Осень 2019

Лекция №8

Лектор: Елена Киршанова *Оформил Филипп Максимов*

1 Тест на простоту Миллера—Рабина

$a^{p-1} \equiv 1 \bmod p,\ p$ — простое (Теорема Ферма)

$p - 1 = 2^k \cdot q,\ q$ — нечётное. Тогда для a, т.ч. $p \nmid a$, либо:

1) $a^q \equiv 1 \bmod p$, либо
2) одно из чисел $a^q, a^{2q}, ..., a^{2^{k-1}q} \equiv -1 \bmod p$

$a^q, a^{2q}...a^{2^k q} \equiv 1 \bmod p$, все предыдущие числа в списке — квадраты друг друга. Тогда либо первое число в списке $a^q \equiv 1 \bmod p$ (и все остальные $\equiv 1 \bmod p$), либо найдётся b в списке, т.ч. $b \not\equiv 1$ и $b^2 \equiv 1 \bmod p$, т.е. $b \equiv -1 \bmod p$.

Если $\exists\ a$, т.ч. $\gcd(a, n) = 1$ и оба условия

$$\begin{cases} a^q \not\equiv 1 \bmod n \\ a^{a^i q} \equiv 1 \bmod n\ \forall i = 0...k-1 \end{cases}$$

выполняются, то a — *свидетель*, что n — составное.

Algorithm 1 Miller—Rabin

Input: n, a

```
n − 1 = 2^k q, q — нечётное
a := a^q mod n
if a ≡ 1 mod n then
    return ⊥
for i = 0 . . . k − 1 do
    if a ≡ 1 mod n then
        return ⊥
    a := a^2 mod n
return «n — составное»
```

Алгоритм повторяется k раз для случайно выбранных $a \in [2..n-2]$

Время работы $\mathcal{O}(k \log^3 n)$, если используется быстрое возведение в степень.

Вероятность ошибки (вернуть $\perp$ для составного n): 2^{-2k}

2 Тест на простоту, основанный на эллиптических кривых

Задача: По данному (большому) числу p определить, является ли p простым числом и, если да, вывести доказательство (*сертификат*) простоты p.

Самый быстрый на сегодняшний день вероятностный алгоритм предложен S. Goldwasser,J. Killan «Primality testing using elliptic curves» в 1986. С последующими улучшениями, он работает за время $\operatorname{poly}\log p$, проверка сертификата простоты: $\mathcal{O}(\log^4 p)$

Детерминированные алгоритмы (Cohen, Lenstra «Primality testing & Jacobi Sums» 1984) работают за квази-полиномиальные от $\log p$ время $(\log p)^{\mathcal{O}(\log\log p)}$

Т.е. детерминированные алгоритмы пригодны для небольших чисел p.

2.1 Предварительные сведения

Теорема 1 (О распределении порядков случайных эллиптических кривых)**.** *Пусть $p > 5$ — простое. $S \subseteq \{p+1-\lfloor\sqrt{p}\rfloor, p+1+\lfloor\sqrt{p}\rfloor\}$. Пусть далее $A, B \leftarrow^{\$} \mathbb{F}_p$. Тогда $\exists\, c$ — константа ($c \in \Theta(1)$), т.ч.*

$$\Pr\left[\#E_{A,B}(\mathbb{F}_p) \in S\right] > \frac{k}{\log p} \cdot \frac{|S|-2}{2\lfloor\sqrt{p}\rfloor+1},$$

где $\#E_{A,B}(\mathbb{F}_p)$ — число $\mathbb{F}_p$-рациональных точек на кривой $E_{A,B} : y = x^3 + Ax + B$

Неформальная интерпретация теоремы: число точек $E_{A,B}$ ведёт себя как случайное число из интервала $\left[p+1-\lfloor\sqrt{p}\rfloor, p+1+\lfloor\sqrt{p}\rfloor\right]$

Лемма 2. *Пусть $n \in \mathbb{Z},\ 2,3 \nmid n;\ p > 3$ — простой делитель n и $4A^3+27B^2 \not\equiv 0 \bmod p$.*

Для любого $x \in \mathbb{Z}/n\mathbb{Z}$ зададим $x_p := x \bmod p$ и для любой точки $L = (x,y) \in E_{A,B}(\mathbb{Z}/n\mathbb{Z})$ зададим $L_p = (x_p, y_p) \in E_{A,B}(\mathbb{F}_p)$, $\infty_p = \infty_x \in E_{A,B}(\mathbb{Z}/n\mathbb{Z})$.

