Московский Физико-технический Институт
(ГУ МФТИ)
Кафедра радиотехники
http://www.re.mipt.ru/infsec

Эссе по курсу "Защита информации"

Обзор проверок на простоту

Выполнил
Студент гр.115
Кучин Борис

Долгопрудный • 2005

Введение

Данная работа ставит целью провести обзор самых известных методов проверки чисел на простоту. Проблема определения того, является ли число простым, интересна, как с чисто научной точки зрения, так как до сих пор не найдено единой аналитической записи для всех простых чисел, так и с практической точки зрения для применения в криптосистемах с открытым ключом. Следует отметить, что все алгоритмы проверки простоты делятся на две больших подгруппы: детерминированные и вероятностные проверки. Алгоритмы первой группы позволяют *точно* сказать, является число простым или составным. Алгоритмы второй группы позволяют это определить, но с некоторой вероятностью ошибки. Многократное их повторение для одного числа, но с разными параметрами, обычно позволяет сделать вероятность ошибки сколь угодно малой величиной.

Хотя простые числа изучаются уже достаточно долго, наибольшее развитие тема вероятностных проверок получила во второй половине двадцатого века именно в связи с необходимостью генерировать большие (сто и более десятичных цифр) простые числа для таких криптосистем как RSA.

Так как работа носит обзорный характер, то большинство доказательств утверждений и корректности работы тестов опущены. Кроме того, для всех современных тестов опущены их подробные описания, так как они потребовали бы введения ненужных в данной работе определений, утверждений и теорем. Из тех же соображений, не был рассмотрен вероятностный тест Соловея-Страссена (*Solovey-Strassen*).

Быстрые тесты для небольших чисел и вероятно простые числа

Небольшие простые числа

Пока не существовало необходимости генерировать большие простые числа, можно было использовать методы проверки, которые достаточно легко реализуемы без применения вычислительной техники и не требуют больших усилий для проверки маленьких чисел. Первым из таких методов является, естественно, полный перебор всех возможных делителей. Чаще всего используют модификацию такого перебора, называемую **пробным делением** (*trial division*): чтобы проверить число на простоту, делим его на все простые меньше либо равные корню из этого числа.

Если нам требуется построить все простые числа, меньшие какого-либо n, то можно воспользоваться так называемым **решетом Эратосфена**: выписываем все числа до n, а затем вычеркиваем те, которые делятся на простые числа меньше либо равные корню из n.

Малая теорема Ферма, слабо возможно простые и псевдопростые числа

В 17 веке французский математик Пьер Ферма выдвинул утверждение, которое лежит в основе практически всех возможных тестов на простоту:

Малая теорема Ферма: Если p простое и a – любое целое, то $a^p = a \pmod p$. В частности, если p не делит a, то $a^{p-1} = 1 \pmod p$.

На основании этой теоремы можно построить достаточно мощный тест на простоту:

Тест Ферма: для $n > 1$ выбираем $a > 1$ и вычисляем $a^{n-1} \bmod n$, если результат не 1, то n составное, если 1 то, n – слабо *возможно простое по основанию a* (***a*-PRP**).

Операция возведения в степень по модулю легко реализуется методом последовательного возведения в квадрат. Часть чисел проходят тест Ферма и при этом являются составными, такие числа называются **псевдопростыми**.

Для любого основания a существует бесконечно много псевдопростых чисел по основанию a. Мы можем сделать тест более точным, проведя его несколько раз для одного и того же числа, но с разными основаниями. Но даже в этом случае существуют **числа Кармайкла**

(*Carmichael*), которые проходят тест Ферма для всех чисел, не являющихся их делителями. Таким образом, мы приходим к выводу, что нам нужен более качественный способ проверки.

Тест Рабина-Миллера и сильно возможно простые числа.

Можно существенно улучшить тест Ферма, заметив, что если n – простое нечетное, то для 1 есть только два квадратных корня по модулю n: 1 и -1. Таким образом, квадратный корень из a^{n-1}, $a^{(n-1)/2}$ равен плюс или минус единице. Если $(n-1)/2$ опять нечетно, то мы сможем снова извлечь корень и так далее. Первый вариант алгоритма, предлагает использовать только одно деление:

Тест Леманна (*Lehmann*)**:** если для какого-либо целого числа a меньшего n **не** выполняется условие $a^{(n-1)/2} = \pm 1 \pmod n$, то число n – составное. Если это условие выполняется, то число n – возможно простое, причем вероятность ошибки не превышает 50%.

