Разложение чисел на множители

Сергей Николенко

Криптография — АУ РАН, осень 2011

Outline

Идея

- Мы строили системы, чья надёжность основана на задачах разложения чисел на множители и дискретных логарифмах.
- И считали большим успехом, если получается свести надёжность системы к сложности решения такой задачи.
- Но что если сами эти задачи окажется легко решить?
- Сегодня мы будем говорить о том, как решать задачи, на которых основана современная криптография.

Простейший метод

- Простейший метод trial division — берём простые числа от 2 до $\sqrt{n}$ и делим на каждое.
- 1977: в колонке Мартина Гарднера появился the RSA-129 challenge: разложить 129-значное число на множители и тем самым взломать секретный код. Ривест тогда же подсчитал, что разложить 125-значное число — задача на 40 квадриллионов лет.
- Однако сейчас RSA-129 можно взломать, и вовсе не только потому, что компьютеры стали быстрее.

Задачка

- Начнём с задачки. Разложите на множители число

$$6319$$

Задачка

- Начнём с задачки. Разложите на множители число
$$6319$$
- Идея:
$$6319 = 6400 - 81 = 80^2 - 9^2 = 71 \times 89.$$
Часто ли нам будет так везти?

Задачка

- Начнём с задачки. Разложите на множители число

$$6319$$

- То же самое работает с любым нечётным составным числом:

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

- Это *метод Ферма* — ещё Ферма его предложил.

Метод Ферма

- Алгоритм: брать $x^2 - n$ для разных x и ждать другой квадрат.
- Например, для числа 6319 всё просто: ближайший квадрат сверху — это 6400, и когда мы вычитаем $80^2 - 6139 = 81$, сразу получаем квадрат.
- Но здесь везения нужно не меньше, чем для пробного деления.

С двух сторон

- Метод пробного деления хорошо работает для очень гладких чисел, когда быстро найдутся простые делители.
- Метод Ферма — для максимально «негладких» чисел, когда делители близки к $\sqrt{n}$.
- Но, конечно, сами по себе они одинаково ужасны.

Метод Крайчика

- 1920s, Maurice Kraitchik, бельгийский математик.
- Улучшение метода Ферма: вместо таких u и v, что

$$u^2 - v^2 = n,$$

можно искать такие u и v, что

$$u^2 - v^2 \equiv 0 \pmod{n}.$$

Метод Крайчика

- Если $u \not\equiv \pm v \pmod{n}$, то из

$$u^2 - v^2 \equiv 0 \pmod{n}$$

тоже получится разложение: n делится на $(u - v)(u + v)$, но не делится ни на $u - v$, ни на $u + v$, следовательно, НОД$(u - v, n)$ — нетривиальный делитель n.

Метод Крайчика

- Алгоритм: по-прежнему брать $x^2 - n$ для разных x.
- Но теперь не ждать другой квадрат, а пытаться его построить, умножая полученные числа:
  - рассмотрим последовательность чисел вида $Q(x) = x^2 - n$;
  - если
    $$Q(x_1)Q(x_2)\dots Q(x_k) = v^2,$$
    то сразу получается, что
    $$(x_1^2 - n)(x_2^2 - n)\dots(x_k^2 - n) = v^2, \text{ т.е.}$$
    $$u^2 = x_1^2 x_2^2 \dots x_k^2 \equiv v^2 \pmod{n}.$$

Метод Крайчика: пример

- Пример: рассмотрим $n = 2041$.
  - Здесь $Q(x)$ начинается с $46^2 = 2116$. Последовательность $Q(46), Q(47), \ldots: 75, 168, 263, 360, 459, 560, 663, \ldots$
  - Квадратов пока нет, но можно заметить, что

$$\begin{array}{rcl rcl} 75 & = & 3 \times 5^2, & 360 & = & 2^3 \times 3^2 \times 5, \\ 168 & = & 2^3 \times 3 \times 7, & 560 & = & 2^4 \times 5 \times 7. \end{array}$$

  - И, поигравшись, вытащить из этого квадрат:

$$75 \times 168 \times 360 \times 560 = 2^{10} \times 3^4 \times 5^4 \times 7^2.$$

  - Теперь

$$\begin{array}{rcl} u & = & 46 \times 47 \times 49 \times 51 \equiv 311 \pmod{2041}, \\ v & = & 2^5 \times 3^2 \times 5^2 \times 7 \equiv 1416 \pmod{2041}, \end{array}$$

    они нам подходят, $\text{НОК}(1416 - 311, 2041) = 13$, и $2041 = 13 \times 157$.