Тогда $\forall L, M \in E_{A,B}(\mathbb{Z}/n\mathbb{Z})$, если $L+M$ определено, то $(L+M)_p = L_p + M_p$

Доказательство. Проверить формулы сложения для $E_{A,B}(\mathbb{Z}/n\mathbb{Z})$, см. лекцию №7. □

Теорема 3 (Критерий простоты)**.** *Пусть $n \in \mathbb{Z},\ 2,3 \nmid n$. Пусть далее $A, B \in \mathbb{Z}/n\mathbb{Z}$ т.ч. $\gcd(4A^3, 27B^2, n) = 1$ и $L \neq \infty$ на $E_{A,B}(\mathbb{Z}/n\mathbb{Z})$. Тогда, если существует простое $q > (n^{1/4}+1)^2$, т.ч. $qL = \infty$, то n — простое.*

Доказательство. От противного: пусть n — составное $\Rightarrow \exists p > 3$, т.ч. $p|n$ и $p \leq \sqrt{n}$.

Заметим, $\gcd(4A^3, 27B^2, p) \neq 0 \bmod p$ (иначе мы бы получили противоречие с условием $\gcd(4A^3, 27B^2, n) = 1$.

Тогда по Лемме 2: $L_p \in E_{A,B}(\mathbb{F}_p)$ и $q \cdot L_p = (qL)_p = \infty_p = \infty \Rightarrow \mathrm{ord}(L_p)$ должен делить q. По Теореме Хассе, $\mathrm{ord}(L_p) \leq \#E_{A,B}(\mathbb{F}_p) \leq (\sqrt{p}+1)^2 \leq (n^{1/4}+1)^2 < q$. Это противоречие, значит, n — простое. □

2.2 Алгоритм: тест на простоту

Идея: Сведём доказательство простоты p к доказательству простоты $q \leq \frac{p}{2} + o(P)$, рекурсивно применим алгоритм к q, пока не получим достаточно малое значение q — такое, что детерминированные тесты будут эффективны.

Для заданного p, построим кривую $E_{A,B}$ над p с точкой $L \in E_{A,B}(\mathbb{F}_P)$ порядка $q \approx p/2$

Условия генерации A, B:

a) $(4A^3 + 27B^2, p) = 1$

b) $\#E_{A,B}(\mathbb{F}_p) \in \left[p + 1 - \lfloor\sqrt{p}\rfloor, p + 1 + \lfloor\sqrt{p}\rfloor\right]$
Использует эффективные алгоритмы подсчёта точек на кривой, например, алгоритм Схофа

c) $\#E_{A,B}(\mathbb{F}_p)$ – чётно

Algorithm 2 gen_curve

Input: p
Output: A, B, q

1: $A, B \xleftarrow{\$} \mathbb{F}_p$ по условиям выше
2: $q = \#E_{A,B}(\mathbb{F}_p)/2$
3: **if** $2 \mid q$ или $3 \mid q$ **then**
4: Вернуться к шагу 1
5: Запустить вероятностный алгоритм проверки q на простоту (Миллера—Рабина) на $\mathcal{O}(\log p)$ шагов (т.е. чтобы вероятность ошибки была $\sim 2^{-\log p}$).

Algorithm 3 find_point

Input: p, q, A, B

1: $x \xleftarrow{\$} \mathbb{F}_p$, что $x^3 + Ax + B$ — квадрат в $\mathbb{F}_p$
2: $y \xleftarrow{\$} \left\{\pm\sqrt{x^3 + Ax + B}\right\}; L := (x, y)$
3: **if** $q \cdot L \neq \infty$ **then**
4: вернуться к шагу 1.
5: **return** L

LB — число бит в числе такое, что детерминированные алгоритмы простоты эффективны для этого числа.

