22. A quick primality test

Prime numbers are one of the most basic objects in mathematics and one of the most basic questions is to decide which numbers are prime (a clearly related problem is to find the prime factorisation of a number). Given a number n one would like a quick way to decide if n is prime, say using a computer. Quick can be given a precise meaning. First one measures the **complexity** by the number of digits, or what comes to the same thing $d = \lceil \log n \rceil$ (we will work base 2, so that this is the same as the number of bits). The larger d the longer it will take to decide if n is prime, in general. We would like an algorithm that runs in **polynomial time**, that is, the algorithm is guaranteed to be over in a time given by a function that is polynomial in d; in essence we are looking for an upper bound for the running time of the form $c \cdot d^m$, where c is some constant and m is an integer.

This very famous problem was solved in 2002 by Manindra Agrawal, Neeraj Kayal and Nitin Saxena, the last two of whom were graduate students, in computer science from India

The basis of their algorithm is the following simple idea. If n is a prime number then

$$a^n = a \mod n,$$

for every integer $1 \leq a \leq n-1$. This at least gives a way to test if a number is not prime.

Definition 22.1. *A natural number n is called a **Carmichael number** if*

$$a^n = a \mod n,$$

for every integer $1 \leq a \leq n-1$ and yet n is not prime.

Unfortunately Carmichael numbers exist; the first such number is $561 = 3 \cdot 11 \cdot 17$. To remedy this, the next idea is that one can test primeness if one works with polynomials, that is, if one works in $\mathbb{Z}_n[x]$ and not just $\mathbb{Z}_n$.

Lemma 22.2. *Let $n \in \mathbb{N}$ be a natural number, $n \geq 2$.*

Assume that a and n are coprime. Then n is prime if and only if

$$(x+a)^n = x^n + a \in \mathbb{Z}_n[x].$$

Proof. If n is prime then the map

$$\phi \colon \mathbb{Z}_n[x] \longrightarrow \mathbb{Z}_n[x] \qquad \text{given by} \qquad \phi(f) = f^n,$$

is a ring homomorphism. In particular

$$(x+a)^n = \phi(x+a) = \phi(x) + \phi(a) = x^n + a^n = x^n + a \in \mathbb{Z}_n[x].$$

Now suppose that n is composite. Pick a prime p that divides m and suppose that $n = p^k m$, where m is coprime to n. The coefficient of x^p is

$$c = a^{n-p}\binom{n}{p}.$$

Now p does not divide a^{n-p} and p^k does not divide $\binom{n}{p}$, so that p^k does not divide c. Therefore the coefficient of x^p in

$$(x+a)^n - x^n - a$$

is not zero in $\mathbb{Z}_n$. □

So now we have a way to test if n is prime. Pick an integer $1 < a < n$. If a and n are not coprime then n is composite. Otherwise compute

$$(x+a)^n - x^n - a \in \mathbb{Z}_n[x].$$

The problem is that this takes way too long. There are n coefficients to compute, and n is exponential in d, not polynomial.

To remedy this, pick a small (for example, something that is polynomial in d) natural number r. Let

$$I = \langle x^r - 1, n\rangle \triangleleft \mathbb{Z}[x],$$

the ideal generated by $x^r - 1$ and n inside the ring $\mathbb{Z}[x]$ and let

$$R = \frac{\mathbb{Z}[x]}{I} = \frac{\mathbb{Z}_n[x]}{\langle x^r - 1\rangle},$$

the quotient ring. Computations in R proceed much faster than in $\mathbb{Z}_n[x]$, since we only need to compute r coefficients and not n. Obviously if n is a prime number then

$$(x+a)^n = x^n + a \in R,$$

The problem is that even if n is composite, there might be a handful of a and r such that

$$(x+a)^n = x^n + a \in R.$$

The trick is to find bounds on a and r which only depend on d.

Recall that $\varphi(r)$ denotes the cardinality of U_r, that is, the number of integers between 1 and $r-1$ coprime to r. We will use the notation $g(d) \sim O(f(d))$ if there is a constant c such that

$$g(d) \leq c \cdot f(d)$$

for all $d \in \mathbb{N}$.

In terms of running time, note that it takes $O(d)$ time to add, multiply or divide two numbers with d digits (aka bits); more generally

it takes $O(d \cdot m)$ time to add, multiply or divide two polynomials of degree m with coefficients with at most d digits.

