

AKS алгоритм проверки числа на простоту

Рубаненко Евгений

2017

Аннотация

В данной работе рассматривается тест Агравала - Каяла - Саксены проверки числа на простоту. Алгоритм работает за полиномиальное время. Приведено доказательство корректности и сравнение с другими алгоритмами проверки числа на простоту.

1 Введение

Долгое время считалось, что изучение простых чисел - пример "чистой" математики. Но в 70-ых годах XX века выяснилось, что простые числа могут быть использованы при создании криптографических алгоритмов. Это послужило толчком в развитии данной области. Для поиска простых чисел существует множество алгоритмов: простых и сложных. Но только в 2002 году был предложен алгоритм, который ответил на вопрос принадлежности задачи распознавания простоты классу P. Основное свойство теста AKS заключается в том, что он одновременно универсален (то есть может использоваться для проверки простоты любых чисел), полиномиален, детерминирован (что гарантирует получение уникального предопределенного результата) и безусловен (то есть корректность алгоритма не зависит от каких-либо недоказанных гипотез), предыдущие алгоритмы обладали лишь тремя из этих четырех свойств.

2 Идея

Идея алгоритма основана на следующей лемме.

Лемма 2.1. Пусть $a \in \mathbb{Z}$, $n \in \mathbb{N}$ и $(a, n) = 1$. Тогда n простое тогда и только тогда, когда

$$(X + a)^n = X^n + a \pmod{n} \quad (1)$$

Доказательство Леммы 2.1. Посмотрим на коэффициент перед X^i , $i \in \{0, \dots, n-1\}$ в многочлене $((X + a)^n - (X^n + a))$. Он равен $\binom{n}{i}a^{n-i}$. Тогда, если n простое, то $\binom{n}{i} = 0 \pmod{n}$ и сравнение (1) верно. Если n составное, то обозначим q - простой делитель n , входящий в его разложение на простые в степени k . Тогда $q^k \nmid \binom{n}{q}$ и $(q, a^{n-q}) = 1$, откуда получаем, что коэффициент при X^q не равен нулю. Но тогда многочлен $((X + a)^n - (X^n + a))$ не равен тождественно нулю, что завершает доказательство леммы. \square

Тогда можно придумать следующий тривиальный алгоритм: выбрать a и проверить (1). Проблема заключается в том, что он не эффективен - в худшем случае придется вычислить n коэффициентов в левой части (1).

Идея теста Агравала - Каяла - Саксены заключается в том, чтобы проверять следующее соотношение

$$(X + a)^n = X^n + a \pmod{X^r - 1, n}, \quad (2)$$

где r - специально подобранное число. Теперь проблема заключается в том, что соотношению (2) могут удовлетворять не только простые n . Далее будет показано, что можно проверить дополнительные условия, из которых будет следовать, что n простое.

3 Используемые обозначения

Большинство используемых обозначений являются общеизвестными. Дополнительную информацию можно найти в [1].

В работе используется символ $O^\sim(t(n))$, что есть $O(t(n) \cdot \text{poly}(\log t(n)))$. Через $\text{НОК}(m)$ обозначен $\text{НОК}(1, 2, \dots, m)$.

4 Алгоритм

Data: n : integer

Result: True, если n простое, False - иначе

if $n = a^b$, где $a \in \mathbb{N}$, $b > 1$ **then**

 return False;

else

$r := \min\{r \mid o_r(n) > \log^2 n\}$;

if $1 < (a, n) < n$, для какого-то $a \leq r$ **then**

 return False;

else

if $n \leq r$ **then**

 return True;

else

for $a := 1$ to $\lfloor \sqrt{\phi(r)} \log n \rfloor$ **do**

if $((X + a)^n \neq X^n + a \pmod{X^r - 1, n})$ **then**

 return False;

end

end

 return True;

end

end

end

Algorithm 1: AKS алгоритм

Дальше будут часто встречаться упоминания конкретных шагов алгоритма, поэтому введем следующую нумерацию:

