DISCLAIMER: These notes are not necessarily an accurate representation of what I said during the class. They are mostly what I intend to say, and have not been carefully edited.

In this lecture we start to use some of the fancier number theoretic tools introduced over the past two weeks. This lecture will revolve around the question of primality testing. Specifically, we show that we can check whether an \( L \) digit number is prime in \( \text{poly}(L) \) time.

1 Simple Algorithms

Recall that a number of value \( n \) has \( \Theta(\log n) \) digits.

So the ‘standard’ algorithm of checking divisibility by everything up to \( \sqrt{n} \) runs in time \( O(\exp(L/2)) \).

For something faster, recall Fermat’s theorem, which states that if \( n \) is prime, then \( x^{n-1} \equiv 1 \pmod{n} \). This can be used in the converse: if we can find some \( 1 < b < n \) such that

\[
 b^{n-1} \not\equiv 1 \pmod{n}
\]

then we can conclude that \( n \) is not prime.

For the important sub-case of

\[
 n = pq
\]

for primes \( p \) and \( q \) though, this does give a good algorithm. By CRT we have

\[
 \mathbb{Z}_n^* \sim \mathbb{Z}_p^* \times \mathbb{Z}_q^* \sim C_p \times C_q.
\]

In this ‘product representation’, let \( (x, y) \) be a pair such that \( x \) is a generator mod \( p \), and \( y \) be anything mod \( q \). Then note that

\[
 x^n \equiv x^{n \mod (p-1)} \equiv x^{q \mod (p-1)} \pmod{p}
\]

If \( q < p \), we cannot have \( (p-1)|q \), so \( x^n \not\equiv 1 \pmod{p} \).

Now note that if we pick \( x \) randomly, it’s a generator mod \( p \) with probability at least

\[
 \frac{\phi(p-1)}{p} \geq n^{-o(1)} p,
\]

so running \( n^{o(1)} \) rounds of this allows us to distinguish between \( n \) prime and \( n = pq \) with high probability.
The same also holds if \( n \) is not square-free. That is, there is some \( p^e \mid n \). Then for any generator \( g \) modulo \( p^e \), we have
\[
g^{n-1} \equiv g^n \mod (p^e-1)^{-1} \mod p^e.
\]

Let \( n = p^e t \) for some \((t, p) = 1\). Then we have
\[
g^n \mod (p^e-1)^{-1} = g^{p^e-1 t - 1}
\]
Then note that because \( e > 1 \),
\[
p^e-1 t - 1 \equiv -1 \mod p,
\]
so \( n - 1 \) cannot be a multiple of \( p^e-1(p - 1) \). So random numbers certify \( n \) not being prime in this case with constant probability as well.

2 Carmichael Numbers

However, this is not foolproof as a prime test: there are composite numbers \( n \) satisfying the above property. These are called Carmichael numbers. As an example, consider
\[
n = 561 = 17 \times 11 \times 3
\]
By CRT we have:
\[
\mathbb{Z}_{561}^* \sim \mathbb{Z}_{17}^* \times \mathbb{Z}_{11}^* \times \mathbb{Z}_3^* \sim C_{16} \times C_{10} \times C_2.
\]
Here \( C_i \) means cyclic group of order \( i \).

Since 16, 10 and 2 all divide \( n-1 = 560 \), raising to the \((n-1)^{th}\) power sends each element of \( \mathbb{Z}_n^* \) to 1.

Lemma 2.1. Every Carmichael number is squarefree.

Proof. if \( p^2 \) divides \( n \), \( \mathbb{Z}_n^* \) has an element \( x \) of order \( p \) (\( \mathbb{Z}_{p^k}^* \) has such an element because it has a generator, and take this generator to power \( p^k - 2(p - 1) \) does the trick) Then since \( x^p \equiv x \), we get
\[
x^{n-1} \equiv x^{-1} \not\equiv 1.
\]

In 1992, Alford, Granville, and Pomerance proved that there are infinitely many Carmichael numbers [AGP94].

On the other hand, what we’ve proven so far already say that the ‘hard’ cases of primality testing are very specialized. The numbers has to the product of at least three primes, all to the first exponent. Such properties, used directly in conjunction with \( n^{1/4} \) type factoring algorithm (Pollard Rho) gives runtimes of the type \( O(n^{1/8}) \) [Pol74]. Next we see how to do better.
3 Strong Pseudoprime Test

A souped up version of this algorithm, once incorporating Euler’s criteria, works though.

