модулю P равен $P-1$. После выбора открытых параметров P, G Боб, который хотел бы получать сообщения от Алисы, определяет свои открытый и секретный ключи. Секретным ключом может быть любое натуральное число $d, 0 < d < P-1$, Боб выбирает его случайно на интервале от 1 до $P-1$, а открытый ключ определяется условиями

$$E \equiv G^d \pmod{P}, \qquad 1 < E < P-1.$$

Если Алиса хочет отправить Бобу некоторое письмо, она должна разбить его на блоки по величине меньшие, чем P и шифровать каждый блок отдельно. Пусть $m, 1 < m < P$ — целое число, представляющее какой-либо из блоков. Для шифрования Алиса поступает следующим образом:

- выбирает случайно какое-нибудь целое число $k, 1 < k < P$,
- вычисляет $C_1 \equiv G^k \pmod{P}$,
- находит $C_2 \equiv m \cdot E^k \pmod{P}$,

Пара чисел $C = (C_1, C_2)$ есть представление блока m из письма Алисы в зашифрованном виде. Шифртекст C пересылается Бобу в открытом виде по сети Интернет.

При каждом шифровании применяется свой ключ k — кратковременный ключ. Поэтому, шифруя одно сообщение дважды, можно получить разные шифртексты.

Чтобы расшифровать полученное сообщение $C = (C_1, C_2)$, Боб вычисляет $C_2 \cdot C_1^{P-1-d} \pmod{P}$. Это число и будет сообщением Алисы. Действительно, пользуясь определениями чисел C_1 и C_2 находим

$$C_2 \cdot C_1^{P-1-d} \equiv m \cdot G^{kd+k(P-1-d)} \equiv m \left(G^{P-1}\right)^k \equiv m \pmod{P}.$$

Последнее сравнение основано на малой теореме Ферма, согласно которой $G^{P-1} \equiv 1 \pmod{P}$.

Если злоумышленник перехватит шифртекст C, то для восстановления оригинала посланного сообщения ему нужно будет решать задачу дискретного логарифмирования $C_1 \equiv G^k \pmod{P}$, что, как указывалось выше, требует очень много времени. Он не сможет определить число k и затем из сравнения для C_2 найти искомый блок m.

Алгоритм Эль-Гамаля можно организовать так, чтобы вычисления проводить не во всей мультипликативной группе поля $\mathbb{F}_P$, а в её подгруппе простого порядка. Это усложнит задачу дискретного логарифмирования. Пусть P, Q большие простые числа, такие, что $P-1$ делится на Q. Выберем первообразный корень g по модулю P, и определим число G условиями

$$G \equiv g^{\frac{P-1}{Q}}.$$

Тогда $G \neq 1$ и циклическая подгруппа $(G) \subset \mathbb{F}_P^*$ имеет простой порядок Q. Далее все вычисления открытого ключа E, секретного ключа d, шифртекста (C_1, C_2), а также расшифрование выполняются по тем же формулам, как и ранее.

Заметим, что построение секретного и открытого ключей в системе Эль-Гамаля требует намного меньше времени, чем в RSA. Среди минусов нужно отметить, что длина шифртекста примерно вдвое превосходит длину шифруемого блока.

Наконец, заметим, что выкладываемые тексты иногда содержат больше информации, чем рассказывалось на лекции. Как правило, это более глубокая информация для заинтересованных слушателей или информация, которую лучше изучать самостоятельно с бумагой и ручкой.

Глава 2

Алгоритмы и сложность

В этой главе мы обсудим ряд полезных на практике быстрых алгоритмов и докажем некоторые оценки их сложности.

2.1 Алгоритм Евклида и теорема Ламе.

Рассмотрим следующую задачу. Пусть a, b — натуральные числа. Требуется найти (a, b) — наибольший общий делитель a, b. Задача эта решается достаточно быстро с помощью хорошо известного алгоритма Евклида без разложения чисел на множители. В основе его лежит равенство $(a, b) = (b, r)$, где r — остаток от деления числа a на b.

Алгоритм 1. Даны: *Натуральные числа a и b; $b < a$.*
Найти: *Наибольший общий делитель (a, b).*

1. Вычислить r — остаток от деления a на b, т.е. найти целое r, удовлетворяющее условиям $a = bq + r,\ 0 \leq r < b$.
2. Если $r = 0$, то $(a, b) = b$, СТОП.
3. Если $r \neq 0$, то заменить пару $\{a, b\}$ парой $\{b, r\}$ и перейти в пункт 1 алгоритма.

Пусть r — остаток от деления числа a на b, т.е. $a = bq + r$, $0 \leq r < b$. По свойствам делимости каждый общий делитель чисел b и r делит число $bq + r = a$ и, значит, принадлежит множеству общих делителей чисел b и a. Точно так же, каждый общий делитель чисел a и b делит число $a - bq = r$, так что принадлежит множеству общих делителей чисел b и r. Отсюда следует совпадение наибольших общих делителей пар чисел a, b и b, r, т.е. равенство

$$(a, b) = (b, r). \tag{2.1}$$

Это равенство позволяет при нахождении наибольшего общего делителя заменять пару чисел a, b другой парой b, r. Заметим, что $r < b$, т.е. одно из двух

чисел, участвующих в алгоритме уменьшилось. Повторяя несколько раз деление с остатком и заменяя каждый раз пару целых чисел новой мы будем каждый раз уменьшать одно из двух чисел (большее), участвующих в работе алгоритма. Ясно, что в какой-то момент одно из чисел станет равным 0 и наибольший общий делитель будет равен второму из чисел.

Рассмотрим алгоритм немного подробнее. Положим $r_0 = a$, $r_1 = b$ и обозначим через $r_2, \ldots, r_n$ — последующие делители в алгоритме Евклида. Тогда получаются следующие равенства

$$\begin{aligned}
a = r_0 &= bq_1 + r_2, && 0 \le r_2 < b, \\
b = r_1 &= r_2q_2 + r_3, && 0 \le r_3 < r_2, \\
r_2 &= r_3q_3 + r_4, && 0 \le r_4 < r_3, \\
&\cdots\cdots\cdots\cdots && \cdots\cdots\cdots\cdots \\
r_{n-2} &= r_{n-1}q_{n-1} + r_n, && 0 \le r_n < r_{n-1}, \\
r_{n-1} &= r_nq_n.
\end{aligned} \tag{2.2}$$

Алгоритм останавливается, когда деление произойдет без остатка. В приведенном выше тексте последний остаток $r_{n+1} = 0$. В соответствии с равенством (2.1) находим

$$(a, b) = (b, r_2) = (r_2, r_3) = (r_3, r_4) = \cdots = (r_{n-1}, r_n) = (r_n, 0) = r_n.$$

Таким образом, наибольший общий делитель равен последнему делителю (он же последний ненулевой остаток) в алгоритме Евклида.

Следующее утверждение было доказано в 1760г. Л. Эйлером и носит его имя.

Теорема 1 (Теорема Эйлера)**.** *Для каждого целого числа a, взаимно простого с модулем m, выполняется сравнение*

$$a^{\varphi(m)} \equiv 1 \pmod{m}.$$

Рассмотрим частный случай теоремы Эйлера, в котором $m = p$ есть простое число. Тогда $\varphi(p) = p - 1$ и получается утверждение, называемое малой теоремой Ферма.

Теорема 2 (Малая теорема Ферма)**.** *Если целое число a не делится на простое число p, то*

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

Из этой теоремы следует, что при любом целом a число $a^p - a = a(a^{p-1} - 1)$ делится на p.

Рассмотрим пример вычисления наибольшего общего делителя двух чисел, а именно, вычислим $(89, 55)$. Пользуясь алгоритмом Евклида, находим

$$\begin{aligned} 89 &= 55 \cdot 1 + 34, & 55 &= 34 \cdot 1 + 21, & 34 &= 21 \cdot 1 + 13, \\ 21 &= 13 \cdot 1 + 8, & 13 &= 8 \cdot 1 + 5, & 8 &= 5 \cdot 1 + 3, \\ 5 &= 3 \cdot 1 + 2, & 3 &= 2 \cdot 1 + 1, & 2 &= 1 \cdot 2 + 0. \end{aligned}$$

Итак, $(89, 55) = 1$, и для вычисления наибольшего общего делителя понадобилось 9 делений с остатком. Числа 55 и 89 есть числа Фибоначчи с номерами 10 и 11. Напомним, что последовательность чисел Фибоначчи определяется рекуррентным уравнением $F_{i+1} = F_i + F_{i-1}$ и начальными данными $F_0 = 0$, $F_1 = 1$. Приведённое выше вычисление показывает, и это легко доказать по индукции, что для вычисления наибольшего общего делителя чисел (F_n, F_{n+1}) нужно не менее $n - 1$ делений с остатком.

Доказываемая далее теорема Ламе, даёт верхнюю оценку для количества делений с остатком в алгоритме Евклида вычисления наибольшего общего делителя двух чисел, т.е. даёт верхнюю оценку сложности алгоритма. Заметим, что эта оценка может подходить очень близко к возможной нижней оценке, доставляемой примером с числами Фибоначчи. Таким образом оценка теоремы Ламе очень точна.

Теорема 3 (Ламе, 1845). *Пусть $a > b$ — натуральные числа. При вычислении (a, b) с помощью алгоритма Евклида будет выполнено не более $5m$ операций деления с остатком, где m есть количество цифр в десятичной записи числа b.*

Доказательство. Положим $r_0 = a$ и обозначим $r_1, r_2, \ldots, r_n$ — последовательность делителей в алгоритме Евклида. Тогда $r_1 = b$,

$$r_{i-1} = q_i r_i + r_{i+1}, \quad 0 \le r_{i+1} < r_i, \ q_i \in \mathbb{N}, \quad i = 1, 2, \ldots, n-1,$$

$r_n = (a, b)$.

Докажем справедливость следующих неравенств

$$r_i \ge \lambda^{n-i}, \qquad i = 1, 2, \ldots, n, \tag{2.3}$$

где $\lambda = \frac{1+\sqrt{5}}{2} = 1,61...$ есть корень квадратного уравнения $\lambda^2 - \lambda - 1 = 0$. Для этого воспользуемся индукцией по индексу i, двигаясь в обратном направлении от n к 1. При $i = n$ имеем $r_n \ge 1 = \lambda^0$. При $i = n - 1$ находим

$r_{n-1} \geq r_n + 1 \geq 2 > \lambda$, так что неравенство (2.3) опять справедливо. Предположим теперь, что $k < n$ и неравенство (2.3) выполняется при всех $i \geq k$. Имеем следующую цепочку равенств и неравенств

$$r_{k-1} = q_k r_k + r_{k+1} \geq r_k + r_{k+1} \geq \lambda^{n-k} + \lambda^{n-k-1} = \lambda^{n-k-1}(\lambda + 1) = \lambda^{n-k+1}.$$

Таким образом, неравенство (2.3) выполняется и при $i = k - 1$. Это доказывает справедливость (2.3) при всех $i = 1, 2, \ldots, n$.

Поскольку десятичная запись b содержит m знаков, то $10^m > b$, и

$$10^m > b = r_1 \geq \lambda^{n-1}.$$

Из этих неравенств следует, что

$$m > (n-1)\log_{10}\lambda > (n-1)/5.$$

Последнее неравенство выполняется, поскольку $\lambda > 10^{1/5} = 1,58....$ Таким образом, $n < 5m + 1$, что завершает доказательство теоремы. □

Из теоремы 3 следует, что алгоритм Евклида работает достаточно быстро.

Алгоритмы, в которых необходимое количество арифметических операций оценивается фиксированной степенью длины записи его входных данных, называются *полиномиальными*. Таким является алгоритм Евклида, а также алгоритмы решения других задач (например, линейных уравнений и сравнений), связанные с алгоритмом Евклида.

2.2 Символы Лежандра и Якоби

В этом параграфе мы рассмотрим вопрос о том, как узнать, разрешимо или нет по простому модулю $p > 2$ квадратичное сравнение. Хорошо известно, что сравнение

$$ax^2 + bx + c \equiv 0 \pmod{p}, \qquad a, b, c \in \mathbb{Z},\ p \nmid a,$$

сводится выделением полного квадрата к сравнению

$$x^2 \equiv d \pmod{p}, \qquad d \in \mathbb{Z}. \tag{2.4}$$

Ответ находится тривиально в случае $p \mid d$, решениями являются числа $x \equiv 0 \pmod{p}$.

Если $p \nmid d$, вопрос о разрешимости (2.4) также может быть решен достаточно быстро. Мы изложим ниже соответствующую теорию, отсылая за доказательствами к [**?**].