Этот тест можно естественным образом улучшить, если извлекать корень по модулю не один раз, а столько, сколько получится.

Тест Рабина-Миллера (*Rabin-Miller*)**:** Запишем $(n-1)$ в виде $2^s d$, где d нечетно, а s неотрицательно: n называется *сильно возможно простым по основанию a* (a-SPRP), если выполняется одно из двух условий:

1. $a^d = 1 (\mathrm{mod}\, n)$ или
2. $(a^d)^{2^r} = -1 (\mathrm{mod}\, n)$ для какого-либо неотрицательного r меньшего s.

Как и для теста Ферма, все числа $n > 1$, которые не проходят этот тест – составные, а числа, которые проходят, *могут* быть простыми. И, что важно, для этого теста нет аналогов чисел Кармайкла.

В 1980 году было доказано, что вероятность ошибки теста Рабина-Миллера не превышает 1/4. Таким образом, применяя тест Рабина-Миллера t раз для разных оснований, мы получаем вероятность ошибки 2^{-2t}.

Объединение тестов

Первым возможным улучшением предложенного теста является использование для небольших целых $n > 1$ в качестве оснований теста Рабина-Миллера последовательных простых чисел больших или равных 2. К примеру, доказаны следующие утверждения:

- Если $n < 2 \cdot 10^{12}$ и является 2, 3, 5, 7 и 11-SPRP, то оно простое.
- Если $n < 3 \cdot 10^{14}$ и является 2, 3, 5, 7, 11 и 13-SPRP, то оно простое.
- Если $n < 3{,}4 \cdot 10^{14}$ и является 2, 3, 5, 7, 11, 13 и 17-SPRP, то оно простое.

Следует обратить внимание на то, что эти тесты не вероятностные, они *доказывают* простоту числа. Логичным завершением рассмотрения такого подхода является утверждение, предложенное Миллером:

Тест Миллера: *если верна расширенная гипотеза Римана (Enhanced Riemann Hypothesis)*, *то* если n является a-SPRP для всех a: $1 < a < 2 (\log n)^2$, то n – простое.

Сама гипотеза Римана слишком сложна, чтобы ее здесь приводить, но если она верна, то мы получим полиномиальный алгоритм, проверяющий является ли n простым числом.

Если рассматривать по отдельности, то все приведенные в предыдущих разделах проверки на простоту работают либо недостаточно надежно, либо недостаточно быстро. Вместе с тем, если объединить тест Рабина-Миллера с пробным делением, мы получаем алгоритм, работающий

намного лучше и того и другого. Дело в том, что для большого n вычислительно проще провести пробное деление на небольшое простое число, а не тест Рабина-Миллера. При этом сразу же отбрасывается достаточно большая часть составных чисел. К примеру, при проверке делимости на 3, 5 и 7 отбрасывается ~50% всех составных нечетных чисел, при проверке делимости на все простые меньшие 256 отбрасывается уже ~80% всех составных нечетных чисел.

Объединенный алгоритм (для k-битного числа):
1. Пробное деление на все числа меньше некоторого граничного числа B. Это число определяется временем полного возведения в степень по модулю k-битного числа и полным временем проверки делимости k-битного числа на небольшое простое число и равно примерно их отношению.
2. Тест Рабина-Миллера, проводимый t раз, где t выбирается из соображений необходимой точности.

При правильном выборе значений B и t можно получить существенно лучшие результаты, чем при использовании просто теста Рабина-Миллера. К примеру, для $k = 500$ и $t = 6$ вероятность ошибки такого алгоритма $\sim 2^{-92}$, что существенно меньше, чем $2^{-2t} = 2^{-12}$.

Классические тесты

В этом разделе рассматриваются алгоритмы, которые позволяют *доказать* простоту чисел. Чаще всего эти алгоритмы используются для генерации простых чисел определенного вида, например, чисел Ферма или чисел Мерсенна. Таким образом, с практической точки зрения они не имеют большой ценности для криптосистем с открытым ключом. С другой стороны, именно эти методы проверки позволили обнаружить подавляющую часть самых больших известных простых чисел. Более того, практически все самые большие известные простые числа имеют вид n - 1 или n + 1. Поэтому логично будет рассмотреть два этих случая.

