Primality testing

Question

Given a positive integer N, determine whether N is a prime number.

- Test each $1 < x < N$. If none of them divides N, then N is prime.
- (*Trial division method*) Just test $1 < x < \sqrt{N}$.

Definition 4.5.1 (Fermat's primality testing)

If you can find an integer $1 < x < N$ such that

$$x^{N-1} \not\equiv 1 \pmod{N}.$$

Then N cannot be prime (by Fermat's little theorem, 4.4.9). Such an integer x is called a *Fermat witness* for the compositeness of N.

Definition 4.5.1 (Fermat's primality testing)

If you can find an integer $1 < x < N$ such that

$$x^{N-1} \not\equiv 1 \pmod{N}.$$

Then N cannot be prime (by Fermat's little theorem, 4.4.9). Such an integer x is called a *Fermat witness* for the compositeness of N.

Note that, even N is composite, it is still possible that

$$x^{N-1} \equiv 1 \pmod{N}.$$

If this is the case, we say x is a *Fermat liar*.

At first glance, it does not make the primality testing any easier:

- We need to compute an exponential, which seems not easy.
- There are Fermat liars. Hence, applying the testing to only one integer $1 < x < N$ maybe not enough. (Clearly, if x is not coprime to N, then it is a Fermat witness, but it is possible that all integers that are coprime to N are Fermat liars. Such a composite N is called a *Carmicheal numbers*.)

At first glance, it does not make the primality testing any easier:

- We need to compute an exponential, which seems not easy.
- There are Fermat liars. Hence, applying the testing to only one integer $1 < x < N$ maybe not enough. (Clearly, if x is not coprime to N, then it is a Fermat witness, but it is possible that all integers that are coprime to N are Fermat liars. Such a composite N is called a *Carmicheal numbers*.)

But it is in fact way faster than the trial division method, which has time complexity $O(\sqrt{N})$. While the Fermat's primality testing has time complexity $O(K \cdot \log(N))$ (K is the number of x you used in the testing).

Theorem 4.5.2

If there is a Fermat witness in $\Phi(N)$ for the compositeness of N, then at least half of the numbers in $\Phi(N)$ are Fermat witnesses.

Theorem 4.5.2

If there is a Fermat witness in $\Phi(N)$ for the compositeness of N, then at least half of the numbers in $\Phi(N)$ are Fermat witnesses.

Proof. Let a be a Fermat witness. If there is no Fermat liar, we are done. Otherwise, if there is a Fermat liar b, then $ab \pmod N$ is a Fermat witness:

$$(ab)^{N-1} = a^{N-1}b^{N-1} \equiv a^{N-1} \not\equiv 1 \pmod N.$$

Theorem 4.5.2

If there is a Fermat witness in $\Phi(N)$ for the compositeness of N, then at least half of the numbers in $\Phi(N)$ are Fermat witnesses.

Proof. Let a be a Fermat witness. If there is no Fermat liar, we are done. Otherwise, if there is a Fermat liar b, then $ab \pmod N$ is a Fermat witness:

$$(ab)^{N-1} = a^{N-1}b^{N-1} \equiv a^{N-1} \not\equiv 1 \pmod N.$$

$b \mapsto ab$

Moreover, since $a \in \Phi(N)$, $\boxed{\cdot a \pmod m}$ is invertible. Hence, we get a injective map from Fermat liars to Fermat witnesses in $\Phi(N)$. Consequently, at least half of $\Phi(N)$ are Fermat witnesses. $\square$

So if we know the composite N is not a Carmicheal number*, then the chance for it to pass K Fermat's primality testing is less than $(\frac{1}{2})^K$. So we don't need many K in general.

*We don't know N a priori. But we can take the distribution of Carmicheal numbers into account. For instance, there are only 8220777 Carmicheal numbers under 10^{20}.

So if we know the composite N is not a Carmicheal number*, then the chance for it to pass K Fermat's primality testing is less than $(\frac{1}{2})^K$. So we don't need many K in general.

Question (Modular exponential)

Fix the modulus m and the base b, effectively compute the natural representative of b^x modulo m.