Определение 1. Символ Лежандра $\left(\frac{d}{p}\right)$ *есть функция от двух аргументов:* d *есть целое число, а* p *— простое нечетное. Значение символа Лежандра равно 1, если сравнение* (2.4) *разрешимо, оно равно* -1*, если это сравнение не имеет решений и оно равно 0, если* $p \mid d$*.*

Определенная так функция обладает рядом свойств, позволяющих вычислять её значения и тем самым получать ответ на вопрос о разрешимости сравнения (2.4). При этом иногда требуется проверять простоту чисел, а иногда раскладывать числа d на простые множители. В настоящее время это очень трудоемкая задача. Ниже мы опишем более совершенный способ вычисления символа Лежандра. Соответствующий алгоритм имеет полиномиальную сложность и не использует ни проверки чисел на простоту, ни разложения на простые множители. В основе его лежит возможность продолжить функцию $\left(\frac{d}{p}\right)$ на множество всех пар взаимно простых целых чисел D, P, где P — произвольное нечетное число. Продолженная таким образом функция носит название *символ Якоби.*

Определение 2. *Пусть* $P = p_1 \cdots p_r$*, где* p_j *— нечетные простые числа,* $D \in \mathbb{Z}$*,* $(D, P) = 1$*. Символ Якоби* $\left(\frac{D}{P}\right)$ *определяется равенством*

$$\left(\frac{D}{P}\right) = \left(\frac{D}{p_1}\right) \cdots \left(\frac{D}{p_r}\right),$$

где $\left(\frac{D}{p_j}\right)$ *— значения символа Лежандра.*

Если P — простое число, то символы Лежандра и Якоби со знаменателем P совпадают.

Вообще говоря, значение символа Якоби не связано с разрешимостью сравнений.

Пример. Легко проверить, что сравнение $x^2 \equiv 2 \pmod{15}$ не имеет решений. Тем не менее символ Якоби в этом случае равен

$$\left(\frac{2}{15}\right) = \left(\frac{2}{3}\right) \cdot \left(\frac{2}{5}\right) = (-1) \cdot (-1) = 1.$$

Следующие свойства подобны соответствующим свойствам символа Лежандра.

Свойства символа Якоби

2. *Если* $a \equiv b \pmod{P}$*, то* $\left(\frac{a}{P}\right) = \left(\frac{b}{P}\right)$*.*

3. $\left(\frac{1}{P}\right) = 1; \quad \left(\frac{-1}{P}\right) = (-1)^{\frac{P-1}{2}}; \quad \left(\frac{2}{P}\right) = (-1)^{\frac{P^2-1}{8}}.$

4. $\left(\frac{ab}{P}\right) = \left(\frac{a}{P}\right) \cdot \left(\frac{b}{P}\right).$

5. *Если P и Q — положительные нечетные взаимно простые числа, то*

$$\left(\frac{P}{Q}\right) \cdot \left(\frac{Q}{P}\right) = (-1)^{\frac{P-1}{2}\frac{Q-1}{2}}.$$

Покажем теперь, как с помощью этих свойств можно вычислить значение символа Якоби. При этом не используется разложение на простые множители, а лишь выделяется максимальная степень числа 2, входящая в разложение числителя. Рассмотрим сначала следующий пример.

Пример. Выяснить, разрешимо ли сравнение

$$x^2 \equiv 753 \pmod{811}$$

Отметим, что число 811 простое, так что для решения задачи достаточно вычислить соответствующий символ Якоби:

$$\left(\frac{753}{811}\right) = \left(\frac{811}{753}\right) = \left(\frac{58}{753}\right) = \left(\frac{29}{753}\right) = \left(\frac{753}{29}\right) = \left(\frac{-1}{29}\right) = 1.$$

При этом вычислении использовались соотношения $753 \equiv 1 \pmod 8$, $811 \equiv 58 \pmod{753}$, $753 \equiv -1 \pmod{29}$, $29 \equiv 1 \pmod 4$. Итак, данное в примере сравнение разрешимо.

Способ вычисления символа Якоби, использовавшийся в рассмотренном примере может быть оформлен в виде следующего алгоритма.

Алгоритм 2. Данные: *Целые взаимно простые числа Q и $P > 2$, P нечетно.*
Найти: *Значение символа Якоби* $\left(\frac{Q}{P}\right)$.

1. Положить $s = 0,\ u = Q,\ v = P.$
2. Найти r — наименьший положительный остаток от деления числа u на v, т.е. целое число, удовлетворяющее условиям

$$u = vq + r, \qquad 0 < r < v;$$

вычислить целое $k \geq 0$ и нечетное t, для которых $r = 2^k t$; положить

$$s \equiv s + k \cdot \frac{v^2 - 1}{8} + \frac{(t-1)(v-1)}{4} \pmod 2.$$

3. Если $t = 1$, положить

$$\left(\frac{Q}{P}\right) = (-1)^s.$$

СТОП.
4. Если $t \geq 3$, положить $u = v,\ v = t$, перейти в пункт 2.

Докажем, что приведенный алгоритм действительно вычисляет символ Якоби и получим оценку его сложности.

Теорема 4. *Приведенный алгоритм вычисляет символ Якоби $\left(\frac{Q}{P}\right)$ за $O(m)$ арифметических операций, где m есть количество цифр в десятичной записи меньшего из чисел P, Q.*

Доказательство. В процессе работы алгоритма число v убывает. Это значит, что алгоритм не может работать бесконечно долго и, следовательно, завершит свою работу.

Обозначим буквой n количество проходов алгоритма через пункт 2. Пусть также $s_j,\ u_j,\ v_j$ — значения параметров $s,\ u,\ v$ перед j-м входом в пункт 2. Тогда $s_1 = 0, u_1 = Q, v_1 = P$. Докажем индукцией по j справедливость равенств

$$\left(\frac{Q}{P}\right) = (-1)^{s_j} \cdot \left(\frac{u_j}{v_j}\right), \qquad j = 1, \ldots, n. \tag{2.5}$$

При $j = 1$ это равенство, очевидно, выполняется. Пусть $j < n$ и (2.5) справедливо. В процессе выполнения пункта 2 алгоритма в j-й раз будут найдены числа $q,\ k,\ t$, для которых $u_j = qv_j + 2^k t$. Тогда

$$\left(\frac{u_j}{v_j}\right) = \left(\frac{2^k t}{v_j}\right) = \left(\frac{2}{v_j}\right)^k \cdot \left(\frac{t}{v_j}\right) = (-1)^{k\frac{v_j^2-1}{8} + \frac{t-1}{2}\frac{v_j-1}{2}} \left(\frac{v_j}{t}\right).$$

Это равенство вместе с (2.5) завершает шаг индукции. Итак, равенство (2.5) справедливо для всех j.

При $j = n$, т.е. при последнем выходе из пункта 2) будем иметь $t = 1$ и, следовательно,

$$\left(\frac{Q}{P}\right) = (-1)^{s_n} \cdot \left(\frac{u_n}{v_n}\right) = (-1)^{s_n} \left(\frac{2}{v_n}\right)^k = (-1)^{s_n + k\frac{v_n^2-1}{8}}.$$

Это доказывает, что алгоритм действительно вычисляет значение символа Якоби $\left(\frac{Q}{P}\right)$.

Учитывая равенства $u_j = v_{j-1}, t = v_{j+1}$, находим, что

$$v_{j-1} \geq v_j + v_{j+1}, \qquad j \geq 2.$$

Получившиеся неравенства, как и в доказательстве теоремы 4 приводят к оценке $v_j \geq \lambda^{n-j}$, доказывающей при $j = 1$, что $n = O(\ln P) = O(m)$. Оценка количества арифметических операций в алгоритме имеет, очевидно, тот же порядок. □

Конец первой лекции.

Лекция 2.

2.3 Возведение в степень.

Пусть A — некоторое множество, замкнутое относительно умножения. Это может быть множество целых чисел или многочленов, классов вычетов по какому-нибудь модулю и другие множества. Пусть также d — натуральное число. В этом параграфе мы обсудим быстрый алгоритм вычисления степени a^d. Тривиальный алгоритм, состоящий в последовательном умножении на a результата предшествующего вычисления, требует $d-1$ умножений. Для некоторых чисел d вычисления могут быть проделаны намного быстрее. Так, например, для $d=32$ можно обойтись всего лишь 5 умножениями

$$a^2, a^4=(a^2)^2, a^8=(a^4)^2, a^{16}=(a^8)^2, a^{32}=(a^{16})^2.$$

А для $d=23$ достаточно 6 умножений

$$a^2, a^3=a^2\cdot a, a^5=a^3\cdot a^2, a^{10}=(a^5)^2, a^{13}=a^{10}\cdot a^3, a^{23}=a^{13}\cdot a^{10}.$$

Следующий общий алгоритм вычисляет степень существенно быстрее тривиального.

Алгоритм 3. Данные: *Элемент $a\in A$ и натуральное число d.*
Найти: *Элемент a^d.*

1. Представить d в двоичной системе счисления, т.е. найти такие числа $d_j\in\{0,\ 1\}$, что $d=d_0 2^r+\cdots+d_{r-1}2+d_r,\ d_0=1$.
2. Положить $a_0=a$ и затем для $i=1,\ldots,r$ вычислить

$$a_i=a_{i-1}^2\cdot a^{d_i}. \tag{2.6}$$

3. Положить $a^d=a_r$.

Теорема 5. *Алгоритм действительно вычисляет степень a^d. Он использует для этого не более $2[\log_2 d]$ умножений в кольце A.*

Доказательство. При всех $i=0,1,\ldots,r$ справедливы равенства

$$a_i=a^{d_0 2^i+\cdots+d_i}. \tag{2.7}$$

Докажем это индукцией по i. При $i=0$ утверждение выполняется, т.к. $d_0=1$. Подставляя (2.7) в равенство $a_{i+1}=a_i^2\cdot a^{d_{i+1}}$, находим равенство (2.7) для a_{i+1}.

Это доказывает (2.7) для всех i. При $i = r$ получаем $a_r = a^d$, что доказывает корректность алгоритма.

Для оценки сложности обозначим символом $c(d)$ количество умножений, необходимых алгоритму для вычисления a^d. Докажем индукцией по d неравенство

$$c(d) \le 2[\log_2 d]. \tag{2.8}$$

Обозначим для этого $m = d_0 2^{r-1} + \cdots + d_{r-1}$. Тогда $d = 2m + d_r \ge 2m$.

При $d = 1,\ 2$ неравенство (**??**), очевидно, выполняется. Пусть теперь $d \ge 3$. В силу алгоритма справедливы соотношения

$$c(d) = c(m) + 1 + d_r \le c(m) + 2 \le 2[\log_2 m] + 2 = 2[\log_2 2m] \le 2[\log_2 d],$$

доказывающие нужное неравенство. □

Рассмотрим некоторые примеры использования описанного алгоритма.

Пример 1. Пусть m — натуральное число и $A = \mathbb{Z}/m\mathbb{Z}$ — кольцо вычетов по модулю m. Алгоритм реализует вычисление степеней в кольце вычетов по модулю m. Классическая теорема Эйлера утверждает, что $a^{\varphi(m)} \equiv 1 \pmod m$, где $\varphi(m)$ — функция Эйлера. Поэтому, заменив показатель степени d его остатком от деления на $\varphi(m)$ (на что потребуется одна арифметическая операция), задачу вычисления $a^d \pmod m$ всегда можно свести к случаю $d \le \varphi(m)$. Это показывает, что сложность алгоритма возведения в степень в кольце вычетов $\mathbb{Z}/m\mathbb{Z}$ есть $O(\ln m)$.

В частности, если $m = p$ — простое нечетное число, то в силу одного из свойств символа Лежандра имеем

$$\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \pmod p,$$

так что алгоритм 3 дает другой способ вычисления символа Лежандра за $O(\log p)$ арифметических операций.

Пример 2. Пусть R — кольцо и последовательность элементов $u_0,\ u_1,\ u_2, \ldots \in R$ задана рекуррентно

$$u_n = a_1 u_{n-1} + \cdots + a_h u_{n-h}, \qquad n \ge h, \quad a_j \in R.$$

Например, можно взять $R = \mathbb{Z}/m\mathbb{Z}$. Как вычислить u_d для заданного индекса d? Введем для этого вектора

$$\overline{u}_n = (u_n, u_{n-1}, \ldots, u_{n-h+1}), \qquad n \ge h.$$

Тогда справедливо равенство $\overline{u}_n = \overline{u}_{n-1} \cdot M$, где

$$M = \begin{pmatrix} a_1 & 1 & 0 & \cdots & 0 \\ a_2 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{h-1} & 0 & 0 & \cdots & 1 \\ a_h & 0 & 0 & \cdots & 0 \end{pmatrix},$$

— квадратная матрица из h строк и столбцов с элементами в кольце R. Очевидно равенство $\overline{u}_n = \overline{u}_h \cdot M^{n-h}$, из которого следует, что вычисление u_d сводится к вычислению M^{d-h} в кольце A квадратных матриц размера h с элементами в R. Сложность вычисления u_d таким способом оценивается величиной $O(h \ln d)$, где постоянная в $O(\cdot)$ абсолютна.