1. Первый шаг - If $n = a^b$, где $a \in \mathbb{N}$, $b > 1$ return False
2. Второй шаг - $r := \min\{r \mid o_r(n) > \log^2 n\}$
3. Третий шаг - If $1 < (a, n) < n$, для какого-то $a \leq r$ return False
4. Четвертый шаг - If $n \leq r$ return True
5. Пятый шаг - Цикл for
6. Шестой шаг - return True

5 Доказательство корректности

Лемма 5.1. $\text{НОК}(m) \geq 2^m$ при $m \geq 9$.

Доказательство Леммы 5.1. Рассмотрим при $m \in \{1, \dots, n\}$ интеграл

$$\begin{aligned} S(n, m) &= \int_0^1 x^{m-1} (1-x)^{n-m} dx = \int_0^1 x^{m-1} \sum_{k=0}^{n-m} \binom{n-m}{k} (-1)^k x^k dx \\ &= \sum_{k=0}^{n-m} \binom{n-m}{k} (-1)^k \int_0^1 x^{m-1+k} dx = \sum_{k=0}^{n-m} \binom{n-m}{k} (-1)^k \frac{1}{m+k} \end{aligned}$$

$\text{НОК}(n)$ делится на $m + k$ при всех $k \in \{0, \dots, n - m\} \implies \text{НОК}(n) \cdot S(n, m) \in \mathbb{N}$. С другой стороны

$$\begin{aligned} S(n, m) &= \frac{1}{m} \int_0^1 (1-x)^{n-m} dx^m = \frac{1}{m} \int_0^1 x^m (n-m)(1-x)^{n-m-1} dx \\ &= \frac{n-m}{m} \int_0^1 x^m (1-x)^{n-m-1} dx = \frac{(n-m)(n-m-1)}{m(m+1)} \int_0^1 x^{m+1} (1-x)^{n-m-2} dx \\ &= \dots = \frac{(n-m)(n-m-1) \dots 1}{m(m+1) \dots (n-1)} \int_0^1 x^{n-1} dx = \frac{(n-m)!}{m(m+1) \dots (n-1)n} \\ &= \frac{(n-m)!(m-1)!}{n!} = \frac{1}{m \binom{n}{m}} \end{aligned}$$

Так как число $\text{НОК}(2n) \cdot S(2n, n) = \frac{\text{НОК}(2n)}{n \binom{2n}{n}}$ целое, то $\text{НОК}(2n)$ делится на $n \binom{2n}{n}$, и $n \binom{2n}{n}$ делит $\text{НОК}(2n+1)$. Аналогичным образом $(n+1) \binom{2n+1}{n+1}$ делит $\text{НОК}(2n+1)$. Отметим, что $(2n+1) \binom{2n}{n} = (n+1) \binom{2n+1}{n+1}$. Учитывая, что $(2n+1, n) = 1$, получаем, что $\text{НОК}(2n+1)$ делится на $n(2n+1) \binom{2n}{n}$, откуда $\text{НОК}(2n+1) \geq n(2n+1) \binom{2n}{n}$.

Так как $\binom{2n}{n}$ наибольшее из $2n+1$ слагаемых биномиального разложения $(1+1)^{2n}$, то $(2n+1) \binom{2n}{n} \geq 4^n$. Отсюда при $n \geq 2$ имеем $\text{НОК}(2n+1) \geq 2^{2n+1}$. Кроме этого, $\text{НОК}(2n+2) \geq \text{НОК}(2n+1) \geq n \cdot 4^n$, так что при $n \geq 4$ выполняется $\text{НОК}(2n+2) \geq 2^{2n+2}$. Таким образом, при $n \geq 9$ имеет место неравенство $\text{НОК}(n) \geq 2^n$ \square

Лемма 5.2. Существует $r \leq \max\{3, \lceil \log^5 n \rceil\}$ такое, что $o_r(n) > \log^2 n$.

