Algebra and Computation Course Instructor: V. Arvind

Lecture 12: The AKS Primality Test

Lecturer: V. Arvind *Scribe: Ramprasad Saptharishi*

1 Overview

We shall take a small detour before we go into factorising polynomials. In this class we shall look at the AKS primality test, an unconditional, deterministic polynomial time algorithm for primality testing.

2 Preliminaries

Given a number $n \in \mathbb{N}$, we wish to test whether the number is prime or not. And since n is given in binary (hence input size is $O(\log n)$), the running time is to be polynomial in $\log n$.

The AKS algorithm uses the following proposition to distinguish between primes and composites.

Proposition 1. *If $(a, n) = 1$, then $(X + a)^n = X^n + a \pmod n$ if and only if n is prime.*

Proof. If n is a prime, then we have already seen in the earlier class that $(X + a)^n = X^n + a^n = X^n + a \pmod n$.

Suppose n was composite and p was a prime divisor of n. Let the largest power of p that divides n be p^k. The coefficient of x^p in $(X+a)^n$ is $\binom{n}{p}a^{n-p}$. a anyway is coprime to n, and hence can be ignored. But it is easy to see that p^k does not evenly divide $\binom{n}{p}$ (since a power of p is knocked off from the n) and hence that term would survive mod$n$. □

There are however two problem with this, firstly being computing $(X + a)^n$ efficiently, but that we saw last class (using repeated squaring and the binary representation of n). A more serious problem is that n is exponential in the input size, and the polynomial would be too large to compare and check if it is $X^n + a$.

Getting around this difficulty was gave the AKS test.

3 The Primality Test

The idea to get around the difficulty of exponential degree is to check it modulo polynomials of "small" degree, a "small" number of times.

We shall give the algorithm first and then show that it is infact correct and that it runs in time polynomial in $\log n$.

Algorithm 1 AKS Primality Test:

Input: n in binary

1: Check if n is of the form a^b for $b \geq 2$. If yes, **output** COMPOSITE
2: Find the least r such that the order of n modulo r (denoted by $O_r(n)$)is at least $4\log^2 n$
3: If $(a, r) \neq 1$ for $1 \leq a \leq r$, **output** COMPOSITE
4: If $n < r$, **output** PRIME
5: **for** $a = 1$ to $\lfloor 2\sqrt{\phi(r)}\log n \rfloor$ **do**
6: **if** $(X + a)^n \neq X^n + a \pmod{n, X^r - 1}$ **then**
7: **output** COMPOSITE
8: **end if**
9: **end for**
10: **output** PRIME

Apart from step 2, it is clear that the algorithm will run in time polyomial in the input length. As for step 2, the following lemma would tell us that we can indeed find such an r quickly.

Lemma 2. *If m is an odd number, then the lcm of $1, 2, \cdots, m$ is atleast 2^{m-1}.*

Proof. Let $m = 2n + 1$. Consider the following integral:

$$\int_0^1 x^n(1-x)^n dx$$

Since $x(1-x) < 1/4$, this integral is upper bounded by 2^{2n}. But if we were to expand $(1-x)^n$ using the binomial theorem, we have

$$\begin{aligned} 2^{-2n} \leq \int_0^1 x^n(1-x)^n dx &= \int_0^1 \sum_{k=0}^{n} (-1)^k \binom{n}{k} x^{n+k} \\ &= \sum_{k=0}^{n} (-1)^k \binom{n}{k} \frac{1}{n+k+1} = \frac{M}{N} \end{aligned}$$

Clearly, the denomiantor is atmost the lcm (L) of the numbers $1, 2, \cdots 2n+1$ and M is atleast 1. Hence $L2^{-2n} > N2^{-2n} \geq 1$ and hence $L \geq 2^{2n}$. $\square$

Lemma 3. *There exists an $r \leq 16\log^5 n$ such that $O_r(n) \geq 4\log^2 n$.*

Proof. Suppose all r's till T were bad, that is, for all $1 \leq r \leq T$ the order of n modulo r was less than $4\log^2 n$. Then, for every r there exists a $j < 4\log^2 n$ such that r divided $n^j - 1$. Therefore, each $1 \leq r \leq T$ divides the product $\prod_{j=1}^{4\log^2 n}(n^j - 1) < n^{16\log^4 n} = 2^{16\log^5 n}$. And therefore, the lcm of the first T numbers divide the product. The bound from the earlier lemma will now force $T < 16\log^5 n$. $\square$

It is now clear that the algorithm runs in time polynomail in $\log n$, each step is clearly a polylog operation. With some work, one can see that this is roughly a $O(\log^{11} n)$ algorithm.

4 Proof Correctness

One way is clear, if the number was a prime then the algorithm would certainly output PRIME. We need to show that if the algorithm outputs PRIME, then it is indeed prime.

Suppose not, let p be a prime divisor of n. And from our initial tests, we know that $p > r$. And further for $a = 1, 2, \cdots l$ where $l = \lfloor 2\sqrt{\phi(r)}\log n \rfloor$,

$$(X + a)^n = X^n + a \pmod{n, X^r - 1} = X^n + a \pmod{p, X^r - 1}$$

Note that from Fermat's little theorem, we have

$$(X + a)^p = X^p + a \pmod{p, X^r - 1}$$

We shall use a small notation here.