2.4 Алгоритм Карацубы умножения целых чисел.

Обычный метод умножения n - значных чисел "столбиком" требует n^2 перемножений цифр. В настоящее время имеются более эффективные методы, использующие существенно меньше операций умножения. Первый такой метод был предложен в 1962г. А.А. Карацубой.

Пусть a и b — произвольные натуральные числа. Они имеют по $[\log_2 a]+1$ и $[\log_2 b]+1$ цифр в двоичной записи. Пусть n — минимальное целое число, такое что

$$\max\Big([\log_2 a]+1,\ \leq 2^n \quad \text{и} \quad [\log_2 b]+1\Big) \leq 2^n.$$

Тогда a, b не более чем 2^n - разрядные целые числа.

Описываемый ниже алгоритм сводит задачу умножения чисел a и b к нескольким умножениям чисел, длина которых в двоичной записи на превосходит $k = 2^{n-1}$, и рекурсивно применяется к этим новым числам. Вычислительный алгоритм называется рекурсивным, если в процессе работы он обращается к самому себе, но с меньшими параметрами. Возникающее при этом множество обращений в конечном итоге сводит задачу к некоторым начальным значениям параметров, при которых выполнение алгоритма становится тривиальным.

Представим числа a и b в виде

$$a = a_1 + 2^k a_2, \qquad b = b_1 + 2^k b_2,$$

где a_1, a_2, b_1, b_2 — числа, длина которых в двоичной записи не превосходит k. Справедливы равенства

$$ab = a_1 b_1 + 2^k (a_1 b_2 + a_2 b_1) + 2^{2k} a_2 b_2$$

и

$$ab = a_1b_1 + 2^k((a_1 + a_2)(b_1 + b_2) - (a_1b_1 + a_2b_2)) + 2^{2k}a_2b_2. \qquad (2.9)$$

Первое из них даёт так называемый школьный алгоритм умножения (цифра уможается на цифру, а потом результаты складываются с коэффициентами - степенями двойки). Этот алгоритм потребует n^2 умножений цифр в двоичной записи чисел a и b.

Обсудим подробнее алгоритм, происходящий от второго тождества. Числа $a_1 + a_2$ и $b_1 + b_2$ могут иметь в двоичной записи длину $k + 1$, но их можно представить в виде

$$a_1 + a_2 = \alpha + 2a_3, \qquad b_1 + b_2 = \beta + 2b_3,$$

где числа a_3 и b_3 имеют длину не больше, чем k, а числа α и β суть нули или единицы. Стало быть, (2.9) можно переписать в виде

$$ab = a_1b_1 + 2^k(\alpha\beta + 2\alpha b_3 + 2\beta a_3 + 4a_3b_3 - (a_1b_1 + a_2b_2)) + 2^{2k}a_2b_2. \qquad (2.10)$$

Таким образом, исходная задача при помощи нескольких операций сложения, вычитания и домножения на степень двойки (которое является просто–напросто приписыванием соответствующего числа нулей) сводится к вычислению трех произведений a_1b_1, a_2b_2 и a_3b_3, каждое из которых является произведением чисел длины не более, чем $k = 2^{n-1}$. К этим трем произведениям рекурсивно применяем описанные рассуждения.

Обозначим через L_n количество битовых операций, необходимое для вычисления данным методом произведения двух произвольных чисел, длина которых в двоичной записи не превосходит 2^n. Тогда найдется такая константа C, что $L_0 \le C$ и

$$L_n \le 3L_{n-1} + 2^nC, \qquad n \ge 1.$$

Отсюда по индукции находим, что

$$L_n \le C(3^{n+1} - 2^{n+1}).$$

Действительно, в предположении, что $L_{n-1} \le C(3^n - 2^n)$, получаем:

$$L_n \le 3C(3^n - 2^n) + C2^n = C(3^{n+1} - 2^{n+1}).$$

Учитывая теперь, что $2^{n-1} < [\log_2 a] + 1$, при $a \ge b \ge 2$ заключаем

$$L_n \le C \cdot 3^{n+1} = 3C \cdot (2^n)^{\log_2 3} < 3C \cdot (2[\log_2 a] + 2)^{\log_2 3} = \\ = 9C \cdot ([\log_2 a] + 1)^{\log_2 3}.$$

Таким образом, если a и b — натуральные числа и $a \geq b$, то перемножить их можно не более, чем за $9C \cdot m^{\log_2 3}$ битовых операций, где $m = [\log_2 a] + 1$ - количество цифр в двоичной записи числа a.

Конец второй лекции.

Лекция 3.

2.5 Вероятностные алгоритмы отсеивания составных чисел.

Рассмотренные ранее алгоритмы относятся к разряду детерминированных. Так называют алгоритмы, в которых результат каждого шага однозначно определяется предшествующими вычислениями. Однако, существуют и другие типы алгоритмов, так называемые вероятностные алгоритмы, весьма удобные на практике.

Пусть N — натуральное число, вообще говоря, большое. Оно может быть либо составным, либо простым. Соответственно, можно рассмотреть две важные в связи с поисками или построением больших простых чисел задачи.

1. N — составное число, и требуется доказать это.

2. N — простое число, и требуется доказать это.

В настоящем параграфе будет рассматриваться первая из задач. Мы покажем, что существуют достаточно быстрые вероятностные алгоритмы, решающие ее и не использующие при этом разложение N на множители.

Выберем целое число $a, 1 < a < N$. Справедливы следующие утверждения:

1) Если наибольший общий делитель $(a, N) > 1$, то N — составное число.

Условие $(a, N) > 1$ легко проверить, вычислив (a, N) с помощью алгоритма Евклида.

2) Если $(a, N) = 1$ и $a^{N-1} \not\equiv 1 \pmod N$, то N — составное число.

Это утверждение следует из малой теоремы Ферма, и его также легко проверить с помощью алгоритма возведения в степень.

К сожалению, утверждений 1)-2) не достаточно для доказательства того, что испытываемое число N — составное. Существуют составные числа, для которых выполняется сравнение $a^{N-1} \equiv 1 \pmod N$ при любом целом $a, (a, N) = 1$.

Определение 3. *Составное число N называется числом Кармайкла, если при любом $a, (a, N) = 1$, выполняется сравнение*

$$a^{N-1} \equiv 1 \pmod N. \tag{2.11}$$

Пример 1. $N = 561 = 3 \cdot 11 \cdot 17$.

Для каждого целого $a, (a, 561) = 1$, имеем $(a, 3) = (a, 11) = (a, 17) = 1$ и

по малой теореме Ферма выполняются сравнения

$$a^2 \equiv 1 \pmod 3, \qquad a^{10} \equiv 1 \pmod{11}, \qquad a^{16} \equiv 1 \pmod{17}.$$

Так как 560 делится на $2, 10$ и 16, то число $a^{560} - 1$ делится на каждую из разностей $a^2 - 1, a^{10} - 1, a^{16} - 1$. Значит, $a^{560} - 1$ делится на 3, 11, 17 и потому делится на произведение этих чисел, т.е. делится на 561. Число 561 есть число Кармайкла.

В 1994г. было доказано, что множество чисел Кармайкла бесконечно. Существование чисел Кармайкла показывает необходимость уточнения свойств 1)-2) для доказательства непростоты чисел. Следующее утверждение позволяет достаточно эффективно решать рассматриваемую задачу.

Тест 1 (Миллер - Рабин). *Пусть $N > 2$ — нечетное натуральное число. Определим целые числа s, t равенством $N - 1 = 2^s t$, где t нечетно. Выберем целое число $a, a > 1$.*
1) Если $(a, N) > 1$, то N составное число.
2) Если $(a, N) = 1$ и выполнены условия

$$a^t \not\equiv 1 \pmod N, \quad a^{2^k t} \not\equiv -1 \pmod N, \quad k = 0, 1, \ldots, s-1, \qquad (2.12)$$

то N составное число.

Справедливо разложение на множители

$$a^{N-1} - 1 = (a^t - 1)(a^t + 1)(a^{2t} + 1) \cdots (a^{2^{s-1}t} + 1).$$

Если N — простое число, то по малой теореме Ферма обе части последнего равенства должны делиться на N. Поскольку N — простое, то хотя бы один из сомножителей в правой части этого равенства должен делиться на N. Значит, хотя бы одно из условий (2.12) будет нарушено. Это доказывает справедливость второго утверждения теста Миллера - Рабина.

Рассмотрим множество $M(N)$ состоящее из чисел $a \in \mathbb{Z}$, $1 \le a < N$, $(a, N) = 1$, для которых нарушается хотя бы одно из условий (2.12).

Если N — простое число, то $\#M(N) = N - 1$. Это следует из справедливости второго утверждения теста Миллера - Рабина. Для составных N множество $M(N)$ будет достаточно малым. Точнее, выполняется доказанное в 1984г. Рабином утверждение.

Теорема 6. *Пусть N — нечетное составное число, $N \neq 9$. Тогда $\#M(N) \le \frac{1}{4}\varphi(N)$.*

Мы не приводим здесь доказательство этой теоремы. Оно элементарно, но длинно, см. книгу [**?**].

Пример 2. *Для* $N = 9$ *имеем* $M(9) = \{1,\ 8\}$ *и* $\#M(9) = 2 = \frac{1}{3}\varphi(9)$.

Приведённый ниже вероятностный алгоритм, удостоверяет, что заданное число — составное, если оно таковым является. Алгоритм основан на тесте Миллера-Рабина.

Алгоритм 4. Данные: *Нечетное число* $N > 9$, *повидимому, составное.*
Доказать: *Число* N *составное.*

1) Вычислить натуральные числа s, t *такие, что* $N - 1 = 2^s t$ *и* t *нечетно.*
2) Выбрать случайным образом число $a, 1 < a < N$ *и проверить выполнимость условий 1) и 2) теста Миллера - Рабина.*
3) Если хотя бы одно из условий теста выполнено, то число N *составное; СТОП.*
4) Если оба условия теста нарушаются, то перейти в пункт 2.

Для оценки сложности этого вероятностного алгоритма, что нужно для сравнения эффективности различных алгоритмов, можно использовать среднее время работы алгоритма, оно же иногда называется математическое ожидание времени работы.

Докажем сейчас, что среднее количество арифметических операций, необходимое для выполнения работы этого алгоритма, т.е. его сложность, не превосходит $c_0 \cdot \ln N$.[1]

Согласно определению множество $M(N)$, количество его элементов оценивается в теореме 6, состоит из всех чисел $a \in \mathbb{Z},\ 1 \le a < N,\ (a, N) = 1$, для которых нарушается хотя бы одно из условий (2.12). Таким образом, любое целое число $a,\ 1 \le a < N$, не принадлежащее этому множеству, подтвердит с помощью теста Миллера - Рабина, что число N составное. Значит при случайном выборе a мы с вероятностью

$$\rho = \frac{N - 1 - \#M(N)}{N - 1} \ge 1 - \frac{1}{4}\frac{\varphi(N)}{N - 1} \ge \frac{3}{4}$$

попадаем на хорошее a (доказывающее непростоту N). Здесь использовалась оценка сверху $\#M(N) \le \frac{1}{4}\varphi(N)$ количества элементов в множестве $M(N)$, справедливая для всех нечетных составных чисел $N > 9$, см. теорему 6.

[1] Здесь и далее буквой c с индексами обозначаются различные абсолютные постоянные, т.е. постоянные, не зависящие ни от каких параметром алгоритма, например $3, \pi$ или e^2.

Пусть A_k — событие, состоящее в том, что при k испытаниях $k-1$ раз попадались плохие a и в k-й раз попалось хорошее. Тогда $p(A_k) = (1-\rho)^{k-1}\rho$. Так как тест Миллера – Рабина использует только вычисление наибольшего общего делителя двух чисел и возведение в степень по модулю N, то при фиксированном a для проверки условий теста 2 требуется не более $c_1 \ln N$ арифметических операций. Для проверки же принадлежности $a \in A_k$ потребуется не более $c_1 k \ln N$ арифметических операций. Поэтому среднее количество арифметических операций в алгоритме не превосходит

$$c_1 \ln N \cdot \sum_{k=1}^{\infty} k(1-\rho)^{k-1}\rho = \rho \cdot c_1 \ln N \sum_{k=1}^{\infty} k(1-\rho)^{k-1}$$

Дифференцируя тождество $\sum_{k=1}^{\infty} x^k = -1 + \frac{1}{1-x}$, находим

$$\sum_{k=1}^{\infty} k x^{k-1} = \frac{1}{(1-x)^2}.$$

Из этого тождества при $x = 1-\rho$ полуаем, что среднее количество арифметических операций в алгоритме не превосходит величины $\rho^{-1} c_1 \ln N \leq \frac{4c_1}{3} \ln N = c_2 \ln N$.