Доказательство Леммы 5.2. При $n = 2 \exists r = 3$, и утверждение верно. Будем считать, что $n > 2$. Обозначим $B = \lceil \log^5 n \rceil$ и рассмотрим произведение

$$P = n \cdot \prod_{i=1}^{\lfloor \log^2 n \rfloor} (n^i - 1)$$

Оценим P сверху

$$P < n \cdot \prod_{i=1}^{\lfloor \log^2 n \rfloor} n^i = \prod_{i=2}^{\lfloor \log^2 n \rfloor + 1} n^i < n^{\frac{1}{2} \log^2 n \cdot (\log^2 n + 3)} \leq n^{\log^4 n} \leq 2^{\log^5 n} \leq 2^B$$

Так как $B = \lceil \log^5 n \rceil > 10$, то можно воспользоваться Леммой 5.1. Значит, $P < \text{НОК}(B)$, так что среди чисел от 1 до B есть число s , на которое P не делится. Если $(s, n) = 1$, то положим $r = s$. Если же $(s, n) > 1$, то рассмотрим $r = \frac{s}{(s, n)}$. Выбранное таким образом r тоже удовлетворяет условию $r \nmid P$ ($n \mid P$, $s \nmid P$, $(n, s) \mid P \implies r \nmid P$), причем $(r, n) = 1$. Так как $r \nmid n^i - 1$, $1 \leq i \leq \lfloor \log^2 n \rfloor$, то $o_r(n) > \log^2 n$. \square

Определение 5.1. Пусть r - некоторое, а p - простое числа. Назовем число $m \in \mathbb{N}$ особым по отношению к многочлену $f(X)$, если

$$f^m(X) = f(X^m) \pmod{X^r - 1, p}$$

Лемма 5.3. Если m и m' являются особыми для многочлена $f(X)$, то $m \cdot m'$ также является особым по отношению к $f(X)$.

Доказательство Леммы 5.3. Так как m является особым для $f(X)$, имеем

$$f^{m \cdot m'}(X) = f^{m'}(X^m) \pmod{X^r - 1, p}$$

Так как m' является особым для $f(X)$, то заменяя X на X^m в определении, получим

$$f^{m'}(X^m) = f(X^{m \cdot m'}) \pmod{X^{m \cdot r} - 1, p} = f(X^{m \cdot m'}) \pmod{X^r - 1, p}$$

где последнее равенство получено исходя из того, что $X^r - 1 \mid X^{m \cdot r} - 1$. Объединяя, получаем

$$f^{m \cdot m'}(X) = f(X^{m \cdot m'}) \pmod{X^r - 1, p}$$

□

Лемма 5.4. Если m является особым для многочленов $f(X)$ и $g(X)$, то оно также является особым для многочлена $f(X) \cdot g(X)$.

Доказательство Леммы 5.4.

$$(f(X) \cdot g(X))^m = f^m(X) \cdot g^m(X) = f(X^m) \cdot g(X^m) \pmod{X^r - 1, p}$$

□

Теорема 5.1. Если n простое, то алгоритм возвращает *True*.

Доказательство Теоремы 5.1. Если n простое, то на первом и третьем шагах алгоритм не может вернуть *False*. Согласно Лемме 2.1. на пятом шаге алгоритм тоже не может вернуть *False*. Тогда алгоритм вернет *True*, и это произойдет либо на 4, либо на 6 шаге. □

Теорема 5.2. Если алгоритм вернул *True*, то n простое.

Если алгоритм вернул *True* на четвертом шаге, то n обязано быть простым, иначе бы на третьем шаге был бы найден простой делитель n . Остается рассмотреть случай, когда алгоритм возвращает *True* на шестом шаге.

Обозначим r - число, найденное на втором шаге, $l = \lfloor \sqrt{\phi(r)} \log n \rfloor$.