Метод Крайчика: алгоритм

- Итак, вот какой алгоритм из метода Крайчика получается.
    1. Выписать последовательность чисел вида $Q(x) = x^2 - n$.
    2. Поиграться с этими числами так, чтобы в произведении $Q(x_1)Q(x_2)\dots Q(x_k)$ получился квадрат v^2.
    3. Разложить n при помощи $u = x_1x_2\dots x_k$ и v, если $u \not\equiv \pm v \pmod{n}$.
- Но что значит «поиграться»?

Метод Крайчика: алгоритм

- Итак, вот какой алгоритм из метода Крайчика получается.
  1. Выписать последовательность чисел вида $Q(x) = x^2 - n$.
  2. Поиграться с этими числами так, чтобы в произведении $Q(x_1)Q(x_2)\dots Q(x_k)$ получился квадрат v^2.
  3. Разложить n при помощи $u = x_1x_2\dots x_k$ и v, если $u \not\equiv \pm v \pmod{n}$.
- Но что значит «поиграться»?
- Всё просто: решить систему линейных уравнений на степени маленьких простых чисел.

Гладкие числа

- *Гладкое число* (smooth number) — число, у которого только маленькие простые делители; Y-гладкое — значит, все делители $\leq Y$.
- Проверять на Y-гладкость можно простым перебором простых чисел до Y; а можно и лучше, но об этом чуть позже.

Гладкие числа для нашего примера

- У нас были 7-гладкие числа; для них можно ввести вектор экспонент размерности, соответствующей к-ву простых чисел до 7, причём векторы нам нужны только $\mod 2$:

$$\begin{aligned}
v(75) &= (0,1,2,0) &\equiv (0,1,0,0) \pmod 2,\\
v(168) &= (3,1,0,1) &\equiv (1,1,0,1) \pmod 2,\\
v(360) &= (3,2,1,0) &\equiv (1,0,1,0) \pmod 2,\\
v(560) &= (4,0,1,1) &\equiv (0,0,1,1) \pmod 2,
\end{aligned}$$

- Теперь видно, что сумма этих векторов $\equiv (0,0,0,0) \pmod 2$, а значит, получится квадрат.

Упражнение. Можно рассматривать и отрицательные вспомогательные числа ($x^2 - n$ для $x < \sqrt{n}$). Но квадрат отрицательным быть не может. Как учесть это в методе?

Непрерывные дроби

- Вместо значений $Q(x) = x^2 - n$ можно рассматривать приближения к $\sqrt{n}$ в виде непрерывной дроби с последовательностью приближений $\frac{a_i}{b_i}$.
- Тогда можно заменить $Q(x)$ на $Q_i = a_i^2 - b_i^2 n$.
- Для квадратного корня есть простые соотношения между a_i и b_i, а теория непрерывных дробей даёт неравенство $|Q_i| < 2\sqrt{n}$, в то время как $Q(x)$ линейно растут.
- И за счёт меньших Q_i проверять гладкость и подбирать соотношения становится проще.

Оценка сложности

- Обозначим через $\psi(X, Y)$ количество Y-гладких чисел от 1 до X. Вероятность того, что случайное число $\leq X$ является Y-гладким, равна $\frac{\psi(X,Y)}{X}$.
- Нам нужно решить систему из $\pi(Y)$ уравнений (по числу простых); чтобы решилась, нужно примерно $\pi(Y)$ неизвестных (коэффициентов).
- Т.е. нужно найти $\pi(Y)$ простых чисел. Проверить число на Y-простоту — это ещё $\pi(Y)$ шагов. Итого ожидаемое время работы

$$\frac{\pi^2(Y)X}{\psi(X, Y)}.$$

Оценка сложности

- Мы хотим для данного X минимизировать выражение

$$\frac{\pi^2(Y)X}{\psi(X, Y)}$$

- Попробуем оценить $\psi(X, Y)$ для $Y = \sqrt{X}$:

$$\psi(X, X^{1/2}) = \lfloor X \rfloor - \sum_{\sqrt{X} < p \leq X} \lfloor X/p \rfloor =$$

$$= X\left(1 - \sum_{\sqrt{X} < p \leq X} \frac{1}{p}\right) + O\left(\frac{X}{\log X}\right).$$

Факты из теории чисел

- Мы будем пользоваться фактами из теории чисел. Для этого примера — теорема Мертенса:

$$\sum_{p<Y} \frac{1}{p} = \ln\ln Y + \gamma + O\left(\frac{1}{\log Y}\right), \text{ где } \gamma \approx 0,261497\ldots$$