На практике алгоритм 4 является очень важной составляющей общей стратегии доказательства простоты чисел. Если заданное число N, о котором не известно простое оно или составное, прошло достаточно много шагов алгоритма 4, то оно наверное будет простым. Ведь вероятность составному числу N выдержать d испытаний не превосходит 4^{-d}. Например, при $d = 100$ она ничтожно мала 4^{-100}. Таким образом, остается только вопрос, как доказать, что это число простое?

Конец третьей лекции.

Лекция 4.

2.6 Доказательство простоты больших чисел

В начале этого параграфа мы обсудим так называемые $(N-1)$ - методы доказательства простоты чисел. Они позволяют, используя некоторую информацию о свойствах простых делителей числа $N-1$, заключить, что N - простое число. Существуют также $(N+1)$ и (N^2+1) - методы, есть и другие обобщения излагаемых ниже идей, но мы их здесь касаться не будем.

Теорема 7 (Лемер, Поклингтон)**.** *Пусть N нечетно, $N-1=F\cdot R$, причем для каждого простого делителя q числа F с некоторым целым b выполнены условия*

$$b^{N-1}\equiv 1 \pmod N, \qquad \left(b^{(N-1)/q}-1,\ N\right)=1. \tag{2.13}$$

Тогда любой простой делитель p числа N удовлетворяет сравнению

$$p\equiv 1 \pmod F$$

В следующем далее доказательстве теоремы 7 используется символ $\nu_q(a)$, обозначающий для любого целого a и простого q кратность, с которой q входит в разложение a на простые сомножители.

Доказательство. Пусть p — простой делитель N, q — простой делитель F и b — число, удовлетворяющее (2.13). Первое из сравнений (2.13) означает, что b не делится на p. Обозначим $a=b^R$. Тогда a не делится на p и по малой теореме Ферма выполняется сравнение $a^{p-1}\equiv 1 \pmod p$. Обозначим буквой d наименьшее натуральное число с условием $a^d\equiv 1 \pmod p$. Разделим $p-1$ на d с остатком: $p-1=ud+v$, где $0\le v<d$. Тогда

$$1\equiv a^{p-1}=(a^d)^u\cdot a^v\equiv a^v \pmod p.$$

Если $v>0$, приходим к противоречию с определением d. Поэтому $v=0$,

$$d|(p-1) \quad \text{и} \quad \nu_q(d)\le\nu_q(p-1). \tag{2.14}$$

Из (2.13) следует, что

$$a^F\equiv 1 \pmod p, \qquad a^{F/q}\not\equiv 1 \pmod p.$$

Отсюда точно так же, как и (2.14) выводятся свойства $d|F$ и $d\nmid F/q$, что возможно лишь в случае, когда

$$\nu_q(F)=\nu_q(d). \tag{2.15}$$

Из (2.14) и (2.15) находим

$$\nu_q(F) = \nu_q(d) \le \nu_q(p-1).$$

Так как последнее неравенство справедливо для любого простого делителя q числа F, то $F|p-1$ или $p \equiv 1 \pmod F$. □

Приведём два следствия теоремы, связанные с доказательством простоты чисел. Если известна достаточно большая часть разложения $N-1$ на простые сомножители, то иногда можно сделать заключение о простоте N.

Следствие 1. *Если в условиях теоремы 7 выполнено неравенство*

$$R \le F + 1$$

то N — простое.

Доказательство. Согласно теореме 7 каждое простое $p|N$ удовлетворяет сравнению $p \equiv 1 \pmod F$ и, значит, удовлетворяет неравенству $p \ge 1 + F$. Предположим, что N — составное. Тогда

$$(F+1)^2 \le N = FR + 1 \le F^2 + F + 1.$$

Получившееся противоречие завершает доказательство. □

Следствие 2. *Пусть F – простое нечётное число, R – чётное число, удовлетворяющие условиям $R \le 4F + 2$ и $N = FR + 1$. Если*

$$b^{N-1} \equiv 1 \pmod N, \qquad \left(b^R - 1,\, N\right) = 1. \tag{2.16}$$

с некоторым целым b из промежутка $1 < b < N$, то N – простое число.

Доказательство. Согласно теореме 7 при $q = F$ каждое простое p, делящее N удовлетворяет сравнению $p \equiv 1 \pmod F$. Кроме того, p нечетно, так что $p \equiv 1 \pmod{2F}$ и $p \ge 2F + 1$. Для составного N имеем неравенства

$$(2F+1)^2 \le N = FR + 1 \le 4F^2 + 2F + 1,$$

что неверно. Значит, N — простое число. □

Предположим теперь, что построенное число N действительно является простым, и мы хотим доказать это с помощью Следствия 2. Если при выбранном b условия (2.13) нарушаются, нужно выбрать другое значение b и повторять эти операции до тех пор, пока такое число не будет обнаружено.

Зададимся вопросом, сколь долго придётся перебирать числа b, пока не будет найдено такое, для которого выполнены условия (2.13). Заметим, что для простого числа N первое условие (2.13), согласно малой теореме Ферма, будет выполняться всегда. Те же числа b, для которых нарушается второе условие (2.13), удовлетворяют сравнению $b^R \equiv 1 \pmod N$. Как известно, уравнение $x^R = 1$ в поле вычетов $\mathbb{Z}/N\mathbb{Z}$ имеет не более R решений. Одно из них — $x = 1$. Поэтому на промежутке $1 < b < N$ имеется не более $R - 1$ чисел, для которых не выполняются условия (2.13). Это означает, что, выбирая случайным образом число b на промежутке $1 < b < N$, при простом N можно с вероятностью большей, чем $1 - F^{-1}$, найти число b, для которого будут выполнены условия следствия 2, и тем доказать, что N действительно является простым числом.

Построенное таким способом простое число N будет удовлетворять неравенству $N > F^2$, т.е. будет записываться вдвое большим количеством цифр, чем исходное простое число F. Заменив теперь число F на найденное простое число N и повторив с этим новым F все указанные выше действия, можно построить еще большее простое число. Начав с какого-нибудь простого числа, скажем, записанного 10 десятичными цифрами (простоту его можно проверить, например, делением на маленькие табличные простые числа), и повторив указанную процедуру достаточное число раз, можно построить простые числа нужной величины.

Алгоритм 5. Данные: *Большое положительное целое число M. Алгоритм строит простое число N, превосходящее границу M.*

1) Определить, например, с помощью таблиц простое число F, превосходящее 10^5. Определить счётчик S_1 и положить $S_1 = 0$.
2) Выбрать случайным образом чётное число R из промежутка $F \le R \le 4F+2$ и положить $N = FR+1$. Если НОД$[N, 15015] > 1$, повторить пункт 2.
3) Вычислить натуральные числа s, t такие, что $N - 1 = 2^s t$ и t нечетно.
4) Выбрать случайным образом число $a, 1 < a < N$. Если
• $(a, N) > 1$ *или*
• $(a, N) = 1$ *и*

$$a^t \not\equiv 1 \pmod N, \quad a^{2^k t} \not\equiv -1 \pmod N, \quad k = 0, 1, \ldots, s-1, \qquad (2.17)$$

то N составное число. Перейти в пункт 2). В противном случае увеличить счётчик S_1 на 1 и, если $S_1 < 20$, перейти в п. 4. Если же $S_1 = 20$, положить $S_1 = 0$ и $S_2 = 0$, перейти в п. 5).

5) Выбрать случайным образом b из промежутка $1 < b < N$. Если N не удовлетворяет хотя бы одному из условий

$$b^{N-1} \equiv 1 \pmod{N}, \qquad \left(b^R - 1,\, N\right) = 1, \tag{2.18}$$

то
5.1) при $S_2 < 10$ увеличить S_2 на 1 и повторить пункт 5).
5.2) В случае же $S_2 = 10$, перейти в пункт 4.
6) Если N удовлетворяет обоим условиям (2.18) *и*
6.1) $N < M$, положить $F = N$ и перейти в пункт 2.
6.2) $N \geq M$, алгоритм останавливается, нужное простое число построено.

2.7 Псевдослучайные числа. Линейный конгруэнтный метод.

В двух предшествующих параграфах мы обсуждали вероятностные алгоритмы, которые, впрочем, дают совершенно верные результаты. Лишь оценки их времени работы носят усреднённый характер. Например, если натуральные числа N и a взаимно просты и $a^{N-1} \not\equiv 1 \pmod{N}$, можно с уверенностью утверждать, что N составное число. Нам не известны нетривиальные натуральные сомножители, на которые раскладывается это число, но мы знаем, что они существуют. А сложность алгоритма, отсеивающего составные числа, по-существу, оценивается средним временем работы, необходимой для нахождения подходящего числа a.

В вероятностных алгоритмах используется случайный выбор каких-либо чисел на нужном промежутке. Конечно, у каждого есть своё представление, какая последовательность может считаться случайной, но вместе с тем, понятно, что с помощью детерминированного алгоритма, используемого компьютером, случайную последовательность не построишь. Можно лишь построить последовательность, имитирующую те или иные свойства случайной последовательности. Такие последовательности называются псевдослучайными, а успех вероятностного алгоритма зависит от того, насколько используемая в нём псевдослучайная последовательность близка по своим свойствам к случайной.

В течение длительного времени основным инструментом для построения псевдослучайных последовательностей был так называемый линейный конгруэнтный метод, предложенный в 1948г. Д. Лемером. Нужная последова-

тельность возникала по правилу

$$x_{n+1} \equiv a \cdot x_n + b \pmod{m}, \qquad 0 \le x_{n+1} < m, \quad n \ge 0, \tag{2.19}$$

где a, b, m и x_0 - некоторые параметры, определяемые специальным образом. Выбор параметров - непростая задача, как показывают следующие далее примеры.

Пример. Последовательность $x_n, n \ge 0$, определённая условиями

$$x_{n+1} \equiv 1986 x_n + 35491 \pmod{2^{16}}, \qquad n \ge 0, \qquad x_0 = 1181,$$

имеет вид

$$1181, 21661, 62661, 13469, 46237, 46237, 46237, \ldots .$$

Все её члены, начиная с пятого места, одинаковы, что, конечно, не свойственно случайным последовательностям.

Любая последовательность, построенная по правилам

$$x_{n+1} = f(x_n) \pmod{m}, \qquad n \ge 0, \qquad 0 \le x_{n+1} < m, \tag{2.20}$$

где x_0 - произвольно выбранное начальное значение и $f(x)$ - функция, определённая для всех целых неотрицательных чисел и принимающая целые значения, будет периодической, а длина её периода не превосходит m. Действительно, существует ровно m различных классов вычетов по модулю m. Поэтому множество классов вычетов $f(x_n) \pmod{m}$, $0 \le n \le m$, состоящее из $m+1$ классов вычетов, должно содержать по крайней мере два одинаковых элемента. Если $f(x_k) \equiv f(x_\ell) \pmod{m}$, $0 \le \ell < k \le m$, то

$$x_{k+1} \equiv f(x_k) \equiv f(x_\ell) \equiv x_{\ell+1} \pmod{m}.$$

Учитывая, что $0 \le x_{\ell+1} < m$, $0 \le x_{k+1} < m$ и разность этих чисел делится на m, заключаем, что $x_{k+1} = x_{\ell+1}$ и $f(x_{k+1}) = f(x_{\ell+1})$. Далее точно так же доказывается, что $x_{k+2} = x_{\ell+2}$ и вообще $x_{n+k-\ell} = x_n$ для любого $n \ge \ell$. Итак, начиная с номера ℓ последовательность начнёт повторяться с периодом $k - \ell \le m$.

Случайная последовательность не должна быть периодической. Поэтому естественно желание выбирать в качестве псевдослучайных, последовательности, имеющие, по возможности, больший период. Справедливо следующее утверждение. *Длина периода последовательности* (2.19) *будет максимальной (равной* m*) в том и только том случае, когда:*

1. b *и* m *взаимно просты,*

2. $a-1$ кратно всем простым делителям числа m,
3. если m кратно 4, то и $(a-1)$ делится на 4.
Мы не будем здесь доказывать это утверждение, см. .
Примеры. Последовательности

$$x_{n+1} \equiv 19381 \cdot x_n + b \pmod{2^{16}}$$

при любом нечётном b и любом x_0 имеют максимальный возможный период $2^{16} = 65536$. Так будет, например, при $b = x_0 = 1$. Если же мы возьмём последовательность, определённую условиями

$$x_{n+1} \equiv 19381 \cdot x_n + 29772 \pmod{2^{16})], \qquad x_0 = 1,$$

то она будет иметь вид

$$1, 49153, 32769, 16385, 1, 49153, 32769, 16385, 1, \ldots.$$

Период её равен 4, конечно же она не является псевдослучайной.