Так как $o_r(n) > 1$, то существует простое p такое, что $p \mid n$ и $o_r(p) > 1$. Понятно, что $p > r$ и $(n, r) = 1$, потому что иначе простота n выяснилась бы или на третьем, или на четвертом шаге. Отсюда следует, что $p, n \in \mathbb{Z}_r^*$. Зафиксируем p, r, l .

На пятом шаге алгоритма проверяется l сравнений. Так как алгоритм выводит *True* на шестом шаге, то все сравнения являются верными, то есть

$$(X + a)^n = X^n + a \pmod{X^r - 1, n}, \quad 0 \leq a \leq l$$

Отсюда следует, что

$$(X + a)^n = X^n + a \pmod{X^r - 1, p}, \quad 0 \leq a \leq l$$

Согласно лемме 2.1. получаем

$$(X + a)^p = X^p + a \pmod{X^r - 1, p}, \quad 0 \leq a \leq l$$

Из последних двух соотношений следует, что

$$(X + a)^{\frac{n}{p}} = X^{\frac{n}{p}} + a \pmod{X^r - 1, n}, \quad 0 \leq a \leq l$$

то есть оба p и $\frac{n}{p}$ являются особыми для $f(X)$. Из Лемм 5.3. и 5.4. следует, что любое число из множества $I = \{(\frac{n}{p})^i \cdot p^j \mid i, j \geq 0\}$ является особым для любого многочлена из множества $P = \{\prod_{a=0}^l (X + a)^{e_a} \mid e_a \geq 0\}$.

Определим теперь два конечных множества

$$G = \{m \pmod{r}, m \in I\}$$

$$\mathcal{G} = \{g(X) \pmod{h(X)}, p, g(X) \in P\}$$

где $h(X)$ - один из неприводимых многочленов степени $o_r(p) > 1$, на которые раскладывается круговой многочлен $Q_r(X)$ над полем F_p (известно, что $Q_r(X) \mid X^r - 1$).

Введем следующее обозначение: $F = F_p(X)/(h(X))$. Тогда F является полем (потому что $h(X)$ неприводим над F_p). Обозначим $|G| = t$. Так как $o_r(n) > \log^2 n$, то $t > \log^2 n$. Также отметим, что раз $(n, r) = (p, r) = 1$, то G является подгруппой \mathbb{Z}_r^* .

Лемма 5.5. $|\mathcal{G}| \geq \binom{t+l}{t-1}$.

Доказательство Леммы 5.5. Напомним, $h(X) \mid Q_r(X)$, X - корень из единицы r -ой степени. Для начала покажем, что любые два различных полинома степени меньшей, чем t , переходят в различные элементы \mathcal{G} . Возьмем два произвольных полинома $f(X) \in P$ и $g(X) \in P$ и допустим противное, то есть что $f(X) = g(X)$ в поле F . Пусть $m \in I$ - некоторое число. Тогда $f^m(X) = g^m(X)$ в поле F . Исходя из того, что m является особым для обоих $f(X)$ и $g(X)$, а $h(X) \mid X^r - 1$, получаем, что

$$f(X^m) = g(X^m)$$

в поле F . Отсюда следует, что X^m является корнем многочлена $Q(Y) = f(Y) - g(Y)$ при всех $m \in G$. Так как $(m, r) = 1$ (потому что G является подгруппой в \mathbb{Z}_r^*), то каждый X^m является корнем из 1 степени r . Тогда в поле F у многочлена $Q(Y)$ найдется $|G| = t$ различных корней. В то же время степень многочлена $Q(Y)$ меньше t согласно выбору $f(X)$ и $g(X)$. Получили противоречие, а значит, $f(X) \neq g(X)$ в поле F . Заметим следующее: $i \neq j$ в F_p , так как $l = \lfloor \sqrt{\phi(r)} \log n \rfloor < \sqrt{r} \log n < r$ и $p > r$. Тогда все элементы $X, X+1, X+2, \dots, X+l$ будут различны в F . Также, учитывая, что степень $h(r)$ больше 1, получаем, что $X+a \neq 0$ для всех $0 \leq a \leq l$. Мы показали, что в \mathcal{G} существует хотя бы $l+1$ полином степени 1. Отсюда следует, что в \mathcal{G} найдется не менее $\binom{t+l}{t-1}$ различных полиномов степени меньше t . \square