We will need a simple result in number theory whose proof we omit:

Lemma 22.3. *The product of all prime numbers r between N and $2N$ is greater than 2^N for all $N \geq 1$.*

Here is the algorithm:

Algorithm 1 Algorithm for primality testing

Require: integer $n > 1$

1: **if** $n = a^b$ for $a \in \mathbb{N}$ and $b > 1$ **then** n is composite.
2: Find the smallest integer r such that the order of n in $\mathbb{Z}_r$ is greater than d^2.
3: **if** the gcd of a and n lies between 1 and n, for some $1 \leq a \leq r$ **then** n is composite.
4: **if** $n \leq r$ **then** n is prime.
5: **for** $a = 1$ to $\lfloor \sqrt{\varphi(r)}d \rfloor$ **do**
6: **if** $(x + a)^n \neq x^n + a \in R$ **then** n is composite.
7: n is prime.

By (22.7) $r \leq d^5$ so that step 4 is relevant only if $n \leq 5,690,034$.

Let us check that this algorithm works. There are two issues. How long will it take this algorithm to run and why does it give the right answer?

Theorem 22.4. *Algorithm* (1) *takes no longer than $O(d^{21/2})$ to run.*

Theorem 22.5. *Algorithm* (1) *returns composite if n is composite and algorithm* (1) *returns prime if n is prime.*

Half of (22.5) is easy:

Lemma 22.6. *If n is prime then algorithm* (1) *returns prime.*

Proof. If n is prime then the conditions in steps (1), (3) and (5) will never be satisfied; so eventually steps (4) or (6) will tell us n is prime. □

To prove (22.5) it remains to show that if the algorithm returns prime then in fact n is prime. If the condition in step 4 is satisfied then it is clear that n is prime, since if we got to step 4 then every integer a less than n is coprime to n.

So to check (22.5) we may assume that we get to step 6. We will need a result about the size of r for both results:

Lemma 22.7. *There is a prime $d^5 \le r \le 2d^5$ for which the order of n in U_r is greater than d^2.*

Proof. Suppose not. Then the order of n in U_r is always at most d^2, for every prime r between $N = d^5$ and $2N$. In particular the order of n in U_r is i, for some $1 \le i \le d^2$, so that r divides $n^i - 1$. Therefore the product of the primes r between N and $2N$ divides the product of $n^i - 1$ between 1 and d^2.

$$\begin{aligned} 2^N &\le \prod_{N \le r \le 2N} r \\ &\le \prod_{1 \le i \le d^2} (n^i - 1) \\ &< n^{\frac{1}{2}d^2(d^2+1)} \\ &\le 2^N, \end{aligned}$$

a contradiction. □

Note a useful trick to compute powers quickly. Suppose we want to compute 3^{16}. First we compute $3^2 = 3 \cdot 3$. Then we compute $3^4 = 3^2 \cdot 3^2$, then $3^8 = 3^4 \cdot 3^4$ and then $3^{16} = 3^8 \cdot 3^8$. This involves four multiplications, rather than 15. To compute a^n, write down n in binary,

$$n = a_d 2^d + a_{d-1} 2^{d-1} + \cdots + a_0.$$

Now compute the powers a^{2^i}, $1 \le i \le d$ and take the product over those i such that $a_i = 1$. For example, consider computing 5^{13},

$$13 = 2^3 + 2^2 + 1 \qquad \text{so that we compute} \qquad 5^{13} = 5^8 \cdot 5^4 \cdot 5.$$

Proof of (22.4). In the first step the number of digits of a is no more than d/b. Given a and b it takes no more than d^2 computations to calculate a^b. So the first step takes at most $O(d^3)$ time to run.

In the second step we need to find an integer r so that the order of n in U_r is at least d^2. Given r we just need to compute n^i modulo r for every $i \le d^2$. This will take at most $O(d^2 \cdot \log r)$ computations. By (22.7) we need only check $O(d^5)$ different values for r. So step 2 takes no longer than $O(d^7)$.

The third step involves computing the gcd of two numbers; each gcd takes only time $O(d)$. The third step takes $O(d^6)$. Step 4 takes time $O(d)$.

Step 5 involves $\sqrt{\varphi(r)}d$ iterations. Each iteration in step 6 involves checking an equation involving polynomials of degree r with coefficients

of size d, where we multiply d polynomials; each equation therefore takes $O(rd^2)$ time to check. Step 5 then takes

$$O(r\sqrt{\varphi(r)}d^3) = O(r^{3/2}d^3) = O(d^{21/2}).$$

It is clear that step 5 dominates all other steps and so this is the time it takes for the algorithm to run. □

It is interesting to observe that in practice Algorithm (1) isn't the right way to check if n is prime. There is a probabilistic algorithm which runs much faster.