Definition 4. *For any function f, a number m is called introspective for f if $f(X)^m = f(X^m) \pmod{p, X^r - 1}$.*

We then have the two simple lemmas.

Lemma 5. *If m and m' are introspective for f, so is mm'.*

Proof.

$$\begin{aligned}
& f(X)^m &=& f(X^m) \pmod{p, X^r - 1} \\
\implies & f(X^{m'})^m &=& f(X^{mm'}) \pmod{p, X^{m'r-1}} \quad \text{(substitute } X^{m'} \text{ for } X\text{)} \\
& &=& f(X)^{mm'} \pmod{p, X^r - 1} \quad (X^r - 1 \text{ divides } X^{rm'} - 1) \\
\implies & f(X)^{mm'} &=& f(X^{mm'}) \pmod{p, X^r - 1}
\end{aligned}$$

□

Lemma 6. *If m is introspective for f and g, then m is introspective for fg.*

Proof. Obvious! □

Let $I = \left\{n^i p^j \ : \ i \geq 0, j \geq 0\right\}$ and $P = \left\{\prod_{a=1}^{l}(X+a)^{e_a} \ : \ e_a \geq 0\right\}$. Then, from the two lemmas, every element of I is introspective for every element in P.

Let G be the subgroup of $\mathbb{Z}_r^\star$, of size t, generated by n and p. Since $O_r(n) \geq 4\log^2 n$, $|G| = t \geq 4\log^2 n$. To do a similar thing for the polynomials, we first need to move from the ring $(\mathbb{F}_p/(X^r-1))$ to a field. Let $X^r - 1 = h_1(X)h_2(X)\cdots h_k(X)$ be the factorization into irreducible factors. Since the primitive r-th root of unity generates all the roots of unity (since we are going mod X^r-1), the primitive root of unity is a root of some irreducible polynomial. Hence we shall look at $\mathbb{F}_r/(h(x))$, which is essentially $\mathbb{F}_r[\eta]$ where η is a primitive r-th root of unity.

Now in the field $\mathbb{F}_r/(h(x))$, look at the multiplicative group. Let $\mathcal{G}$ be the subgroup generated by $\{(X+a)\}_{1\leq a\leq l}$, the set of polynomials in P that are non-zero in $\mathbb{F}_p/(h(x))$.

Lemma 7. $|\mathcal{G}| \geq \binom{t+l-2}{t-1}$

Proof. Since $l = \lfloor 2\sqrt{\phi(r)}\log n\rfloor < 2\sqrt{r}\log n < r < p$, each of $\{(X+a)\}_{1\leq a\leq l}$ are distinct in $\mathbb{F}_p/(h(x))$. At worst $h(x)$ can be equal to one of them, hence at least $l-1$ of them are non-zero and distinct.

Let f and g be two polynomials from P of degree less than t. Suppose $f(x) = g(x)$ in $\mathbb{F}_p/(h(x))$, then we also have $f^m(x) = g^m(x)$ in the field. If we choose $m \in G$, then since m is introspective for f and g, we have $f(X^m) = g(X^m)$ in the field.

Since we can identify X with a primitive r-th root of unity, each of the X^m are distinct. And infact, each of them is a root of the polynomial $H(Y) = f(Y) - g(Y)$. The size of the group G is t but the degree of f and g is less than t, which gives an absurd situation of the polynomial $f-g$ having more roots than its degree in the field.

Hence, if two polynomials of degree less than t are chosen from P, they are mapped to different elements in $\mathbb{F}_p/(h(x))$.

$$|\mathcal{G}| \geq \left|\left\{\prod_{1\leq a\leq l}(X+a)^{e_a} \ : \ \sum e_a \leq t\right\}\right|$$

As we remarked earlier, there can be at most one $X + a_0$ that becomes zero in the field. So essentially, we just need to find the different integer solutions to $\sum e_a \leq t$, summing over all $a \neq a_0$, and this is equal to $\binom{t+l-2}{t-1}$. Hence $|\mathcal{G}| \geq \binom{t+l-2}{t-1}$. □

Lemma 8. *If n is not a power of a prime, $|\mathcal{G}| < n^{2\sqrt{t}}$*

Proof. Consider the subset $I' = \left\{n^i p^j \ : \ 0 \leq i, j \leq \sqrt{t}\right\}$. Hence each element in I' is bounded by $n^{\sqrt{t}} p^{\sqrt{t}} < n^{2\sqrt{t}}$. And further, if n is not a power of a prime, $|I'| = (1+\sqrt{t})^2 > t$. Since $|G| = t$, there exists two distinct m, m' of I' such that $m = m' \pmod{r}$, and hence $X^m = X^{m'} \pmod{r, X^r - 1}$. Also, for any $f \in \mathcal{G}$, $f(X)^m = f(X)^{m'}$. Therefore, if you consider the polynomial $Y^m - Y^{m'}$, every element of $\mathcal{G}$ is a root. And since the degree is bounded by $n^{2\sqrt{t}}$, this forces $|\mathcal{G}| < n^{2\sqrt{t}}$. □

With the choice of l, it is easy to see that the above two lemmas give conflicting bounds (lower bound greater than upper bound), which gives us the desired contradiction to the assumption that n is composite.

Hence, summarizing it in a theorem:

Theorem 9. *The algorithm returns* PRIME *if and only if the input is a prime.* □

5 A short note on identity testing

I haven't taken notes here... would be nice if someone could fill this part.