Лемма 5.6. Если n не является степенью p , то $|\mathcal{G}| \leq n^{\sqrt{t}}$.

Доказательство Леммы 5.6. Рассмотрим следующее подмножество I :

$$\bar{I} = \left\{ \left(\frac{n}{p} \right)^i \cdot p^j \mid 0 \leq i, j \leq \lfloor \sqrt{t} \rfloor \right\}$$

Если n не является степенью p , то в множестве \bar{I} содержится хотя бы $(\lfloor \sqrt{t} \rfloor + 1)^2 > t$ различных элементов. Так как $|G| = t$, то найдутся два числа m_1 и m_2 , сравнимых по модулю r . Без ограничения общности будем считать, что $m_1 > m_2$. Тогда

$$X^{m_1} = X^{m_2} \pmod{X^r - 1}$$

Рассмотрим некоторый многочлен $f(X) \in P$. Тогда

$$\begin{aligned} f^{m_1}(X) &= f(X^{m_1}) \pmod{X^r - 1, p} \\ &= f(X^{m_2}) \pmod{X^r - 1, p} \\ &= f^{m_2}(X) \pmod{X^r - 1, p} \end{aligned}$$

То есть

$$f^{m_1}(X) = f^{m_2}(X)$$

в поле F . Следовательно, $f(X) \in \mathcal{G}$ и является корнем многочлена $Q(Y) = Y^{m_1} - Y^{m_2}$ в поле F . Так как многочлен $f(X)$ был выбран произвольно, то многочлен $Q(Y)$ имеет не менее $|\mathcal{G}|$ корней в F . Оценив степень многочлена $Q(Y)$ как $m_1 \leq \left(\frac{n}{p} \cdot p \right)^{\sqrt{t}} \leq n^{\sqrt{t}}$ получаем, что $|\mathcal{G}| \leq n^{\sqrt{t}}$. \square

Доказательство Теоремы 5.2. Допустим, что алгоритм вернул *True*. Согласно Лемме 5.5., для $t = |G|$ и $l = \lfloor \sqrt{\phi(r)} \log n \rfloor$ имеем

$$\mathcal{G} \geq \binom{t+l}{t-1} \geq^{(1)} \binom{l+1+\lfloor \sqrt{t} \log n \rfloor}{\lfloor \sqrt{t} \log n \rfloor} \geq^{(2)} \binom{2\lfloor \sqrt{t} \log n \rfloor + 1}{\lfloor \sqrt{t} \log n \rfloor} >^{(3)} 2^{\lfloor \sqrt{t} \log n \rfloor + 1} \geq n^{\sqrt{t}}$$

$$(1) : t > \sqrt{t} \log n; (2) : l = \lfloor \sqrt{\phi(r)} \log n \rfloor \geq \lfloor \sqrt{t} \log n \rfloor; (3) : \lfloor \sqrt{t} \log n \rfloor > \lfloor \log^2 n \rfloor \geq 1$$

Согласно Лемме 5.6., $|G| \leq n^{\sqrt{t}}$, если n не является степенью p . Тогда $n = p^k$, $k > 0$. Если $k > 1$, то алгоритм вернул бы *False* на первом шаге. Тогда $k = 0$ и $n = p$. \square

6 Анализ временной сложности алгоритма

Теорема 6.1. Алгоритм определяет простоту числа за время $O^{\sim}(\log^{\frac{21}{2}} n)$.