- Значит,

$$\sum_{\sqrt{X}<p\leq X} \frac{1}{p} = \sum_{p\leq X} \frac{1}{p} - \sum_{p\leq\sqrt{X}} \frac{1}{p} =$$

$$= \log\log X - \log\log\sqrt{X} + O\left(\frac{1}{\log\sqrt{X}}\right) = \log 2 + O\left(\frac{1}{\log\sqrt{X}}\right), \text{ и}$$

$$\psi(X, X^{1/2}) = (1-\log 2)X + O\left(\frac{X}{\log X}\right).$$

Факты из теории чисел

- Аналогично, $\psi(X, X^{1/u}) = (1 - \log u)X + O(\frac{X}{\log X})$ для $u \in [1, 2]$.
- Есть обобщение этого результата, которое нам и нужно (без доказательства): для любого $\epsilon > 0$, если $X \to \infty$, $u \to \infty$, причём $X^{1/u} > (\log X)^{1+\epsilon}$, то

$$\frac{\psi(X, X^{1/u})}{X} = u^{-(1+o(1))u}.$$

То, что мы минимизируем

- А нам нужно минимизировать $\frac{\pi^2(Y)X}{\psi(X,Y)}$, то есть, выразив $Y = X^{1/u}$, примерно $\frac{X^{2/u}X}{u^{-u}\log^2\left(X^{1/u}\right)}$, что будет минимизироваться при

$$u \approx 2\sqrt{\frac{\log X}{\log\log X}} \quad \text{(проверьте!)}.$$

Оценка сложности

- Итак, минимум будет достигаться, когда Y порядка $e^{\frac{1}{2}\sqrt{\log X \log\log X}}$, а сам минимум при этом порядка $e^{2\sqrt{\log X \log\log X}}$.
- Здесь X — это средний размер числа, которое получается из метода; это порядка $2\sqrt{n}$ для метода непрерывных дробей и $n^{1/2+\epsilon}$ для $Q(x) = x^2 - n$ (мы будем перечислять x до $\sqrt{n} + n^{\epsilon}$).
- Иначе говоря, оценка сложности получается порядка $e^{\sqrt{2\log n \log\log n}}$.

Сложностные обозначения

- В этой теории меняются только две константы, поэтому вводят обозначение

$$L_n[s;c] = e^{c(\log n)^s(\log\log n)^{1-s}}.$$

- Для сравнения асимптотики надо сначала сравнить s, а если они равны, то c.
- Наша оценка сложности для метода Крайчика в этих обозначениях:

$$e^{\sqrt{2\log n\log\log n}} = L_n\left[\frac{1}{2};\sqrt{2}\right].$$

О доказательствах

- Доказали ли мы оценку на время работы алгоритма (по модулю фактов теории чисел, конечно)?

О доказательствах

- Доказали ли мы оценку на время работы алгоритма (по модулю фактов теории чисел, конечно)?
- Вовсе нет! Мы почему-то предположили, что наши числа $Q(x_i)$ — это случайные Y-гладкие числа.
- Это ниоткуда не следует. Кроме того, мы предположили, что не будем постоянно натыкаться на «неинтересные» пары (u, v) (для которых $u \equiv \pm v \pmod{n}$). Это тоже неизвестно почему.
- Но тут ничего не поделаешь.

Outline

1. Метод Крайчика
   - Введение
   - Метод Крайчика
   - Гладкие числа и оценка сложности
2. Решето квадратичное и не только
   - Решето Эратосфена и квадратичное решето
   - Решение линейной системы

Как улучшить метод Крайчика

- Сложность метода Крайчика была равна

$$\frac{\pi^2(Y)X}{\psi(X,Y)}.$$

- Что мы здесь в принципе можем улучшить, а что — нет?

Как улучшить метод Крайчика

- Сложность метода Крайчика была равна

$$\frac{\pi^2(Y)X}{\psi(X,Y)}.$$

- Что мы здесь в принципе можем улучшить, а что — нет?
- Вероятность того, что попадётся гладкое число, нам неподконтрольна, это факт жизни.
- А вот $\pi^2(Y)$ — это оценка, которой мы оценили сложность проверки $\pi(Y)$ чисел на Y-гладкость.
- И это не математическая теорема, а оценка сложности очевидного метода. Который можно улучшить.

Решето Эратосфена

- Решето Эратосфена: чтобы найти простые числа от 2 до n, нужно выписать все числа:

$$2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, \ldots,$$

а потом вычёркивать те, которые делятся на $2, 3, 5, 7, \ldots$:

$$\begin{array}{l}
2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 \ldots \\
2, 3, \cancel{4}, 5, \cancel{6}, 7, \cancel{8}, 9, \cancel{10}, 11, \cancel{12}, 13, \cancel{14}, 15 \ldots \\
2, 3, \cancel{4}, 5, \cancel{6}, 7, \cancel{8}, \cancel{9}, \cancel{10}, 11, \cancel{12}, 13, \cancel{14}, \cancel{15} \ldots \\
\ldots
\end{array}$$

- А как таким же способом найти Y-гладкие числа?