So if we know the composite N is not a Carmicheal number*, then the chance for it to pass K Fermat's primality testing is less than $(\frac{1}{2})^K$. So we don't need many K in general.

Question (Modular exponential)

Fix the modulus m and the base b, effectively compute the natural representative of b^x modulo m.

The time complexity $O(\log N)$ for exponential computation can be achieved by *binary exponentiation algorithms*.

The basic idea of binary exponentiation algorithms is:

1. Write the exponent in binary digits: $x = \sum_{i=0}^{n-1} a_i 2^i$
2. Then we have
$$b^x = b^{\sum_{i=0}^{n-1} a_i 2^i} = \prod_{i=0}^{n-1} (b^{2^i})^{a_i}.$$
3. The natural representative of b^{2^i} can be computed by iterating squares:
$$b \longmapsto b^2 \longmapsto (b^2)^2 = b^{2^2} \longmapsto \cdots \longmapsto (b^{2^{i-1}})^2 = b^{2^i}$$

Example 4.5.3

Apply Fermat's primality testing to 91 with the base 2.

Example 4.5.3

Apply Fermat's primality testing to 91 with the base 2.

We first write the exponent $91 - 1 = 90$ into binary digits:

$90 = 2^6 + 2^4 + 2^3 + 2.$ 1011010

Example 4.5.3

Apply Fermat's primality testing to 91 with the base 2.

We first write the exponent $91 - 1 = 90$ into binary digits:
$90 = 2^6 + 2^4 + 2^3 + 2$.

We can compute natural representatives of 2^{2^i} as follows:

	2	2^2	2^{2^2}	2^{2^3}	2^{2^4}	2^{2^5}	2^{2^6}
modulo 91	2	4	16	74	16	74	16

Example 4.5.3

Apply Fermat's primality testing to 91 with the base 2.

We first write the exponent $91 - 1 = 90$ into binary digits:
$90 = 2^6 + 2^4 + 2^3 + 2$.

We can compute natural representatives of 2^{2^i} as follows:

	2	2^2	2^{2^2}	2^{2^3}	2^{2^4}	2^{2^5}	2^{2^6}
modulo 91	2	4	16	74	16	74	16

Then we have

$$2^{90} = 2^{2^6} \cdot 2^{2^4} \cdot 2^{2^3} \cdot \cancel{2^{2^3}} \cdot 2^{2} \equiv \underbrace{\underbrace{16 \cdot 16}_{74} \cdot 74}_{16} \cdot 4 \equiv 64 \pmod{91}$$

$\not\equiv$ -1

So 2 witness 91 being a composite.

Some remarks for binary exponentiation algorithms:

- We do not need natural representatives of b^{2^i}. Instead, using *minimal representatives** maybe more effective.

minimal abs value

*The *minimal representative* of a congruence class α (modulo m) is the representative a of α such that $-\frac{m}{2} < a \leqslant \frac{m}{2}$.

Some remarks for binary exponentiation algorithms:

- We do not need natural representatives of b^{2^i}. Instead, using *minimal representatives** maybe more effective.
- The dynamic of $\boxed{(\,\cdot\,)^2 \pmod{m}}$ will eventually fall in a circle since $\mathbb{Z}/m$ is finite. So we only need a finite step to generating all the natural representatives of b^{2^i}.

*The *minimal representative* of a congruence class α (modulo m) is the representative a of α such that $-\frac{m}{2} < a \leqslant \frac{m}{2}$.

Some remarks for binary exponentiation algorithms:

- We do not need natural representatives of b^{2^i}. Instead, using *minimal representatives** maybe more effective.
- The dynamic of $\boxed{(\,\cdot\,)^2 \pmod{m}}$ will eventually fall in a circle since $\mathbb{Z}/m$ is finite. So we only need a finite step to generating all the natural representatives of b^{2^i}.
- We still need to do the multiplication of n congruence classes, but we may do it in a clever way (such as: pairing a square).

*The *minimal representative* of a congruence class α (modulo m) is the representative a of α such that $-\frac{m}{2} < a \leqslant \frac{m}{2}$.