Свойство иметь максимальный период ещё не обеспечивает хорошие с точки зрения случайности свойства последовательности. Например, рекуррентное уравнение

$$x_{n+1} \equiv x_n + 1 \pmod{2^{16}}, \qquad x_0 = 1,$$

определяет последовательность максимального периода

$$1, 2, 3, 4, \ldots, 65534, 65535, 0, 1, 2, 3, 4, \ldots,$$

очевидно не случайную.

Рассмотрим ещё один пример. Определим последовательность уравнением

$$x_{n+1} = 16565 \cdot x_n + 4051, \qquad n \geq 0, \qquad x_0 = 1. \tag{2.21}$$

Начало этой последовательности имеет вид

$$1, 20616, 65531, 52298, 37, 27132, 65279, 6686, 1801, 18736, 52931,$$
$$65458, 22701, 548, 37703, 61702, 63761, 26840, 12427, 8730$$

Эта последовательность имеет максимальный период и на первый взгляд выглядит как случайная. Тем не менее использовать её, как псевдослучайную не имеет смысла, ведь для всех $n \geq 0$ выполняется соотношение

$$x_{n+2} + 7x_n - 2 \equiv 0 \pmod{[)2^(16)]}. \tag{2.22}$$

Предпринимались попытки усложнить функцию $f(x)$, стоящую в (2.20), с тем, чтобы исключить соотношения, подобные (2.22), и другие нежелательные эффекты. С этой целью рассматривались полиномы второй и третьей степени, дробно линейные функции. Параллельно разрабатывались тесты для проверки свойств, характерных для случайных последовательностей. Перспективнее оказались последовательности в рекуррентном определении котоых присутствуют функции от нескольких переменных, например

$$x_{n+q} = f(x_{n+q-1}, x_{n+q-2,\ldots,x_{n+1},x_n}),$$

имеющие существенно большую длину периода.

В криптографических приложениях последовательности, конструируемые с помощью линейного конгруэнтного метода не используются. Для того, чтобы построить случайное число записываемое в двоичной системе счисления 1024 цифрами 0 и 1, а в настоящее время это нижняя граница величины допустимых простых чисел в задачах дискретного логарифмирования, нужно приложить друг к другу 64 случайных числа, записываемые 16 битами. При этом должна быть обеспечена их независимость. В приведённом выше примере (2.21) уже соседние три члена последовательности связаны соотношением с маленькими коэффициентами. В следующем подразделе мы раскажем о том, как строить псевдослучайные последовательности чисел размером, скажем в 256 битов. Чтобы построить псевдослучайное число размером в 1024 бита, достаточно поместить рядом[2] 4 члена псевдослучайной последовательности чисел, каждое из которых записывается 256 битами. В этой конструкции используются так называемые хеш-функции.

Конец четвёртой лекции.

[2]Эта операция называется конкатенацией.

Лекция 5.

2.8 Построение больших простых чисел.

Алгоритмы построения больших простых чисел носят рекурсивный характер. Чтобы построить большое простое число, нужно построить возрастающую последовательность простых чисел. Число нужного размера появляется в конце этой последовательности, члены же её принадлежат последовательности уменьшающихся промежутков, с построения которых начинается алгоритм. Эта конструкция вместе со всеми пояснениями обсуждается в настоящем параграфе.

Сначала мы рассмотрим следующую задачу: Для каждого заданного натурального $N \geq 11$ **построить простое число Q, лежащее на промежутке** $2^{N-1} \leq Q < 2^N$.

1. Для каждого числа $N \leq 18$ требуемое простое число можно найти непосредственно. Например, на промежутке от 2^{17} до 2^{18} их имеется 10749 штук, и каждое составное число этого промежутка делится на какое нибудь простое $p < 2^9 = 512$. Для нахождения простого числа в нужном маленьком промежутке (правый конец не превосходит 2^{18}) можно воспользоваться алгоритмом пробных делений, см. подраздел 2.10.1.

Далее будет найдена последовательность целых чисел a_k, b_k, q_k таких что

$$q_k \text{ - простые}, \quad 2^{a_k} \leq q_k < 2^{b_k}, \quad a_k = b_k - 1, \quad 1 \leq k \leq m, \quad b_m = N.$$

Тогда $Q = q_m$ - искомое простое число.

2. Пусть N - натуральное число, превосходящее 18. Построим последовательность целых чисел, начинающуюся с N, в которой за числом ℓ следует число $\lceil \frac{\ell}{2} \rceil + 1$. Легко видеть, что эта последовательность убывает при $\ell \geq 4$. Более того, в ней обязательно встретится целое число из промежутка $11 \leq \ell \leq 18$. Кроме того, не трудно проверить, что при любом $\ell \geq 19$ выполняется неравенство $\lceil \frac{\ell}{2} \rceil + 1 \geq 11$.

Например, для $N = 1024$ эта последовательность имеет вид

$$N = 1024, \quad 513, \quad 258, \quad 130, \quad 66, \quad 34, \quad 18.$$

Итак,

с какого бы числа N ни начиналась указанная последовательность, она обязательно содержит единственное число из промежутка $11 \leq x \leq 18$.

Начиная с этого числа, перенумеруем в обратном порядке все члены последовательности буквами $b_1, b_2, \ldots, b_m$. *Тогда* $11 \le b_1 \le 18$ *и* $b_m = N$. *Для каждого индекса* k, $1 \le k \le m$, *положим* $a_k = b_k - 1$.

3. Простое число q_1 выбирается так, как это указано в пункте 1, ведь $11 \le b_1 \le 18$ и $2^{a_1} < q_1 < 2^{b_1}$. Например, в случае $b_1 = 11$ на роль q_1 может быть взято любое из 137 простых чисел, лежащих на отрезке от 1024 до 2048, скажем 1933.

Предположим теперь, что $k \ge 2$ и простое число q_{k-1} уже построено. Для краткости обозначим

$$\ell = b_k, \quad q = q_{k-1}. \quad \text{Тогда} \quad a_k = \ell - 1, \quad b_{k-1} = \lceil \frac{\ell}{2} \rceil + 1, \quad a_{k-1} = \lceil \frac{\ell}{2} \rceil.$$

Число q_k будет строиться с помощью Следствия (2) из теоремы (7), см. также алгоритм (5). При этом $F = q$, $R = 2t$, где t - некоторое натуральное, оно будет выбираться с помощью испытаний, и

$$c = 2tq + 1 = FR + 1. \tag{2.23}$$

Для того, чтобы число c попало в нужный интервал, т.е. $2^{\ell-1} < c < 2^{\ell}$, на параметр t накладываются условия

$$\frac{2^{\ell-1}}{2q} \le t \le \frac{2^{\ell} - 1}{2q}. \tag{2.24}$$

Действительно, в этих ограничениях имеем

$$c = 2tq + 1 \ge 2^{\ell-1} + 1 > 2^{\ell-1} = 2^{a_k}$$

и

$$c = 2tq + 1 < 2^{\ell} = 2^{b_k}.$$

Строгое неравенство выполняется в силу того, что c нечётно, а 2^{ℓ} чётное число. Не трудно проверить, что для любых положительных чисел $u < v$ на интервале $u < t < v$ лежит не менее $v - u - 1$ целых чисел t. Согласно неравенству (2.24) можно утверждать, что количество целых t, для которых число c принадлежит интервалу $2^{\ell-1} < t < 2^{\ell}$ не меньше, чем $\frac{2^{\ell-}}{2q} - 1$. Например, при $b_1 = 11, \ell = b_2 = 19$ и $q = 1933$ на промежутке $2^{18} < c < 2^{19}$ имеется 10 простых чисел вида $2tq + 1$. Самые маленькие из них соответствуют числам $t = 65, 72, 75, 77, 81$. При $t = 77$ имеем $c = 297683$.

4. Как правило, строящиеся большие простые числа должны иметь случайный характер. Поэтому на интервале от $2^{\ell-1}$ до 2^{ℓ} выбирается случайное число x, а затем от него начанается отсчёт чисел t, соответствующие им числа c проверяются на простоту. Как только попадётся простое c, оно выбирается в качестве следующего простого q_k. Если при очередном t соответствующее число c выйдет за верхний предел заданного промежутка, следующее значение t принимается равным наименьшему возможному значению и вычисления продолжаются.

Выберем псевдослучайное целое число x на промежутке $2^{\ell-1} \le x < 2^{\ell}$ и положим $t = \lceil \frac{x}{2q} \rceil$. Определим также число c равенством (2.23).

Из неравенства $x \ge 2^{\ell-1}$ следует (2.23).

5. *Если t оказалось настолько большим, что*

$$2tq + 1 \ge 2^{\ell},$$

то полагаем $t = \lceil \frac{2^{\ell-1}}{2q} \rceil$. Если с помощью следствия 2 при $F = q$ и $R = 2t$ выбирая псевдослучайные числа b на промежутке от 2 до $c - 2$, удаётся доказать, что число c простое, то выбираем $q_k = c$. Для того, чтобы с помощью следствия 2 можно было доказать простоту числа c, параметры R и F должны быть связаны неравенством $R \le 4F + 2$, которое может быть записано в виде $t \le 2q + 1$. Используя границы для t и q, находим

$$q^2 > 2^{2\lceil \frac{\ell}{2} \rceil} \ge 2^{\ell} > 2qt \quad \text{и} \quad t < \frac{1}{2}q < 2q + 1,$$

так что следствие 2 применимо.

Если $k = m$, алгоритм завершает свою работу. В противном случае увеличиваем k на единицу и переходим в пункт 3 к следующему промежутку $[a_k, b_k)$.

6. *Если же c оказывается составным, то параметр t увеличивается на 1 и алгоритм переходит в пункт 5.*

Следующий далее алгоритм, по заданным числам L и N строит пары простых чисел P, Q с условиями

$$Q|(P-1), \qquad 2^{L-1} \le P < 2^{L}, \qquad 2^{N-1} \le Q < 2^{N}.$$

Мы ограничимся здесь только формальным описанием его основных моментов.

Алгоритм 6. Даны *натуральные числа* L *и* $N, L > N$.

Построить *"случайные"простые числа* P, Q *битовой длины* L *и* N *с условием* $Q|(P-1)$.

1. С помощью описанного выше алгоритма построить "случайное" простое число Q *с длиной записи* N *битов.*

2. Построить последовательность простых чисел p_k*, связанных соотношениями*

$$p_k = 2tQp_{k-1} + 1, \quad 1, \ldots, m, \qquad 2^{L-1} \le p_m < 2^L.$$

Для проверки на простоту чисел $c = 2tQp_{k-1} + 1$ *использовать следствие 2 с параметрами* $F = p_{k-1}$, $R = 2tQ$.

3. Положить $P = p_m$.

2.9 Первообразные корни и дискретное логарифмирование.

Пусть p – простое нечётное число. Множество классов вычетов целых чисел по модулю p составляет поле $\mathbb{F}_p$, а ненулевые элементы этого поля образуют циклическую мультипликативную группу порядка $p-1$. Согласно малой теореме Ферма выполняется сравнение $a^{p-1} \equiv 1 \pmod{p}$. Наименьшее натуральное число d с условием $a^d \equiv 1 \pmod{m}$ называется *показателем* числа a по модулю p. Например, показатель числа 1 по модулю p равен 1, а показатель -1 по нечётному модулю p равен 2. *Целое число* a*, не делящееся на* p*, называется первообразным корнем по модулю* p*, если его показатель равен* $p-1$. Например, показатель числа 5 по модулю 23 равен 22. Мы проверим это позже. Число 5 есть первообразный корень по модулю 23. Его класс вычетов по модулю 23 порождает мультипликативную группу всех ненулевых классов вычетов по модулю 23.

В 1801 году К.Ф. Гаусс опубликовал два доказательства следующей теоремы.

Теорема 8. *Для каждого простого нечетного числа* p *существуют первообразные корни по модулю* p*. Количество не сравнимых друг с другом по модулю* p *первообразных корней равно* $\varphi(p-1)$*, где* $\varphi(n)$ *– функция Эйлера.*

Для практического нахождения первообразных корней при небольших p удобно пользоваться следующим утверждением.

Теорема 9. *Целое число g, не делящееся на простое нечетное p, будет первообразным корнем по модулю p в том и только том случае, если для любого простого числа q, делящего $p-1$, выполняется*

$$g^{\frac{p-1}{q}} \not\equiv 1 \pmod{p}.$$

Если известны все простые делители числа $p-1$, то проверка условий теоремы 9 выполняется достаточно быстро с помощью алгоритма, изложенного в параграфе 2.3. Вот пример.

Справедливы сравнения по модулю 23

$$5^{11} = 5 \cdot 25^5 \equiv 5 \cdot 2^5 \equiv -1 \pmod{23}$$

и $5^2 \equiv 2 \pmod{23}$. Применяя теорему 9 к простому числу $p = 23$ и $c = 5$, заключаем, что 5 есть первообразный корень по модулю 23.

Множество первообразных корней при заданном p достаточно велико, поэтому выбирая числа g случайным образом на промежутке $0 < g < p$, можно с большой вероятностью попасть на первообразный корень и доказать это с помощью теоремы 9.