Решето Эратосфена

- Точно так же, только теперь мы не просто вычёркиваем числа, а заменяем их на результат деления, и интересуют нас те числа, которые превратятся в 1.
- Кроме того, чтобы отследить степени, нужно делить на простые числа несколько раз, при каждой степени.

Решето Эратосфена

- Например, найдём 5-гладкие числа:

сначала :	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	..
на 2 :	1	1	3	2	5	3	7	4	9	5	11	6	13	7	15	..
на 3 :	1	1	1	2	5	1	7	4	3	5	11	2	13	7	5	..
на 4 :	1	1	1	1	5	1	7	2	3	5	11	1	13	7	5	..
на 5 :	1	1	1	1	1	1	7	2	3	1	11	1	13	7	1	..
на 8 :	1	1	1	1	1	1	7	1	3	1	11	1	13	7	1	..
на 9 :	1	1	1	1	1	1	7	1	1	1	11	1	13	7	1	..
5-гладкие :	1	2	3	4	5	6		8	9	10		12			15	..

Решето Эратосфена: упрощения

- Как быстро-быстро делить на заранее заданные числа?

Решето Эратосфена: упрощения

- Как быстро-быстро делить на заранее заданные числа?
- Большинство алгоритмов делают так: считают очень грубый $\log$ от всех чисел (например, количество битов).
- Потом вместо деления можно вычитать $\log 2$, $\log 3$ и т.д. (тоже грубые логарифмы).
- У кого в конце концов получится около нуля, тот и гладкий.
- Кроме того, обычно не проверяют степени простых чисел — на этом мы потеряем некоторые гладкие числа, но не слишком много.

Решето Эратосфена: сложность

- Такая проверка на гладкость очень быстро происходит. Нам нужно для каждого простого числа $p < Y$ сделать $\frac{N}{p}$ операций. На это есть теорема Мертенса:

$$\sum_{p<Y} \frac{1}{p} = \ln \ln Y + \gamma + O\left(\frac{1}{\log Y}\right), \text{ где } \gamma \approx 0,261497\ldots$$

- Иначе говоря, мы проверим N чисел на Y-гладкость за время $O(N \log \log Y)$.
- Но у нас не все числа, а очень специальные: $Q(x) = x^2 - n$. Что делать?

Квадратичное решето

- А то же самое. Рассмотрим последовательность $Q(x) = x^2 - n$ для $x = x_0 = \lceil \sqrt{n} \rceil, x_0 + 1, \ldots$.
- Для каких значений x $Q(x)$ будет делиться на p?

Квадратичное решето

- А то же самое. Рассмотрим последовательность $Q(x) = x^2 - n$ для $x = x_0 = \lceil\sqrt{n}\rceil, x_0 + 1, \ldots$.
- Для каких значений x $Q(x)$ будет делиться на p?
  - Если n — квадрат по модулю p, то $x^2 - n \equiv 0 \pmod{n}$ iff $x \equiv a$ или $b \pmod{p}$, где a и b — корни из n по модулю p.
  - Если n — не квадрат $\pmod{p}$, то делиться никогда не будет.
- Значит, можно просто так же вычёркивать те $Q(x)$, для которых x делится на a или b.
- То же самое, конечно, работает для любого другого многочлена степени 2.

Оценка сложности

- В итоге получилось, что вместо $\pi^2(Y)$ в числителе оценки стало $\pi(Y)\log\log X$. Сама оценка сложности:

$$\frac{\pi(Y)X\log\log X}{\psi(X,Y)};$$

аналогичные соображения насчёт $Y = X^{1/u}$ приводят на этот раз к

$$u = \sqrt{\frac{(2+o(1))\log X}{\log\log X}}, \qquad Y = e^{\sqrt{\left(\frac{1}{2}+o(1)\right)\log X\log\log X}},$$

и оценка сложности для $X = n^{\frac{1}{2}+o(1)}$ получается

$$L_n\left[\frac{1}{2};1\right] = e^{(1+o(1))\sqrt{\log n\log\log n}}.$$

- Мы сэкономили: вместо $L_n\left[\frac{1}{2};\sqrt{2}\right]$ получили $L_n\left[\frac{1}{2};1\right]$.