Проверки чисел вида n + 1

В этом разделе мы будем рассматривать тесты на простоту числа n, основанные на том, что мы знаем полное или частичное разложение на множители числа $n - 1$. Такой подход может показаться странным – если мы можем разложить число такого порядка на множители, то мы должны уметь и определять является ли такое число простым. На самом деле, разложение на множители $n - 1$ можно достаточно просто найти, если n имеет определенный вид, в качестве примера можно выбрать *числа Ферма* $F_n = 2^{2^n} + 1$ или такие числа как $n! + 1$.

В 1891 году Лукас (*Lucas*) предложил формулировку малой теоремы Ферма, которую можно использовать в качестве практического теста:

Тест Лукаса: рассмотрим целое $n \geq 3$. Если для каждого простого q, делящего $n - 1$ существует целое a такое что:
1. $a^{n-1} = 1 \pmod{n}$, и
2. $a^{(n-1)/q} \neq 1 \pmod{n}$,
то n – простое.

Как видно, для этой проверки нам необходимо знать полное разложение $n - 1$ на простые множители. Более сильной версией утверждения была бы такая, где нам требовалось бы знать не полное, а *частичное* разложение $n - 1$ на простые множители. Такой вариант был сформулирован в 1914 году Поклингтоном (*Pocklington*). Его немного дополненная и измененная версия выглядит так:

Тест Поклингтона: рассмотрим целое $n \geq 3$ и $n = FR + 1$ (то есть F делит $n - 1$), причем $F > R$, НОД(F, R) = 1 и известно разложение F на простые множители. Тогда, если для любого простого q, делящего F существует такое целое a > 1, что
1. $a^{n-1} = 1 \pmod{n}$, и
2. НОД($a^{(n-1)/q} - 1, n$) = 1
то n – простое.

Это утверждение дает нам возможность сформулировать несколько важных следствий для чисел определенного вида.

Теорема Пепина (*Pepin*)**:** пусть F_n это n-ое число Ферма и $n > 1$. Тогда F_n - простое тогда и только тогда, когда $3^{(F_n-1)/2} = 1 (\mathrm{mod}\, F_n)$.

Теорема Прота (*Proth*)**:** Пусть $n = h \cdot 2^k + 1$, причем $2^k > h$. Если существует такое целое a, что $a^{(n-1)/2} = -1 \pmod{n}$, то n – простое.

В качестве иллюстрации последней теоремы, можно привести такой пример, что пятое по величине из известных простых чисел имеет вид $28433 \cdot 2^{7830457} + 1$, найденное на основании теоремы Прота.

Проверки чисел вида *n* - 1

В этом разделе мы будем рассматривать числа только определенного вида. Можно было бы рассмотреть общие последовательности Лукаса, но так как их основное практическое приложение это числа Мерсенна, то их мы и будем рассматривать. Эти числа нас интересуют по двум причинам: во-первых, с ними очень удобно работать на двоичном компьютере и, во-вторых, в списке самых больших известных простых чисел 7 из первых 10 позиций принадлежат числам Мерсенна.

Числами Мерсенна (*Mercennes*) называются числа вида $2^s - 1$.

В 1930 году Лукасом (*Lucas*) и Лемером (*Lehmer*) было сформулировано следующее утверждение: пусть s – простое, тогда число Мерсенна $n = 2^s - 1$ является простым тогда и только тогда, когда $S(n - 2) = 0$, где $S(0) = 4$ и $S(k+1) = S(k)^2 - 2 \pmod{n}$.

На основании этого факта можно построить проверку на простоту, которая будет точно говорить является ли для заданного s число Мерсенна простым.

Тест Лукаса-Лемера:
1. С помощью пробного деления проверяем, является ли заданное s простым, если нет, то получившееся число Мерсенна – составное.
2. Задаем $S(0)$ равное 4.
3. Для k от 1 до $s - 2$ вычисляем $S(k) = S(k-1)^2 - 2 \pmod{n}$.
4. Если в результате получился 0, то число Мерсенна простое, иначе – составное.

На сегодняшний день вопрос о существовании бесконечного числа чисел Мерсенна остается открытым. Пока найдено 42 числа, данные о 10 самых больших можно найти в таблице ниже.