Если g – первообразный корень по модулю p, то числа

$$1, g, g^2, \ldots, g^{p-2} \tag{2.25}$$

имеют разные наименьшие неотрицательные остатки. Множество (2.25) состоит из $p-1$ чисел. Поэтому для каждого целого числа a, не делящегося на p, найдется единственное целое k, удовлетворяющее условиям

$$a \equiv g^k \pmod{p}, \qquad 0 \le k < p-1. \tag{2.26}$$

Число k называют *индексом* a по модулю p при основании g. Это определение и формулируемые ниже свойства индексов напоминают свойства обычных логарифмов действительных чисел. Поэтому иногда в современной литературе индексы называют дискретными логарифмами, а процесс их нахождения – дискретным логарифмированием. Используются также обозначения $ind_g a$, $Log_g a$. Указание на первообразный корень иногда опускается. Итак, имеем

$$a \equiv g^{ind_g a} \pmod{p}, \qquad 0 \le ind_g a < p-1. \tag{2.27}$$

Свойства индексов описывает следующие простые утверждения.

1. Если целые числа a, b не делятся на p, то

$$ind_g ab \equiv ind_g a + ind_g b \pmod{p-1}.$$

2. Если a, b – первообразные корни по модулю p, то для любого числа c, не делящегося на p, выполняется сравнение

$$ind_b c \equiv ind_b a \cdot ind_a c \pmod{p-1}.$$

Сформулируем теперь задачу дискретного логарифмирования по простому нечетному модулю.

Дано простое число p. Для заданных чисел $a, b \in \mathbb{Z}$, не делящихся на p, требуется решить сравнение

$$b^x \equiv a \pmod{p}. \tag{2.28}$$

Решение этой задачи очень трудоемко в вычислительном отношении. Не случайно в конце практически всех учебников по элементарной теории чисел приводятся таблицы индексов (дискретных логарифмов).

Лучшие из известных алгоритмов дискретного логарифмирования по простому модулю p, использующие вычисления в полях алгебраических чисел, требуют $O(e^{2(\ln p)^{1/3}(\ln\ln p)^{2/3}})$ арифметических операций. Впрочем, эта оценка условна, ибо опирается на ряд недоказанных, но весьма правдоподобных гипотез теории чисел.

Рекордный по величине простого числа p результат в задаче дискретного логарифмирования был установлен в декабре 2019 года. Международной группе математиков и программистов удалось сосчитать дискретный логарифм по модулю простого числа, записываемого 240 десятичными цифрами (795 битами). Применённый ими алгоритм требует достаточно глубоких познаний в теории алгебраических чисел, и мы его здесь не объясняем.

Излагаемый далее метод решения сравнения (2.28) требует $O(p^{1/2}\ln p)$ арифметических операций. Он был придуман в 1962г. А.О.Гельфондом и в русскоязычной литературе называется "метод согласования". Аналогичный метод был предложен в 1971г. Д.Шенксом. Его английское название может быть переведено как "метод больших и малых шагов".

Лемма 1. *Пусть p — простое нечетное число и пусть $H = [\sqrt{p}] + 1$. Тогда для каждого целого числа d, $1 \le d < p$, найдутся целые числа u, v, удовлетворяющие условиям*

$$1 \le u, v \le H, \qquad Hu - v = d.$$

Доказательство. Положим

$$u = \left[\frac{d}{H}\right] + 1, \qquad v = Hu - d.$$

Тогда

$$\frac{d}{H} < u \leq \frac{d}{H} + 1,$$

откуда видим, что, во–первых,

$$0 < u < \frac{p}{\sqrt{p}} + 1 = \sqrt{p} + 1,$$

а во–вторых, $0 < Hu - d \leq H$. Остается воспользоваться тем, что числа u и v целые. □

Алгоритм 7 (Алгоритм Гельфонда)**.** Данные: *Простое число* $p \geq 3$, *первообразный корень* g *по модулю* p, *число* $b \in \mathbb{Z}$, $p \nmid b$.
Найти: *Решение сравнения* $g^x \equiv b$.
1. Вычислить $H = [\sqrt{p}] + 1$.
2. Положить $c \equiv g^H \bmod p$, $1 \leq c < p$.
3. Составить два набора чисел

$$S_1 = \{c^u \bmod p : 1 \leq u \leq H\}, \quad S_2 = \{bg^v \bmod p : 1 \leq v \leq H\}.$$

4. Упорядочить по возрастанию оба набора S_1 *и* S_2. *Найти совпавшие элементы этих наборов, то есть такие числа* u, v, *для которых*

$$c^u \equiv bg^v \pmod{p}. \tag{2.29}$$

5. Положить

$$ind_g b = Hu - v. \tag{2.30}$$

Решение заданного сравнения имеет вид $x \equiv ind_g b \pmod{p-1}$.

Пересечение множеств S_1 и S_2 не пусто, так как сравнение (2.29) равносильно представлению (2.30), а последнее, согласно лемме 1, существует всегда.

Вычисления в пунктах 1 и 2 требуют $O(\ln p)$ арифметических операций. Вычисления в пункте 3 требуют $O(H \ln p) = O(\sqrt{p} \ln p)$ арифметических операций. Для упорядочения каждого из множеств S_1, S_2 нужно $O(H \ln H) = O(\sqrt{p} \ln p)$ арифметических операций. Для нахождения одинаковых элементов в упорядоченных множествах S_1, S_2 нужно $O(H) = O(\sqrt{p})$ арифметических операций. Общее же количество операций в алгоритме Гельфонда равно $O(\sqrt{p} \ln p)$.

Пример. Решить сравнение

$$2^x \equiv 23 \pmod{37}. \tag{2.31}$$

Решение. Так как $36 = 2^2 \cdot 3^2$ и

$$2^{18} \equiv -1 \not\equiv 1 \pmod{37}, \qquad 2^{12} \equiv 26 \not\equiv 1 \pmod{37},$$

то 2 - первообразный корень по модулю 37 и, значит данное сравнение разрешимо. В соответствии с алгоритмом вычисляем $H = [\sqrt{37}] + 1 = 7$ и $c = 2^7 \equiv 17 \pmod{37}$. Множества S_1 и S_2 состоят из чисел $c^n \pmod{37}$, и $b \cdot a^n \pmod{37}$ при $n = 1, 2, \ldots, 7$, т.е.

$$S_1 = \{17, 30, 29, 12, 19, 27, 15\}, \qquad S_2 = \{9, 18, 36, 35, 33, 29, 21\}.$$

Эти два множества имеют общий элемент 29, который стоит на $u = 3$ месте в первом множестве и на $v = 6$ месте во втором множестве. Таким образом, $ind_2 23 = 7 \cdot 3 - 6 = 15$, а решение сравнения (2.31) имеет вид $x \equiv 15 \pmod{37}$.

Если в сравнении (2.28) число b не является первообразным корнем по модулю p, то это сравнение можно переписать в равносильном виде

$$x \cdot ind_g b \equiv ind_g a \pmod{p-1}. \tag{2.32}$$

Индексы чисел a, b можно вычислить с помощью алгоритма Гельфонда. После этого можно решить линейное сравнение (2.32), см. параграф **??**. Все найденные числа x составят решения (2.28). Если сравнение (2.32) не имеет решений, то не будет их иметь и сравнение (2.28).

Алгоритм Гельфонда, конечно, быстрее полного перебора, но он работает недопустимо медленно, чтобы решать задачу дискретного логарифмирования для простых чисел размером в 250 десятичных цифр.

Конец пятой лекции.

Лекция 6.

2.10 Задача разложения целых чисел на множители.

В этом разделе будут рассматриваться простейшие методы разложения целых чисел на простые сомножители, т.е. методы поиска для заданного целого $N > 1$ простых чисел $p_1, \ldots, p_r$ таких, что

$$N = p_1^{k_1} \cdots p_r^{k_r}, \qquad k_j \geq 1,\ k_j \in \mathbb{Z}.$$

При этом будет предполагаться, что разлагаемое число N составное, в чем можно убедиться с помощью тестов из параграфа 2.5.

Достаточно уметь решать более простую задачу о разложении целого числа на два меньших множителя, т.е. задачу о решении в целых числах $a > 1, b > 1$ уравнения $N = ab$. Действительно, в этом случае выполняются неравенства $a < N, b < N$, можно разложить на два меньших множителя каждое из чисел a, b, и продолжать далее эту процедуру, пока такое разложение будет возможным, т.е. до тех пор, пока все сомножители не станут простыми.

Существующие алгоритмы разложения чисел на множители могут быть распределены на группы в зависимости от количества арифметических операций, которые алгоритм требует для своей работы.

1) *Экспоненциальные* алгоритмы используют $O(N^c)$ арифметических операций. Здесь c — положительная постоянная.

2) *Субэкспоненциальные* алгоритмы требуют $O(e^{c(\ln N)^\alpha (\ln \ln N)^\beta})$ арифметических операций. Здесь α, β, c — положительные постоянные, $\alpha + \beta = 1$. Заметим, что при $\alpha = 1$, т.е. $\beta = 0$, оценка совпадает с оценкой сложности экспоненциальных алгоритмов.

Для наиболее быстрого из субэкспоненциальных алгоритмов, так называемого *метода решета числового поля*, или метода просеивания в числовых полях имеем $\alpha = \frac{1}{3}, \beta = \frac{2}{3}$.

Алгоритмы полиномиальной сложности, $\alpha = 0, \beta = 1$, для задачи факторизации не известны, и весьма вероятно, что их не существует.

Деятельность по разложению чисел на множители сочетает в себе черты инженерной науки, поскольку во многом опирается на допущения, основанные на опыте и не имеющие теоретических обоснований, а с другой стороны она сродни искусству, так как зачастую продолжительность работы алгоритма и результат зависят от удачного выбора параметров.

2.10.1 Алгоритм пробных делений.

Пусть $d_1 < d_2 < \ldots$ — последовательность целых чисел, содержащая все простые числа. Алгоритм пробных делений состоит в последовательном делении N на числа $d_1, d_2, \ldots$, не превосходящие $\sqrt{N}$. Если N составное число, то оно имеет простой делитель $p \le \sqrt{N}$ и потому будет разложено на множители.

Этот алгоритм часто используется для нахождения всех простых делителей числа N, не превосходящих некоторой заданной границы B.

Последовательность d_i может совпадать с множеством простых чисел. Но в этом случае необходим алгоритм, строящий все простые числа до заданной границы. Иногда бывает проще использовать последовательности, содержащие и составные числа, но менее сложные в реализации.

Пример 1. Каждое простое число $p > 6$ удовлетворяет одному из двух сравнений $p \equiv 1 \pmod 6$ или $p \equiv 5 \pmod 6$. Поэтому последовательность

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 25, \ldots$$

содержащая все числа вида $6n \pm 1, n \in \mathbb{N}$, включает в себя и все простые числа. Но не только их. Например, она содержит 25 и 91, и многие другие составные числа. Но простое правило порождения этой последовательности облегчает поиск делителей N:

$$d_1 = 2,\ d_2 = 3,\ d_3 = 5, \quad d_{2k} = d_{2k-1} + 2, \quad d_{2k+1} = d_{2k} + 4, \quad k \ge 2.$$

При таком выборе d_i алгоритм вообще говоря требует $O(N^{1/2})$ арифметических операций и $O(1)$ памяти. Конечно, он очень медленный, но позволяет быстро отсеивать составные числа, имеющие малые простые делители.

Пример 2. В качестве модуля в сравнениях вместо 6 можно взять число $m = 2 \cdot 3 \cdot 5 = 30$. Все простые числа $p > 5$ будут содержаться в восьми прогрессиях с разностью 30, начинающихся с чисел $1, 7, 11, 13, 17, 19, 23, 29$, т.е.

$$30k + 7, 30k + 11, 30k + 13, 30k + 17, 30k + 19, 30k + 23, 30k + 29, 30k + 31.$$

Соответствующая последовательность d_i порождается формулами

$$\begin{aligned}
d_1 = 2, d_2 = 3, d_3 = 5, &\qquad d_{8k+4} = d_{8k+3} + 2,\\
d_{8k+5} = d_{8k+4} + 4, &\qquad d_{8k+6} = d_{8k+5} + 2,\\
d_{8k+7} = d_{8k+6} + 4, &\qquad d_{8k+8} = d_{8k+7} + 2,\\
d_{8k+9} = d_{8k+8} + 4, &\qquad d_{8k+10} = d_{8k+9} + 6,\\
d_{8k+11} = d_{8k+10} + 2, &\qquad d_{8k+12} = d_{8k+11} + 6, \quad k \ge 0.
\end{aligned}$$

Соответствующая последовательность $\{d_i\}$ будет содержать меньшую долю составных чисел. Если $m = \prod_{p \le r} p$, то количество арифметических прогрессий, составляющих эту последовательность равно $\varphi(m)$, а доля чисел из $\{d_i\}$ в натуральном ряде есть $\dfrac{\varphi(m)}{m} = \prod_{p \le r} \left(1 - \dfrac{1}{p}\right)$. При $m = 6$ она равна $\dfrac{1}{3}$, а при $m = 30$ имеем $\frac{1}{4}$.

