Primality Tests (using randomised algorithm)

Let's say n is odd since the even case is very trivial there are 3 methods with randomised algorithm

1) Fermat test

2) Miller Rabin test

3) Solovay Strassen

Fermat test:

If n is prime then for any 'a' we have a^(n-1) = 1(mod n) this suggests the Fermat test for a prime number.

Pick a random number a from (1....n-1) and check if a^(n-1) == 1(mod n). If not, then n must be composite else it may be prime.

We may get equality even when n is not prime

E.g.: 561 = 3*11*17 -> a ^560 = 1 (mod n)

Wondering why it says it may be prime let's checkout

Here, no matter what a we pick ,561 always passes the Fermat test despite being a composite number

So why that's so, as long as a is co-prime with n (Carmichael numbers)

If a is not co-prime to n then the Fermat test fails, but in this case, we may as well forget tests and recover a factor of n simply by computing GCD (a, n).

Note: If a factor of such a number is encountered while randomise check it will always give the right result.

Miller-Rabin test:

Miller-Rabin is definitely better, by recalling n is prime if and only if the solution of n^2 = 1 (mod n)

Where n = ±1

So, if n passes a Fermat test that is a^(n-1) = 1, then we also can check a^(n-1)/2 = ±1.

However, number like 1729 still fools.

What about iterations?

Here we continue halving the exponent until we reach a number besides 1. If its anything but –1 then n must not be composite.

e.g. let's 2^s be the largest power of 2 dividing n-1

n-1 = 2^s*q for some odd number q, each member of the sequence

$a^{(n-1)} = (a^{2^s *q}), (a^{2^{(s-1)} *q}), \ldots, a^q$.

Is a sq. Root of the preceding member then if n is prime, this seq. begins with 1 and either every member is 1 or the first member of the Seq. Not equal to 1 is –1

Simply...

If the above seq. Does not begin with 1, or the first member of the seq. that is not 1 is also not –1 then n is not prime

It turns out for any composite n, including Carmichael numbers the probability n passes the miller-Rabin test is almost ¼.

If n fails the miller-Rabin test with a seq. Starting with 1, then we have a non-trivial sq. Root of

1 (mod n), So we can effectively factor n, thus Carmichael number always easy to factor.

When applied on a number of the form p* q where p, q are large primes then miller-Rabin fails because the seq. Doesn't start with 1.

Algorithm:

1)Given n find $n-1 = 2^s * q$ for some odd q

2)Pick a from [1...n] randomly

3)If $a^q=1$ then n passes (and exits)

4)For i = 0, s-1 see if $a^{((2^i) *q)} = -1$, If so, n passes otherwise n is a composite.

Solovay Strassen:

It's a probabilistic test for prime number.

We need to deal with two type of symbols somewhat they are related let's see them.

Legendre symbol -

This symbol is defined for a pair of integer a and p such that p is prime. It is denoted by (a/p) and

= 0 if a% p = 0

a/p = 1 if there exits an integer k such that $k^2 = a \pmod p$

= -1 otherwise

Other way: -

$(a/p) = (a^{((p-1)/2)} \% p)$ -----condition (i)

Jacobian symbol -

This symbol is a generalization of Legendre symbol, where p is replaced by n where n is n = p1 ^k1 * …. *pn ^kn.

The Jacobean symbol is defined as

(a/n) = (a/p1) ^k1 * (a/p2) ^k2 *…...*(a/pn)^ kn

If n is taken as a prime number then the Jacobean is equal to the Legendre symbol. These symbols have properties given below:

1) (a/n) = 0 if GCD (a, n) != 1, Hence (0/n) = 0. This is because if GCD (a, n) != 1, then there must be some prime pi such that pi divides both a and n. In that case (a/pi) = 0 [by definition of Legendre Symbol].
2) (ab/n) = (a/n) * (b/n). It can be easily derived from the fact (ab/p) = (a/p) (b/p) (here (a/p) is the Legendry Symbol).
3) If a is even, then (a/n) = (2/n) *((a/2)/n).

4) (a/n) = (n/a) *(-1) ^((n-1) *(a-1)/4)if a and n are both odd.

Algorithm for Solovay-Strassen:

Step 1 Pick a random element a < n

Step 2 if gcd(a, n) > 1 then

Step 3 return COMPOSITE

Step 4 end if

Step 5 Compute $a^{(n-1)/2}$using repeated squaring and (a/n)using the Jacobian algorithm.

Step 6 if (a/n)not equal to $a^{(n-1)/2}$then

Step 7 return Composite.

Step 8 else

Step 9 return Prime.

Step 10 endif

Running Time: Using fast algorithms for modular exponentiation, the running time of this algorithm is O(k·log^3 n), where k is the number of different values of a we test.

Accuracy: It is possible for the algorithm to return an incorrect answer. If the input n is indeed prime, then the output will always correctly be probably prime. However, if the input n is composite then it is possible for the output to be incorrectly probably prime. The number n is then called a Euler-Jacobi pseudoprime.