Номер	*s*	Дес. цифр
33	859433	258716
34	1257787	378632
35	1398269	420921
36	2976221	895932
37	3021377	909526

Номер	*s*	Дес. цифр
38	6972593	2098960
39	13466917	4053946
40	20996011	6320430
41	24036583	7235733
42	25964951	7816230

Современный подход

Итак, осталось рассмотреть лишь современные подходы к проблеме определения простоты числа. Первым из таких подходов является алгоритм APR (1976-1983), основывающийся на идеях, рассмотренных в последнем разделе, в частности на теореме Поклингтона. Кроме того, для определения простоты числа используется теория эллиптических кривых, разработанная в 1986-1992 годах. Наконец, летом 2002 года индийские математики сумели построить полиномиальный алгоритм, строго определяющий является ли число простым. Последние годы работы ведутся в этом направлении. Кроме того, с ростом возможностей вычислительной техники и повсеместным распространением компьютерных технологий, стали популярны сетевые проекты по нахождению больших простых чисел. В результате деятельности одного из таких проектов – GIMPS (Great Internet Mersenne Prime Search) были обнаружено практически все самые большие числа Мерсенна (на момент написания эссе, последнее число было найдено в начале 2005 года).

Неоклассические алгоритмы: ARP и ARP-CL

В разделе о классических алгоритмах мы рассматривали работу с числами n, для которых мы знаем разложение числа n -1 или n + 1. Можно пойти дальше и рассматривать, например, числа $n^2 + n + 1$ и $n^2 – n + 1$. А можно рассмотреть число вида $n^m – 1$ для больших m. Тогда, любое простое число q такое, что $q – 1$ делит m, по малой теореме Ферма будет делить $n^m – 1$.
Эта идея (а также некоторые другие) позволяют для m = 5040 построить произведение простых чисел большее 10^{52}. Теперь, если показать, что верны теоремы аналогичные классическим (в частности, теореме Поклингтона), мы сможем работать со всеми числами длиной менее 100 цифр – по сути, без явного разложения на множители.
Было показано, что всегда существует целое число m:

$$m < (\log n)^{\log \log \log n}$$

для которого простые q делящие $n^m – 1$, такие что $q – 1$ делит m, имеют произведение порядка квадратного корня из n.
Эти рассуждения дают примерное (крайне примерное) представление о том, как Adleman, Pomerance и Rumely построили свой алгоритм. Вскоре после того, как он был представлен, Cohen и Lenstra модифицировали этот тест и получили практическую версию под названием APRT-CL (улучшение состояло в замене одной из операций суммами Якоби).

Эллиптические кривые: ECPP

Этот современный вариант проверок на простоту также опирается на теорему Поклингтона, но уже для эллиптических кривых. Смысл метода состоит в переходе от групп порядка $n – 1$ и $n +$ 1 к гораздо большему диапазону размеров групп, в котором мы можем искать пригодную для "разложения".
На практике обычно используется версия алгоритма под названием *тест Аткина* или *Elliptic Curve Primality Proving algorithm*.

AKS

Летом 2002 года индийские математики Агравал (*Agrawal*), Кайал (*Kayal*) и Саксена (*Saxena*) нашли *полиномиальный детерминированный* алгоритм проверки числа на простоту. Их алгоритм основывался на следующей версии малой теоремы Ферма:
Теорема: Пусть a и p взаимно простые целые числа и $p > 1$. Число p является простым тогда и только тогда, когда $(x - a)^p = (x^p - a) \pmod p$
Доказательство: Если p – простое, то p делит биномиальные коэффициенты ${}_pC_r$ для r = 1, 2, ... p-1. Значит, $(x - a)^p = (x^p - a^p) \pmod p$ и соответствие этого выражения требованию теоремы

следует из малой теоремы Ферма. Теперь, пусть p – составное и пусть q – простой делитель p. Пусть q^k максимальная степень q которая делит p. Тогда q^k не делит ${}_pC_q$ и взаимно просто с a^{p-q}. Отсюда, коэффициент перед x^q в левой части требуемого равенства не равен нулю, а в правой равен. Противоречие.
Сам алгоритм для числа $n \geq 23$ (это "странное" число получается из одного из требований для корректной работы алгоритма) выглядит следующим образом:

Тест AKS (псевдокод):

```
if (n is has the form a^b with b > 1) then output COMPOSITE

r := 2
while (r < n) {
    if (gcd(n,r) is not 1) then output COMPOSITE
    if (r is prime greater than 2) then {
        let q be the largest factor of r-1
        if (q > 4sqrt(r)log n) and (n^(r-1)/q is not 1 (mod r)) then
break
    }
    r := r+1
}

for a = 1 to 2sqrt(r)log n {
    if ( (x-a)^n is not (x^n-a) (mod x^r-1,n) ) then output COMPOSITE
}

output PRIME;
```