2.10.2 ρ — метод Полларда.

Пусть $f(x)$ — "достаточно случайный"многочлен. Выберем "случайно" $x_1 \in \mathbb{Z}, 1 < x_1 < N$, и рассмотрим последовательность

$$x_{k+1} \equiv f(x_k) \pmod N, \qquad 0 \le x_{k+1} < N, \quad k \ge 1, \tag{2.33}$$

Так как количество классов вычетов по модулю N не превосходит N, то существуют такие индексы $i, j, 0 \le i, j < N$, что $x_i \equiv x_j \pmod N$ и значит, последовательность $x_k \pmod N$ зацикливается. Период этой последовательности не всегда начинается с самого начала, она может иметь предпериодическую часть. Символически такую последовательность можно изобразить в виде греческой буквы ρ. Подходящая снизу ножка этой буквы соответствует предпериодической части последовательности, она переходит в замкнутую петлю, соответствующую периодической части. Эта аналогия и дала название алгоритму.

Полларда обнаружил, что для простых p, как правило, длина периода и предпериодическая часть последовательности

$$x_{k+1} \equiv f(x_k) \pmod p, \qquad k \ge 1,$$

ограничены сверху величиной $c\sqrt{p}$, где c — некоторая константа.

Идея алгоритма состоит в том, чтобы последовательно вычислять наибольшие общие делители $(x_{2i} - x_i, N)$, $i = 1, 2, 3, \ldots$. Если p — простой делитель N, ℓ — длина периода и a — длина предпериодической части последовательности $x_{k+1} \equiv f(x_k) \pmod p$, то для номера i, удовлетворяющего условиям

$$i > a, \qquad \ell | i, \tag{2.34}$$

числа x_i и $x_{i+\ell}$ сравнимы друг с другом по модулю p. Мы не знаем чисел ℓ, i, но знаем, что разность $2i - i = i$ делится на ℓ, поэтому $x_{2i} \equiv x_i \pmod p$, так что $(x_{2i} - x_i, N) > 1$. Ясно, что существует число i, удовлетворяющее условиям (2.34) и неравенству $i \le a + \ell = O(\sqrt{p})$. Конечно, может случиться, что $N | x_{2i} - x_i$. Тогда нужно выбирать иное начальное значение x_1.

Если p — наименьший простой делитель N, то $p \leq N^{1/2}$, так что $i = O(N^{1/4})$.

В следующем ниже алгоритме вычисляются пары $\{x_i, x_{2i}\}, i \geq 1$. Для нахождения следующей пары $\{x_{i+1}, x_{2i+2}\}$ нужно вычислить

$$x_{i+1} \equiv f(x_i) (\mathrm{mod}\, N), \quad x_{2i+1} \equiv f(x_{2i}) (\mathrm{mod}\, N),$$
$$x_{2i+2} \equiv f(x_{2i+1}) (\mathrm{mod}\, N),$$

т.е. выполнить $O(1)$ арифметических операций.

Алгоритм 8. Дано: *составное число* N. Найти: *нетривиальный делитель* N .

1. Выбрать случайно $x_1 \in \mathbb{Z}, 1 < x_1 < N$, *и положить*

$$x = f(x_1) \pmod N, \qquad y = f(x) \pmod N.$$

2. Вычислить $d = (y - x, N)$. *Если* $1 < d < N$ *алгоритм останавливается, нетривиальный делитель* d *числа* N *найден.*

3. Если $d = N$, *перейти в п. 1.*

4. Положить

$$x = f(x) \pmod N, \quad z = f(y) \pmod N, \quad y = f(z) \pmod N$$

и перейти в пункт 2 алгоритма.

Если все удачно сложится, то нетривиальный делитель числа N будет найден за $O(p^{1/2})$ арифметических операций, где p — наименьший простой делитель N, т.е. за $O(N^{1/4})$ арифметических операций.

В силу оценки сложности $O(p^{1/2})$ алгоритм удобен для нахождения не очень больших делителей p числа N, если они есть.

В качестве $f(x)$ обычно выбираются многочлены вида $x^2 + a$. Например, можно взять $f(x) = x^2+1$ или $f(x) = x^2-1$. В то же время выбор $f(x) = x^2-2$ и $x_1 = 2$ не очень удачен, так как в этом случае последовательность x_k имеет период 1.

Приведем теперь некоторые правдоподобные соображения в пользу того, что длина периода и предпериодическая часть последовательности $x_{k+1} \equiv f(x_k) \pmod p$ оцениваются сверху величиной $O(p^{1/2})$.

2.10.3 Парадокс дней рождения.

Пусть $\mathcal{R}$ - множество, состоящее из r элементов. Из принципа ящиков Дирихле следует, что в любой последовательности $z_1, z_2, \ldots z_{r+1}$, $z_i \in \mathcal{R}$, обязательно найдутся два одинаковых элемента. Но можно проверить, что выбирая

случайным образом множество $z_1, z_2, \ldots, z_m$ с теми же условиями на z_i при достаточно большом, но всё же существенно меньшем r значении m, можно с вероятностью, большей $\frac{1}{2}$ попасть на набор, имеющий два одинаковых элемента. Этот кажущийся парадокс носит название "парадокс дней рождения" по причине, о которой мы расскажем немного позже, а сейчас докажем, при некоторых естественных предположениях, указанную оценку на вероятность "хорошей" выборки.

Будем выбирать случайным образом наборы

$$\overline{z} = \{z_1, \ldots, z_m\}, \qquad z_i \in \mathcal{R}, \tag{2.35}$$

предполагая, что все они равновероятны. Какова вероятность того, что выбранная таким образом совокупность $z_1, z_2, \ldots, z_m$, содержит по крайней мере два одинаковых элемента?

Каждое значение z_i может равняться одному из r элементов множества $\mathcal{R}$. Значит количество последовательностей вида (2.35) равно r^m. Сколько же среди них последовательностей с различными элементами. Попробуем построить такие последовательности. На первом месте может стоять любое из возможных r значений для z_1. На втором месте может стоять лишь $r-1$ значений, отличных от стоящего на первом месте. На третьем месте может стоять лишь $r-2$ значения, отличных от первых двух. Действуя так далее мы сможем построить $r(r-1)(r-2)\cdots(r-m+1)$ последовательностей, состоящих из m различных значений. Легко видеть, что каждая такая последовательность будет учтена в этом процессе. Следовательно, доля последовательностей с различными значениями z_i будет равна

$$\mu = \frac{r(r-1)...(r-m+1)}{r^m}.$$

Для оценки μ сверху перепишем

$$\mu = \prod_{k=1}^{m-1}\left(1-\frac{k}{r}\right) \quad \text{и} \quad \ln\mu = \sum_{k=1}^{m-1}\ln(1-\frac{k}{r}). \tag{2.36}$$

При любом действительном x, $0 \le x < 1$ справедливо неравенство

$$\ln(1-x) \le -x. \tag{2.37}$$

Для его доказательства рассмотрим непрерывную на промежутке $[0;1)$ функцию $g(x) = x + \ln(1-x)$. Так как производная $g'(x) = \frac{-x}{1-x}$ отрицательна на интервале $(0;1)$, то $g(x)$ убывает на множестве $0 \le x < 1$, и, значит, на этом множестве выполняется неравенство (2.37).

Взяв $x = \frac{k}{r}$, находим $\ln(1 - \frac{k}{r}) < -\frac{k}{r}$, $1 \le k < m$ и из (2.36)

$$\ln \mu < -\sum_{k=1}^{m-1} \frac{k}{r} = -\frac{m^2 - m}{2r} < -\frac{(m-1)^2}{2r}.$$

Выберем теперь $m = \lceil \sqrt{2r \ln 2} \rceil + 1$. Тогда $(m-1)^2 = \lceil \sqrt{2r \ln 2} \rceil^2 \ge 2r \ln 2$ и $\ln \mu < -\ln 2$. Так доказано неравенство $\mu < \frac{1}{2}$. Заметим, что при $r \ge 5$ выполняется $m < r$.

Если увеличить константу $\sqrt{2 \ln 2}$ в определении m, оценка μ уменьшится и, значит, вероятность попасть на набор, имеющий по крайней мере два равных элемента, станет больше $\frac{1}{2}$. Например, в случае $m = \lceil 4\sqrt{r} \rceil$ вероятность попасть на выборку, имеющую 2 или более одинаковых элементов, превосходит $0,9996\ldots$.

Вернёмся теперь к последовательности (2.33) и возьмём постоянную в определении m большую, чем $\sqrt{2 \ln 2}$. Из сказанного выше следует, что для "хорошего" многочлена $f(x)$ и быть может после нескольких случайных выборов числа x_1 мы за $O(\sqrt{p} \ln p)$ арифметических операций сможем найти последовательность $x_1, x_2, \ldots, x_m$, имеющую два равных числа. Длины предпериода a и периода ℓ этой последовательности удовлетворяют неравенству $a + \ell \le M$. Действительно, a есть номер первого из двух равных чисел, и $a + \ell$ - номер второго из них чисел. Это объясняет наблюдение Полларда о длине предпериода и периода последовательности в ρ - методе Полларда.

Конечно, это рассуждение не является доказательством, но оно дает правдоподобное объяснение, почему ρ — метод Полларда достаточно быстро на практике находит небольшие простые делители составных чисел.

Заметим также, что скорость работы этого алгоритма может быть увеличена, если наибольший общий делитель $(x_{2i} - x_i, N)$ вычислять не на каждом шаге. Например, можно для последовательности $i = 0, d, 2d, \ldots$ вычислять $(\prod_{k=i+1}^{i+d}(x_{2k} - x_k), N)$, выбрав d не очень большим.

Рассмотрим ещё одну ситуацию, котороая собственно и дала название корневому понижению оценки числа арифметических опереций при нахождении в последовательности равных элементов. Представим себе, что студенческая группа или просто группа знакомых собралась на вечеринку по какому-то поводу. В группе 23 человека и требуется определить, какова вероятность, что среди приглашенных есть двое, дни рождения которых приходятся на одну дату. На первый взгляд кажется, что эта вероятность очень маленькая. Но рассмотрим ситуацию чуть подробнее. Если год, в котором происходит это

событие не високосный, т.е. состоит из $p = 365$ дней, то вероятность выбрать 23 различные даты из 365 равна

$$\mu = \prod_{k=1}^{22} \left(1 - \frac{k}{365}\right) = 0,492703\ldots = \\ = \frac{369979785662179593401824991341667570443833518472560 64}{750918832685153501254262074252231475632698059082031 25}.$$

Последнее значение точное. Таким образом, любой случайный набор из 23 дат невисокосного года с вероятностью большей $\frac{1}{2}$ содержит две одинаковых даты.

Конец шестой лекции.

Лекция 7.

2.11 Эллиптические кривые.

Уравнение

$$y^2 = x^3 + ax + b, \tag{2.38}$$

с коэффициентами a, b, удовлетворяющими условию $4a^3+27b^2 \neq 0$, определяет на плоскости кривую, называемую *эллиптической*. Неравенство означает, что многочлен $x^3 + ax + b$ не имеет кратных корней. Докажем этот факт.

Лемма 2. *Многочлен $f(x) = x^3 + ax + b$ имеет кратные корни в том и только том случае, когда $4a^3 + 27b^2 = 0$.*

Доказательство. Обозначим корни производной $f'(x) = 3x^2 + a$ буквами x_1 и $x_2 = -x_1$ Для того, чтобы эти числа были действительными необходимо и достаточно выполнение неравенства $a \leq 0$. Подставим эти корни в многочлен $f(x)$ и перемножим получившиеся числа. В результате, учитывая, что $x_1^2 = -\frac{a}{3}$, получим

$$f(x_1)f(-x_1) = (x_1^3 + ax_1 + b)(-x_1^3 - ax_1 + b) = b^2 - (x_1^3 + ax_1)^2 = \\ = b^2 - x_1^2(x_1^2 + a)^2 = b^2 + \frac{4a^3}{27}.$$

Многочлен $f(x)$ имеет кратный корень в том и только том случае, когда он и его производная имеют общий корень, т.е. $f(x_1) = 0$ или $f(-x_1) = 0$, что по доказанному равносильно равенству $b^2 + \frac{4a^3}{27} = 0$ или равенству $4a^3 + 27b^2 = 0$. □

Если многочлен $f(x)$ имеет кратные корни, то кривая может быть параметризована рациональными функциями и её исследование сильно упрощается.