О решении линейной системы

- Мы тут всё время забываем о решении линейной системы.
- Но она тоже занимает время. Какое? У нас $\pi(Y) \approx Y/\log Y$ уравнений и неизвестных.
- Наивный метод Гаусса работает $O(n^3)$; это для нашего $Y = e^{\sqrt{\left(\frac{1}{2}+o(1)\right)\log X \log\log X}}$ даст $e^{\left(\frac{3}{2}+o(1)\right)\sqrt{\log n \log\log n}}$, т.е. больше, чем $L_n\left[\frac{1}{2};1\right]$.
- Значит, нужны другие методы.

Метод Гаусса

- Вариант метода Гаусса всё равно применяется на первом этапе, для упрощения; нужно:
  1. удалить все столбцы с ≤ 1 ненулевым элементом и все строки, в которых эти ненулевые элементы были;
  2. пометить столбцы как «тяжёлые» или «лёгкие», в зависимости от к-ва ненулевых к-нтов;
  3. для каждой строки, у которой только один ненулевой к-нт ± 1 в лёгком столбце, повычитать её из других строк с к-нтами в этом столбце, обнулив все остальные строки;
  4. повторить, пока матрица не станет достаточно маленькой.
- При этом матрица всё равно останется разреженной, и с ней можно работать более быстрыми алгоритмами.

Алгоритмы для разреженных матриц

- Алгоритм Ланцоша (Lanczos) — почти не использует дополнительной памяти, кроме матрицы, доказательства верхней оценки нет, но на практике всегда $O(n^2)$.
- Алгоритм Видеманна (Wiedemann) — тоже $O(n^2)$; мы алгоритм рассмотрим.
- Копперсмит (Coppersmith): блочный алгоритм Видеманна.

Алгоритм Видеманна

- Задача: найти такой вектор $\mathbf{w}$, что $A\mathbf{w} = \mathbf{0}$.
- Рассмотрим случайные векторы $\mathbf{x}$ и $\mathbf{z}$, а также $\mathbf{y} = A\mathbf{z}$. Рассмотрим

$$\mathbf{x}^\top \mathbf{y}, \mathbf{x}^\top A\mathbf{y}, \mathbf{x}^\top A^2\mathbf{y}, \mathbf{x}^\top A^3\mathbf{y}, \ldots$$

Минимальный многочлен

- Вспомним линейную алгебру: у матрицы A размера $n \times n$ есть минимальный многочлен p степени $n_0 \leq n$, для которого $p(A) = 0$.
- Пусть минимальный многочлен p: $\sum_{i=0}^{n_0} p_i A^i = 0$. Значит,

$$\sum_{i=0}^{n_0} p_i \mathbf{x}^\top A^i \mathbf{y} = 0,$$

и этот многочлен также порождает и нашу последовательность.
- Но когда мы изучали поточные шифры, у нас был алгоритм Берлекампа-Месси как раз для того, чтобы находить порождающие многочлены!

Алгоритм Видеманна

- Итак, мы применяем алгоритм Берлекампа-Месси и получаем такие коэффициенты q_i, что

$$\sum_{i=0}^{n_0} q_i \mathbf{x}^\top A^i \mathbf{y} = 0.$$

- Мы надеемся, что при этом заодно и

$$\sum_{i=0}^{n_0} q_i A^i \mathbf{y} = 0, \text{ и, т.к. } \mathbf{y} = A\mathbf{z}, \quad M\left(\sum_{i=0}^{n_0} q_i A^i \mathbf{z}\right) = 0,$$

  и мы надеемся, что $\mathbf{w} = \sum_{i=0}^{n_0} q_i A^i \mathbf{z} \neq 0$, ведь тогда это и есть решение.

- Наши надежды часто (по $\mathbf{x}$ и $\mathbf{y}$) будут оправдываться.

Итоги

- Как бы то ни было, мы считаем, что наши алгоритмы решения систем работают за $O(n^2)$.
- И для квадратичного решета при $Y = e^{\sqrt{\left(\frac{1}{2}+o(1)\right)\log X \log\log X}} = e^{\left(\frac{1}{2}+o(1)\right)\sqrt{\log n \log\log n}}$ получается как раз $e^{\sqrt{(1+o(1))\log n \log\log n}}$, т.е. шаги решения системы и её построения эквивалентны по сложности.
- На практике обычно делают систему поменьше, т.к. решето очень легко распараллелить, а решение системы — никак.

Спасибо за внимание!

- Lecture notes и слайды будут появляться на моей homepage:
  http://logic.pdmi.ras.ru/~sergey/
- Присылайте любые замечания, решения упражнений, новые численные примеры и прочее по адресам: sergey@logic.pdmi.ras.ru, snikolenko@gmail.com.