Algorithm 4 prove_prime

Input: p

1: $i = 0, p_0 = p$
2: **while** $p_i > 2^{LB}$ **do**
3: $(A_i, B_i), p_{i+1} \leftarrow$ gen_curve(p)
4: $L_i \leftarrow$ find_point(p_i, p_{i+1}, A, B)
5: $i := i + 1$
6: **if** $i \geq (\log p)^{\log\log p}$ или $2 \mid p_i$ или $3 \mid p_i$ **then**
7: Вернуться к шагу 1
8: Проверить p_i на простоту детерминированным алгоритмом (*В лабе можно использовать встроенную функцию* `is_prime()`)
9: **if** p_i не доказано простым **then**
10: Вернуться к шагу 1
11: **return** $C = ((A_0, B_0), L_0, p_1, ..., (A_{i-1}, B_{i-1}), L_{i-1}, p_{i-1})$

2.2.1 Корректность

- p — простое. Тогда выход C — сертификат: 'свидетельство' простоты p. На шагах 3, 4 мы получаем кривую E_{A_i,B_i} и точку L_i порядка p_{i+1}, удовлетворяющие условиям Теоремы 3.
- p — составное. Тогда получим делители p на шаге 4 (или раньше). $(2, 3 \nmid p)$ алгоритма find_point(), аналогично алгоритму факторизации.

2.2.2 Корректность

Alg. 2 Самый затратный шаг — подсчёт $\#E_{A,B}(\mathbb{F}_p)$ (условия генерации, пункт b).
Алгоритм Схофа: $\widetilde{\mathcal{O}}(\log^8 p)$. Вероятность, что $\#E_{A,B}(\mathbb{F}_p)$ лежит в нужном интервале — Теорема 1

Alg. 3 Самые затратные шаги:

Шаг 1: $x \xleftarrow{\$} \mathbb{F}_p$ — кв. вычет с вероятностью $\mathcal{O}(1)$.

Шаг 4: быстрое умножение на q:
$\mathcal{O}(\log q \cdot \log^2 p) = \mathcal{O}(\log^3 p)$

Alg. 3 В каждой Итерации шага 2, p_i уменьшается на 2, т.е. ожидаем $\mathcal{O}(\log p)$ итераций.
Доминирующий шаг: условия генерации для gen_curve() — (b)
$\Rightarrow$ общее время работы: $\mathcal{O}(\log^9 p)$

Количество кривых $E_{A,B}$, не удовлетворяющих свойствам (a)—(c) условий генерации для gen_curve() $= \mathcal{O}(\log^3 p)$ (эвристика)

2.3 Проверка сертификата

Algorithm 5 check_prime

Input: $p_0, C = ((A_0, B_0), L_0, p_1, ..., (A_{i-1}, B_{i-1}), L_{i-1}, p_{i-1})$
Output: {Reject, Accept}

1: **for** $j = 0$ ... $i - 1$ **do**
2: assert $(2 \nmid p_j)$ *(a)*
3: assert $(3 \nmid p_j)$ *(b)*
4: assert $(4A_j^3 + 27B_j^2, 1) = 1$ *(c)*
5: assert $(P_{j+1} > (p_j^{1/4} + 1)^2)$ *(d)*
6: assert $L_j \neq \infty$ *(e)*
7: assert $p_{j+1}L_j = \infty$*(f)*
8: **return** Accept

2.3.1 Корректность

Если check_prime() возвращает Accept, $\Rightarrow p_i$ — простое $\Rightarrow p_{i-1}$ простое по Теореме 3 ($\Rightarrow$ ... $\Rightarrow p_0$ — простое)

Условия (a),(b) проверяются на шаге 6 алгоритма 4

(c) — шаг (a) условия генерации

(d) — Теорема Хассе: $\#E_{A,B}(\mathbb{F}_{p_j}) \geq (\sqrt{p_j} - 1)^2 \Rightarrow$

$$p_{j+1} = \frac{\#E(\mathbb{F}_{p_j}}{2} \geq \frac{(\sqrt{p_j} - 1)^2}{2} > (p_j^{1/4} + 1)^2 \quad \forall p_j > 37$$

Для столь малых p_j проверка на простоту тривиальна.

(e), (f) проверяются в find_point, шаг 3.

2.3.2 Время работы

Проверка каждого $p_j : \mathcal{O}(\log^3 p)$ — шаг (f) самый затратный.

Всего: $\mathcal{O}(\log p)$ различных p_j в сертификате $C \Rightarrow \mathcal{O}(\log^4 p)$