Например, для многочлена $x^3 - 3x + 2 = (x - 1)^2(x + 2)$ выполняется тождество

$$(t^3 - 3t)^2 = (t^2 - 2)^3 - 3(t^2 - 2) + 2,$$

означающее, что при любом t формулы $x = t^2 - 2, y = t^3 - 3t$ задают некоторую точку на кривой $y^2 = x^3 - 3x + 2$. Легко увидеть, что так может быть получена любая точка (x, y) этой кривой. Соответствующее значение переменной t определяется формулой $t = \frac{y}{x-1}$. Теперь достаточно проверить

равенства

$$\left(\frac{y}{x-1}\right)^2 - 2 = x + \frac{y^2 - (x-1)^2(x+2)}{(x-1)^2} = x,$$
$$\left(\frac{y}{x-1}\right)^3 - 3\frac{y}{x-1} = \frac{y}{(x-1)^3}(y^2 - (x-1)^2(x+2) + (x-1)^3) = y.$$

Точка кривой $(1, 0)$, для которой выражение $\frac{y}{x-1}$ не определено, получается по указанным формулам при $t = \sqrt{3}$.

Эллиптическими называются и другие кривые, уравнения которых с помощью взаимно-обратных преобразований, задаваемых рациональными функциями от переменных, могут быть преобразованы к виду (2.38).

Название эллиптические кривые происходит от названия эллиптические функции. Эти функции не выражаются через элементарные, но хорошо изучены, благодаря важной роли, которую они играют в приложениях. Через них, в частности, можно выразить длину дуги эллипса, с чем и связано их название. Эллиптические кривые параметризуются эллиптическими функциями так же как окружность параметризуется синусом и косинусом угла. Для определённости будем предполагать, что коэффициенты a, b уравнения (2.38) суть рациональные числа, такими же будут и точки кривых. Иногда кривые будут предполагаться размещёнными на действительной плоскости. В дальнейшем будут рассматриваться такие кривые и над конечными полями. Именно этот случай используется в криптографии.

Теория, описывающая свойства решений в рациональныхСами кривые числах или, как мы будем писать в дальнейшем *рациональных решений*, имеет большую историю и богата глубокими результатами. Тем не менее в ней есть естественные открытые вопросы, т.е. вопросы, ответы на которые не известны. Вот некоторые из них.

1. **Есть ли решение?** До сих пор не известен алгоритм, позволяющий для любого данного уравнения вида (2.38) с рациональными коэффициентами a, b определить, есть у него решение в рациональных числах или нет.

2. **Как найти решение, если известно, что оно существует?** То, что ответ на этот вопрос достаточно сложен, показывает следующий пример. Известно, что уравнение $y^2 = x^3 + 877x$ имеет бесконечно много рациональных решений. Решение этого уравнения с наименьшими знаменателями имеет вид

$$x = \frac{3754945281271621931055040699420927923462 01}{6215987776871505425463220780697238044100},$$

$$y = \frac{2562562679889268093887768340455130896486691532043566034647869 49}{4900780232197875889598029339959289250960616164707799792610 00}.$$

3. **Конечно или бесконечно множество решений?**
4. **Какова структура множества решений?**

2.11.1 Сложение точек на эллиптической кривой

Ещё древние греки, а впоследствии Ньютон и Эйлер придумали способы построения новых рациональных точек на эллиптических кривых по уже имеющимся. При этом по существу использовались методы построения касательных или секущих к эллиптическим кривым. Этот метод можно рассматривать не только как метод "размножения"рациональных точек, но и как некоторую операцию, выполняемую на множестве точек эллиптической кривой.

Пусть P и Q две точки кривой (2.38). Проведём через P и Q прямую линию и третью точку её пересечения с кривой отразим симметрично относительно оси OX. Получившуюся точку назовём суммой P и Q. Будем обозначать эту точку $P + Q$, см. рисунок 3.

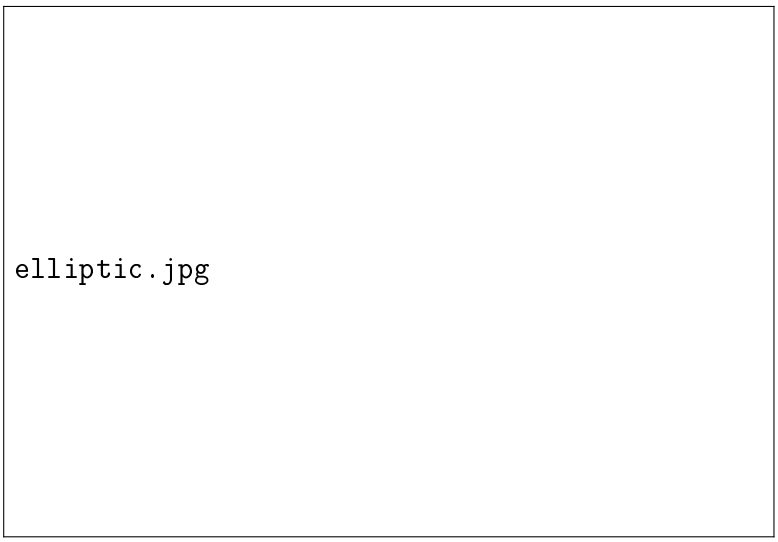

Рис. 2.1: Операция сложения точек на эллиптической кривой.

Если точки P и Q одинаковы, проведём касательную в точке P и точку пересечения с кривой отразим симметрично относительно оси OX. Получившуюся точку будем обозначать $2P$, см. правый рис. 4. Легко проверить с

помощью таких же соображений, что $P + O = P$, а также $P + R = O$, см. левый рисунок 4.

Рис. 2.2: Операция сложения точек на эллиптической кривой.

Операция сложения в координатах задается следующим образом.

Теорема 10. *Если* (x_1, y_1) - *координаты точки* P *и* (x_2, y_2) - *координаты точки* Q, *причём обе точки отличны от* O, *а также* $Q \neq -P$, *то координаты* (x_3, y_3) *точки* $P + Q$ *можно вычислить по формулам*

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2, \\ y_3 = \lambda(x_1 - x_3) - y_1, \end{cases} \quad \text{где} \quad \lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1}, & \text{если } x_2 \neq x_1, \\ \frac{3x_1^2 + a}{2y_1}, & \text{если } x_2 = x_1 \text{ и } y_2 = y_1. \end{cases} \tag{2.39}$$

Кроме того, по определению для любой точки P *выполняются равенства* $P + O = O + P = O$ *и* $P + (-P) = O$, *где* $-P = (x_1, -y_1)$.

Доказательство. В случае $x_2 \neq x_1$ тангенс угла наклона прямой, проходящей через точки P, Q равен $\frac{y_2 - y_1}{x_2 - x_1} = \lambda$ и уравнение этой прямой имеет вид

$$y - y_1 = \lambda(x - x_1). \tag{2.40}$$

В этом легко убедиться, подставив в уравнение (2.40) координаты точек P и Q. Из определения точки $P + Q$ следует, что числа $(x_3, -y_3)$ удовлетворяют уравнениям (2.38) и (2.40). Выразив переменную y через x из уравнения (2.40) и подставив получившееся выражение в уравнение (2.38), найдём кубическое уравнение

$$(y_1 + \lambda(x - x_1))^2 = x^3 + ax + b,$$

корнями которого являются абсциссы трёх точек пересечения кривой (2.38) и прямой (2.40), т.е. числа x_1, x_2, x_3. Полученное уравнение может быть переписано также в виде $x^3 - \lambda^2 x^2 + \ldots = 0$, где опущены слагаемые имеющие

степень по переменной x меньшую чем 2. Согласно теореме Виета сумма $x_1 + x_2 + x_3$ должна быть равна λ^2, откуда следует выражение для x_3 из равенств (2.39). Точка $(x_3, -y_3)$ лежит на прямой PQ. Из (2.40) теперь следует равенство $-y_3 - y_1 = \lambda(x_3 - x_1)$ и далее выражение для y_3 из (2.40). В первом случае терема доказана.

Рассмотрим теперь второй случай, когда $Q = P$. В этом случае вместо секущей прямой PQ рассматривается касательная к кривой (2.38), проведённая в точке P. В случае $y_1 = 0$ имеем равенство $P = -P$ и находим $P + P = P + (-P) = O$. Теорема доказана и в этом случае.

Далее считаем $Q = P$ и $y_1 \neq 0$. В этом случае можно найти на плоскости маленькую окрестность точки $P = (x_1, y_1)$, в которой уравнение (2.38) определяет $y = y(x)$ как однозначную функцию от переменной x, а именно $y(x) = \sqrt{x^3 + ax + b}$, где ветвь квадратного корня выбрана так, чтобы значение $\sqrt{x_1^3 + ax_1 + b}$ равнялось y_1. Эта функция удовлетворяет в малой окрестности точки x_1 тождеству $y(x)^2 = ax^3 + ax + b$. Дифференцируя это тождество по переменной x, находим $2y(x)y'(x) = 3x^2 + a$. Подставляя в полученное тождество $x = x_1$ и пользуясь равенством $y(x_1) = y_1$, находим

$$y'(x_1) = \frac{3x_1^2 + a}{2y_1} = \lambda.$$

Так что в этом случае число λ равно тангенсу угла наклона касательной к кривой, определённой равенством (2.38), проведённой в точке P. Далее все рассуждения проводятся в точности так же, как и в первом случае, с учётом равенства $Q = P$. □

Введённое выше определение суммы точек на первый взгляд кажется странным, однако оно обладает рядом удобных и полезных свойств: точка O играет такую же роль, как и ноль в множестве целых чисел, для каждой точки P есть единственная обратная, обозначаемая $-P$, т.е. точка с условием $P + (-P) = O$, операция сложения коммутативна, т.е. $P + Q = Q + P$ для любых двух точек P, Q и ассоциативна, т.е. для любых трёх точек P, Q, R эллиптической кривой выполняется равенство $(P + Q) + R = P + (Q + R)$ (доказательство последнего факта не просто). Другими словами множество точек эллиптической кривой с такой операцией становится *группой*.

В 1901г. французский учёный А. Пуанкаре (1854-1912) высказал предположение, что *все рациональные точки эллиптической кривой, определённой над полем рациональных чисел* $\mathbb{Q}$ *могут быть получены из конечного числа таких точек проведением хорд и касательных.* По другому: *группа рациональных точек имеет конечное число образующих.*

Гипотезу Пуанкаре доказал в 1922 году английский математик Л. Морделл (1888-1972).

Известен пример эллиптической кривой с числом образующих, не меньшим 28, но точное число образующих в этом случае не известно. Кривые с числом образующих группы рациональных точек большим 28 в настоящее время не найдены. Тем не менее предполагается, что

Существуют эллиптические кривые со сколь угодно большим числом независимых образующих.

Но эта гипотеза до сих пор не доказана. Далее мы не будем встречаться с эллиптическими кривыми, определёнными над полем рациональных или действительных чисел.

2.11.2 Эллиптические кривые над конечными полями.

Пусть $p > 3$ - простое число. Рассмотрим теперь уравнение (2.38), предполагая, что его коэффициенты принадлежат $\mathbb{F}_p$. Множество пар чисел x, y из $\mathbb{F}_p$, удовлетворяющих уравнению (2.38), также будем называть эллиптической кривой, добавляя при этом "над полем $\mathbb{F}_p$". Эта "кривая"состоит из конечного множества точек, ведь каждая переменная x и y может принимать не более p значений.

Например, кривая, определённая уравнением $y^2 = x^3 + 2$ над множеством $\mathbb{F}_{11}$ состоит из точек

$$(1,5)), (7,2), (6,3), (9,7), (10,1), (4,0), (10,10), (9,4), (6,8), (7,9), (1,6) \tag{2.41}$$

и ещё точки O, которую по аналогии тоже можно называть бесконечно удалённой точкой. Указанная эллиптическая кривая, определённая над полем $\mathbb{F}_{11}$, состоит из 12 точек.

То же уравнение над полем $\mathbb{F}_7$ определяет кривую

$$(0,3), (0,4), (3,1), (3,6), (5,1), (5,6), (6,1), (6,6), \tag{2.42}$$

состоящую из 9 точек: 8 указанных выше точек и ещё бесконечно удалённая точка O.

Точки кривой, определённой над конечным полем, тоже можно складывать. В этом случае, конечно, нельзя проводить касательные, но можно воспользоваться формулами (2.39), выполняя все вычисления в поле $\mathbb{F}$ по указанным в (2.39) формулам. И в этом случае точка O будет играть роль нуля, а операция сложения будет коммутативной и ассоциативной.

Например, если в случае кривой $y^2 = x^3 + 2$ обозначить $P = (1, 5)$, то все точки в (2.41) будут иметь вид

$$P,\ 2P,\ 3P,\ \ldots\ 11P,$$

а $12P = O$.

А для каждой точки Q кривой (2.42) выполняется равенство $3Q = O$.

Конец седьмой лекции и части 1 .