The history of this test is murky. Computer scientists often call it the Miller-Rabin test, after two influential papers by G. Miller [Mil76] and M. Rabin [Rab80]. The idea was also known previously to Dubois (1971) and J. Selfridge (unpublished). For this reason, some prefer the name “strong pseudoprime test”.

The idea is to write down
\[ n = 2^st + 1 \]

and consider for some \( a \),
\[ a^t, a^{2t}, a^{4t}, \ldots, a^{2^it} \]

these numbers become 1 at some point, and if they don’t reach 1 via \(-1\), we get
\[ x^2 \equiv 1 \pmod{n} \]

for some \( x \not\equiv \pm 1 \pmod{n} \). Rearranging and factoring gives
\[ n \mid (x - 1)(x + 1) \]

which contradictions \( n \) being prime. Note that in this case, we can also extract a factorization of \( n \) as well. So this means that the majority of time we run this algorithm, it will simply give \( a^t \equiv 1 \pmod{n} \).

This leads to the algorithm:

\[ \text{MR}(n) /* n > 1, \text{odd} */ \]

1. Write \( n - 1 = 2^k * m \), where \( m \) is odd.

2. Choose \( a, 1 \leq a \leq n - 1 \), at random.

3. Compute the sequence
\[ x_0 = a^m \pmod{n} \] (3)
\[ x_1x_0^2 \pmod{n} \] (4)
\[ \ldots \] (5)
\[ x_k = x_{k-1}^2 \pmod{n} /* x_k = a^{n-1} */ \] (6)

4. if the last item in the sequence before a 1 is \(-1\), say “prime”. In all other cases, say “composite”.

We now prove that for a random \( a \), this algorithm works with probability at least \( 1/2 \). The cases of \( n = pq \) and \( p^2 \mid n \) were already considered earlier, and the \( n \) even case is trivial. So it remains to consider the case of
\[ n = p_1 \cdot p_2 \cdot \ldots \cdot p_k \]
for some $k \geq 3$, and odd primes $p_1 < p_2 < \ldots < p_k$.

The main idea is that if we pick a tuple $(x_1 \ldots x_k)$ (in the CRT sense), and consider exponents $e_1 = t, e_2 = 2t, e_i = 2^t = n - 1$, it’s very unlikely for us to go from

$$x_i^{e_i} \equiv -1 \pmod{p_i} \quad \forall i$$

to

$$x_i^{2e_i} \equiv 1 \pmod{p_i} \quad \forall i$$

at the same step. Aka. the $k$ cyclic groups can’t behave in a ‘synchronized’ manner.

There are two steps for this. Both obtained from considering the $k$ cyclic groups independently.

1. The number of $x$ with $x^t \equiv 1 \pmod{p_i}$ is at most $(p_i - 1)/2$. This is because $p_i - 1$ is even, while $t$ is odd, so $t \pmod{p_i}$ is odd. The number of such elements satisfying

$$x^t \equiv 1 \pmod{p_i}$$

is then at most $t \pmod{p_i}$, which is at most $(p_i - 1)/2$ since it divides $p_i - 1$. This gives at most $1/2$ residues for each $p_i$, so the total fraction of such numbers is at most $2^{-k}$.

2. Suppose $x^{2st} \equiv 1 \pmod{p_i}$ for some $s > 0$, then $x^{2s^{-1}t}$ is equally ‘likely’ to be $\pm 1$. Formally, to see this via generators, we have $2st \pmod{p_i}$ is even, so by considering $x = g^z$ for some generator $g$, we have that the exponents of $g$ to either

$$2st \mod (p_i - 1) \quad \text{and} \quad 2st \mod (p_i - 1) + p_i - 1$$

can be $x^{2s^{-1}t}$, and only one of them can be $1$. So we get that at most half of the $x$s satisfy

$$x^{2s^{-1}t} \equiv x^{2st} \equiv 1 \pmod{(p_i - 1)}.$$

Once again, multiplying this over the $k$ prime factors gives that at most $2^{-k}$ of them are bad.

That least at least $1 - 2 \cdot 2^{-k} \geq 3/4$ of the residues certify that $n$ is not a prime.

References