We will need the following useful way to characterise prime numbers:

Lemma 22.8. *Let $n \in \mathbb{N}$ be an odd natural number $n \geq 3$.*

Then n is prime if and only if 1 has precisely two square roots in the ring $R = \mathbb{Z}_n$. Moreover if n is composite we can find at least four square roots, all of which are coprime to n.

Proof. Note that square roots of 1 correspond to roots of the polynomial $x^2 - 1 \in R[x]$. Note also that ± 1 (which are distinct, as $n > 2$) are two roots of $x^2 - 1$.

Suppose that n is prime. Then $F_n = \mathbb{Z}_n$ is a field and the quadratic polynomial $x^2 - 1$ has at most two roots. Therefore it has exactly two roots.

Now suppose that n is composite. Then $n = kl$ where k and l are coprime, not equal to one. By the Chinese remainder theorem,

$$\mathbb{Z}_n = \mathbb{Z}_{kl} \simeq \mathbb{Z}_k \oplus \mathbb{Z}_l.$$

But then $(\pm 1, \pm 1)$ are at least four different roots of $x^2 - 1$. □

So here is the probabilistic algorithm. Suppose we are given an odd natural number $n > 2$. Pick an integer $1 \leq a \leq n$ at random. If a and n aren't coprime then n is composite; we can check this very quickly.

Otherwise compute

$$a^{n-1} \mod n - 1.$$

It this is not 1 then n is not prime by Fermat. As n is odd,

$$m = \frac{n-1}{2}$$

is an integer. Let

$$b = a^m \in R.$$

We can compute b quickly. Also

$$b^2 = a^{2m} = a^{n-1} = 1 \in R,$$

so that b is a square root of 1. If $b \neq \pm 1$ then n is composite. If m is even and $b = 1$ then let

$$l = \frac{m}{2},$$

and compute

$$c = a^l \in R.$$

Then $c^2 = b = 1$ so that if $c \neq pm1$ then n is composite.

We can keep going until what remains is odd. If at any stage we find a square root of 1 which is not ± 1 then we call a a **witness**.

It is not hard to see that at least half of the numbers $1 \leq a \leq n$ coprime to n are witnesses. So if we pick a hundred numbers a at random and none of them are witnesses then n is prime with probability

$$\frac{1}{2^{100}}.$$

Since it is now more likely that the sun will explode in five minutes (in which case we won't care if n is prime) or more practically that the computer will make an error, than n is composite, we can safely assume that n is prime.

Of course this probabilistic algorithm is a little unsatisfying. The AKS test runs slower but at least we know that n is prime. However note that if we find a witness then we know that n is not prime. One can adapt the AKS test to give a fast probabilistic algorithm to show that if n is prime then the algorithm will tell us n is prime with probability $1/2$ (let's say). So now keep alternating between the two probabilistic algorithms. With vanishingly small probability, one algorithm will tell us that n is either prime or composite.

It is perhaps instructive to consider how both algorithms work in a concrete example. In terms of cryptography the most interesting examples are when n is a product of two different primes. Let's take $p = 19$ and $q = 53$. Then $n = 1007$. We check if n is prime.

First the probabilistic algorithm. We pick a random integer $1 \leq a \leq n$. Let's take the random integer $a = 2$. Now

$$1007 = 2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^3 + 2^2 + 2 + 1$$

(google is your friend for this). In particular $d = 9$. Now

$$2^2 = 2 \cdot 2 = 4, 2^{2^2} = 16, 2^{2^3} = 256, 2^{2^4} = 81,$$
$$2^{2^5} = 519, 2^{2^6} = 492, 2^{2^7} = 384, 2^{2^8} = 434, 2^{2^9} = 47.$$

Thus

$$2^{503} = 2 * 4 * 16 * 81 * 519 * 492 * 384 * 434 * 47 = 493,$$

which is not 1. Thus n is composite. If 2 didn't work, then we would try 3, etc.

In the case of the first Carmichael number, 561, one can check that

$$2^{230} = 166,$$

so that 2 is a witness.

For the AKS-test, we first test that n is not a pure power. The square root of 1007 is less than 32. It is easy to see that n is not a square. The cube of 10 is 1000, and so it is easy to see that n is not a cube. If n is a fourth power it is a square. The fifth root of 1007 is less than 5; $4^5 = 1024 \neq n$. The seventh root of n is less than 3, so that n is not a pure power.