Лемма 6.1. Первый шаг алгоритма работает за время $O^{\sim}(\log^3 n)$.

Доказательство Леммы 6.1. На первом шаге проверяется, что $n \neq a^b$. Для этого надо перебрать $O(\log n)$ вариантов для a . Для конкретного a с помощью бинарного поиска проверяется, что не существует подходящего b . Перебор b требует $O(\log n)$ времени, а вычисление каждого числа вида a^b - $O^{\sim}(\log n)$. Тогда общая сложность первого шага составит $O^{\sim}(\log^3 n)$. \square

Лемма 6.2. Второй шаг алгоритма работает за время $O^{\sim}(\log^7 n)$.

Доказательство Леммы 6.2. На втором шаге алгоритма находится такое r , что $o_r(n) > \log^2 n$. Это можно сделать следующим образом: в цикле по r будем проверять, что $n^k \not\equiv 1 \pmod{r}$ для всех $k \leq \log^2 n$. Для конкретного r потребуется не больше $O(\log^2 n)$ умножений по модулю r , откуда сложность одной итерации - $O^{\sim}(\log^2 n \log r)$. Согласно Лемме 5.2., необходимое r найдется, причем перебрать придется всего $O(\log^5 n)$ значений. Тогда общая сложность второго шага составит $O^{\sim}(\log^7 n)$. \square

Лемма 6.3. Третий шаг алгоритма работает за время $O(\log^6 n)$.

Доказательство Леммы 6.3. Третий шаг алгоритма - цикл из r итераций. На каждой итерации вычисляется НОД двух чисел, что требует $O(\log n)$ времени. Тогда общая сложность третьего шага составит $O(r \log n) = O(\log^6 n)$. \square

Лемма 6.4. Пятый шаг алгоритма работает за время $O^{\sim}(\log^{\frac{21}{2}} n)$.

Доказательство Леммы 6.4. Пятый шаг алгоритма - цикл из $\lfloor \sqrt{\phi(r)} \log n \rfloor$ итераций. На каждой итерации полином степени r возводится в степень n (что требует $O(\log n)$ времени); его коэффициенты можно оценить как $O(\log n)$. Таким образом, каждая итерация требует $O^{\sim}(r \log^2 n)$ времени. Тогда общая сложность пятого шага составит

$$O^{\sim}(r \sqrt{\phi(r)} \log^3 n) = O^{\sim}(r^{\frac{3}{2}} \log^3 n) = O^{\sim}(\log^{\frac{21}{2}} n)$$

\square

Доказательство Теоремы 6.1. Так как четвертый шаг алгоритма выполняется за $O(\log n)$, то из Лемм 6.1. - 6.4. следует, что временная сложность алгоритма составляет $O^{\sim}(\log^{\frac{21}{2}} n)$. \square

7 Сравнение с другими алгоритмами

В данной работе было проведено сравнение трех алгоритмов:

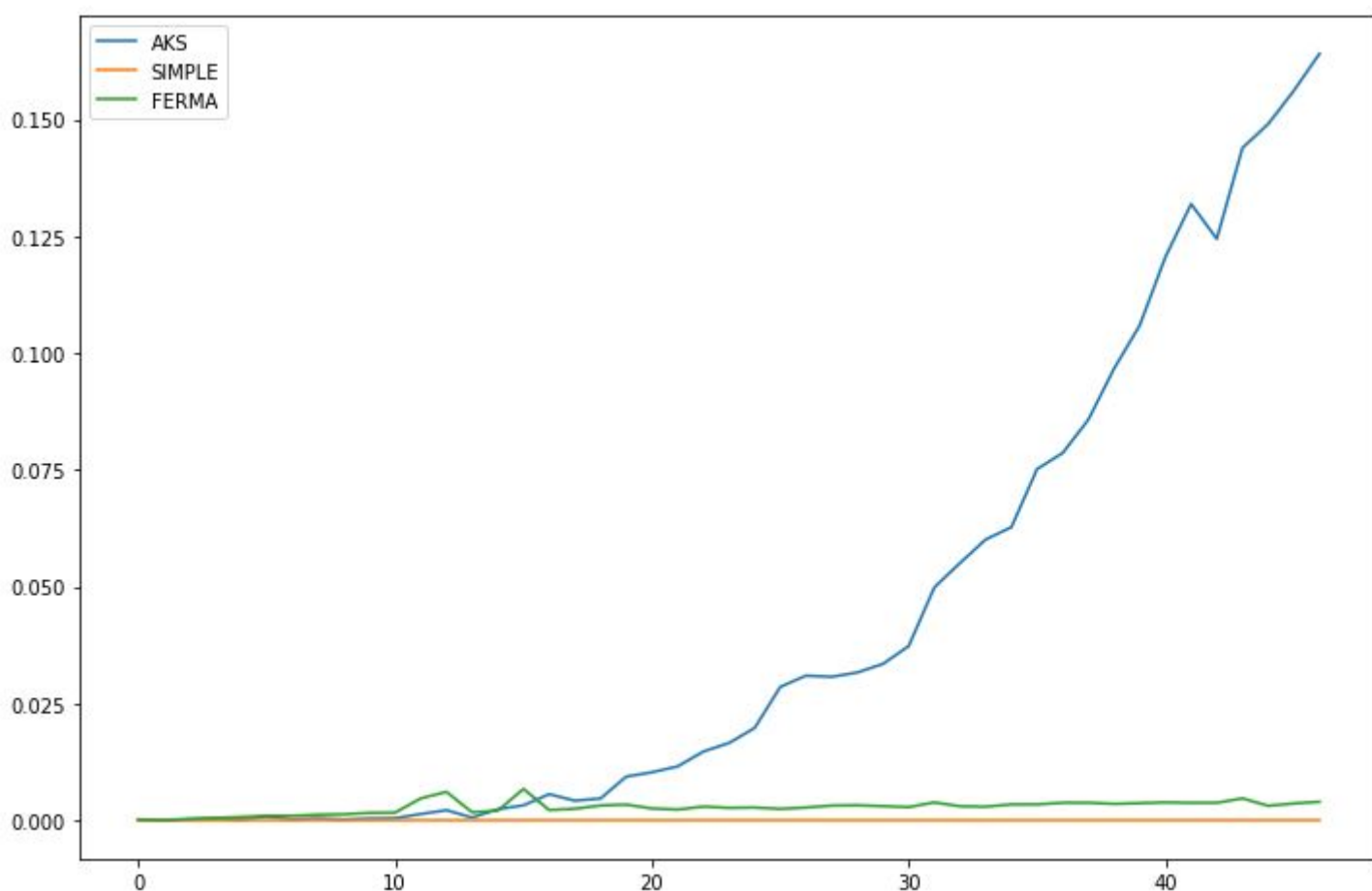
1. AKS алгоритм
2. Вероятностный тест Ферма
3. Простая проверка до корня из числа

(дополнительную информацию про тест Ферма можно найти в [3])

Сначала было проверено, что алгоритмы работают корректно - правильно проверяют простоту чисел в диапазоне $[1, 1000]$.

Затем были проведены измерения времени работы представленных алгоритмов.

Следующий график демонстрирует время работы алгоритмов (учитывалось время работы только на простых числах).



Более подробно код алгоритмов и проведенное исследование можно посмотреть в [4].

8 Источники информации

- [1] Э.Б. Винберг, Курс алгебры (2011)
- [2] Agrawal, Manindra; Kayal, Neeraj; Saxena, Nitin (2004). "PRIMES is in P"
- [3] <https://habrahabr.ru/post/205318/>
- [4] <https://github.com/svinkapeppa/5-semestr/tree/master/complexity>