Доказательство корректности работы этого теста частично основывается на доказанной выше теореме (условия на q и r), частично на более сложных утверждениях, выходящих за рамки этой работы.
Несмотря на то, что этот тест является полиномиальным и детерминированным его использование на сегодняшний день осложнено невысокой скоростью работы. На текущий момент требуется построить "умный" алгоритм, который будет работать быстрее, чем приведенный выше. Уже сейчас существует множество улучшений предложенного алгоритма, уменьшающих постоянные, используемые во временном анализе, по крайней мере, в 2000000 раз.

Сводная таблица

Чтобы подвести некий итог обзора, составим сводную таблицу с типами проверок и областями их применения.

Тест	Тип теста	Где используется
Пробное деление	детерминированный	В чистом виде не используется из-за большой вычислительной сложности. Пробное деление на маленькие простые числа используется как один из шагов во многих тестах.
Ферма	вероятностный	В чистом виде не используется нигде. Может использоваться на начальной стадии проверки простоты для очень больших чисел.
Леманна	вероятностный	Не используется, так как хуже аналогичного вероятностного теста Рабина-Миллера.
Рабина-Миллера	вероятностный	В чистом виде может использоваться в криптосистемах с открытым ключом для построения простых ключей длиной 512, 1024 и 2048 бит. Но в большинстве случаев эта проверка вытеснена объединенным тестом.
Миллера	детерминированный	На практике не используется, так как пока не доказана расширенная гипотеза Римана.
Объединенный (Р-М + пробное деление)	вероятностный	В криптосистемах с открытым ключом для построения простых ключей длиной 512, 1024 и 2048 бит.
Лукаса	детерминированный	Для получения больших простых чисел определенного вида.
Поклингтона	детерминированный	Для получения больших простых чисел с частично известной факторизацией $n - 1$. Также на основании теорем аналогичной этой, но для других групп, построены тесты APR и ECPP.
Пепина	детерминированный	Для получения больших простых чисел Ферма.
Прота	детерминированный	Для получения больших простых чисел определенного вида.
Лукаса-Лемера	детерминированный	Для получения больших простых чисел Мерсенна.
APR	детерминированный	В качестве детерминированной быстрой проверки на простоту.
ECPP	детерминированный	В качестве детерминированной быстрой проверки на простоту.
AKS	детерминированный	В качестве детерминированной полиномиальной проверки на простоту.

Заключение

Как видно из вышеприведенной таблицы, различные проверки на простоту служат для двух целей:
(1) для получения очень больших простых чисел из академического интереса или для участия в каком-либо финансируемом проекте, скажем, в GIMPS
(2) для генерации простых чисел заданного размера для использования в криптографических алгоритмах.
В зависимости от того, какую задачу требуется решить и нужно выбирать метод. Для задачи (2) хорошо себя зарекомендовала комбинация из пробного деления и теста Рабина-Миллера. Для задачи (1) лучше всего использовать классические проверки Прота и Лукаса-Лемера. Если требуется определить, является ли заданное большое число n простым, то здесь может подойти комбинация из пробного деления, разложений чисел $n - 1$ и $n + 1$ на простые множители и, возможно, какие-то из современных алгоритмов. Следует отметить, что современные алгоритмы работают очень медленно по сравнению с классическими, и лучше использовать их готовые реализации, так как самостоятельно код написать довольно сложно.

Литература

- *"Finding primes and proving primality"*
  http://primes.utm.edu/prove/index.html
- *"Handbook of Applied Cryptography"*
  A. Menezes, P. van Oorschot, S. Vanstone
  http://www.cacr.math.uwaterloo.ca/hac/
- *"Прикладная криптография"*
  Брюс Шнайер
  М. : ТРИУМФ 2002. – 816 с.
- *"Proving primality after Agrawal-Kayal-Saxena"*
  D. Bernstein
  http://cr.yp.to/papers/aks.pdf
- *"О доказательстве простоты чисел (следуя работе M. Agrawal, N. Kayal, N.Saxena)"*
  Ю. В. Нестеренко
  http://www.cryptography.ru/db/msg.html?mid=1169256
- *"Detecting perfect powers in essentially linear time"*
  D. Bernstein
  http://cr.yp.to/papers/powers.pdf