Next we find a prime r such that the order of n in U_r is at least d^2. 83 is a prime bigger than $81 = 9^2$. If the order of $n = 1007$ in U_{83}, the non-zero elements of the field $\mathbb{F}_{83}$, is not $82 = 2 * 41$, it must be either 2 or 41. $1007 = 11$, modulo 83.

$$1007^2 = 38$$

so that the order of 1007 is not 2 in U_{83}. But it is 41. Let's try 89. $89 - 1 = 88 = 8 \cdot 11$. 1007 modulo 89 is 28.

$$28^8 = 39 \qquad \text{and} \qquad 28^{11} = 37,$$

so that 1007 has order $88 > 81$ in U_{89}, the non-zero elements of the field $\mathbb{F}_{89}$. As r is prime, $\phi(r) = r - 1$, so that $\phi(89) = 88$. Next we check that n is coprime to every number less than 88. It isn't so we are done.

If we ignore this and assume we get to the next step, we are supposed to compute

$$(x + a)^n - x^n - a \in \frac{\mathbb{Z}_{1007}[x]}{\langle x^{89} - 1 \rangle},$$

for every $1 \leq a \leq \lfloor \sqrt{88}89 \rfloor = 84$. Exercise for the reader.

Let us now turn to the proof of (22.5). By (22.6) we may assume that r is prime. We want to prove:

Proposition 22.9. *Let $n > 1$ be an odd natural number which is not a pure power such that n has no prime factors less than a prime number r, the order of n in $\mathbb{F}_r$ is at least d^2, where $d = \lceil \log n \rceil$ and*

$$(x + a)^n = x^n + a \in R$$

for every integer $1 \leq a \leq \sqrt{r}n = A$, where R is the ring

$$R = \frac{\mathbb{Z}[x]}{\langle n, x^r - 1 \rangle}.$$

Then n is prime.

We will assume that n is composite and derive a contradiction. Suppose that p is a prime dividing n. Then we have

$$(x+a)^n = x^n + a,$$

in the smaller ring

$$\frac{\mathbb{Z}[x]}{\langle p, x^r - 1\rangle} = \frac{\mathbb{F}_p[x]}{\langle x^r - 1\rangle}.$$

Now $\mathbb{F}_p[x]$ is a UFD. So the polynomial $x^r - 1$ factors into irreducible polynomials which are prime. Let $h(x) \in \mathbb{F}_p[x]$ be a prime polynomial dividing $x^r - 1$. Then

$$\mathbb{F} = \frac{\mathbb{Z}[x]}{\langle p, h(x)\rangle} = \frac{\mathbb{F}_p[x]}{\langle h(x)\rangle},$$

is an integral domain, since the ideal $\langle h(x)\rangle$ is prime. On the other hand, it is easy to see that $\mathbb{F}$ has finitely many elements, so that $\mathbb{F}$ is a finite field.

Definition 22.10. *Let G be a group. The* **exponent** *of G is the least common multiple of the orders of the elements of G.*

Lemma 22.11. *Let G be a finite abelian group of order n.*

Then the exponent m of G is smallest value of r such that $g^r = e$. In particular $m = n$ iff G is cyclic.

Proof. By the classification of finitely generated abelian groups, we may find integers $m_1, m_2, \ldots, m_k$ such that

$$G \simeq \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \ldots \mathbb{Z}_{m_k},$$

where m_i divides m_{i+1}. In this case it is clear that $m = m_k$. □

Lemma 22.12. *Let G be a finite subgroup of the multiplicative group of a field F.*

Then G is cyclic.

Proof. Let m be the exponent of G and let n be the order of G. Now G is abelian as F is a field. Thus $m \leq n$ and for every element α of G, $\alpha^m = 1$, so that every element of G is a root of the polynomial

$$x^m - 1 \in F[x].$$

But a polynomial of degree m has at most m roots, and so $n \leq m$. But then $m = n$ and G is cyclic. □

Thus $\mathbb{F}^*$ is a cyclic group. Let G be the subgroup of $\mathbb{F}^*$ generated by x, $x+1$, $x+2$, $\ldots$, $x+A$. The trick is to give lower and upper bounds for the size of G; it is then a relatively easy matter to check these bounds are incompatible.

Note that G has lots of generators and relatively few relations; this is because the order of $n \in U_r$ is relatively large. It is therefore not too hard to give lower bounds for the size of G.

To give upper bounds on the size of G is more complicated; the idea is to generate lots of identities, deduced from knowing that $(x+a)^n = x^n + a \in \mathbb{F}$.