CSC498 AKS Primality Test

Rayhan Fazal

November 2023

1 Abstract

This is a study done on the AKS Primality Test which was an algorithm developed by Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, who are computer scientists at the Indian Institute of Technology Kanpur. This algorithm is a deterministic polynomial time algorithm for determining whether or not a given positive integer (strictly greater than 1) is prime or composite. The goal of this study is to present the AKS algorithm, and prove that it is in the class P. We will present background information required to understand the algorithm, present the algorithm itself, give a proof of correctness, show that it is in P, provide some examples of tracing through the output of the algorithm, and finally discuss improvements that can be made to the algorithm, according to the authors of AKS.

2 Background information

In this section we will discuss essential background information in number theory that is required to understand the algorithm. AKS relies primarily on a generalized version of Fermat's Little Theorem, and we will also discuss several other theorems, definitions, and lemmas that are also a part of understanding AKS and its proof of correctness. We will refer to the Lemmas that we will present, later on in our proof of correctness.

The AKS primality test is based on a more general version of Fermat's Little Theorem. So let's first list Fermat's little theorem and prove it.

Lemma 1:

Let p be a prime number. Then for any integer x, we have:

$$x^p = x \mod p$$

So in other words, if we compute x^p and divide it by p, the remainder of that will be the same as the remainder of dividing x by p.

Proof:

Recall the binomial theorem, which states that:

$$(x+y)^p = \sum_{0 \leqslant i \leqslant p} \binom{p}{i} x^i y^{p-i} \tag{1}$$

So the binomial coefficients are integers. Now, we want to show that the prime p divides all the binomial coefficients. Recall that the combination formula (p choose i) can be written as:

$$\binom{p}{i} = \frac{p!}{i!(p-i)!}$$

with $1 \leqslant i \leqslant p-1$. So if we rearrange the above equation for $p!$, we get:

$$p! = \binom{p}{i} \cdot i! \cdot (p-i)! \tag{2}$$

Now, we said that the binomial coefficient is an integer, which allows us to make the following argument. Since p is prime, it can divide the LHS of (1) (since the only numbers that divide p are 1 and p). It can also divide the RHS of (1), but cannot divide $i!$ nor $(p-i)!$ for $0 < i < p$ since both of these numbers are essentially products of integers that are less than p, and as p is prime, those integers are not divisible by p. So using the fact that $p|ab$ (a and b are positive integers) implies that either $p|a$ or $p|b$, that means that it is not the case that $p|i!$ or $p|(p-i)!$ (so p can't divide $i!$ or $(p-i)!$) which means that it must be the case that $p|\binom{p}{i}$ and so p must divide the binomial coefficient $\binom{p}{i}$. So what this show us is that $\binom{p}{i} \equiv 0 \pmod{n}$

Now, we will prove Fermat's Little Theorem for positive integers x by using induction on x:

Proof:

Base case: Let $x = 1$. Then, we get:

$$1^p = 1 \mod p \tag{3}$$

(2) is indeed true because we know that p is prime, and so the only numbers that divide it are 1 and p.

Induction case: Assume that for all positive integers x and prime p, we have that:

$$x^p = x \mod p \tag{4}$$

So (4) is the [IH]. We want to show that

$$(x+1)^p = x+1 \mod p \tag{5}$$

So, to do this, notice that by the binomial theorem equation in (1) above:

$$(x+1)^p = \sum_{0 \leqslant i \leqslant p} \binom{p}{i} x^i 1^{p-i} \tag{6}$$

$$= x^p + \sum_{0 < i < p} \binom{p}{i} x^i + 1 \tag{7}$$

Notice that all the coefficients in (7) above, which is $\binom{p}{i}$ are divisible by p (which we proved earlier at the top of this page). This means that:

$$(x+1)^p = x^p + 0 + 1 \tag{8}$$
$$= x + 1 \mod p \tag{9}$$

So we get (8) by seeing that $\binom{p}{i}$ is divisible by p, and we get (9) by [IH] in (4). as needed. □

Let's do some quick examples of Fermat's Little Theorem. Suppose $p = 3$. Let's try $x = 1, 2, 3$:

$$1^3 \mod 3 = 1 = 1 \mod 3$$
$$2^3 \mod 3 = 2 = 2 \mod 3$$
$$3^3 \mod 3 = 0 = 3 \mod 3$$

Let $r \in \mathbb{N}$, $a \in \mathbb{Z}$ with a and r being co-prime.

The order of a modulo r is defined as the smallest number k such that $a^k \equiv 1 \pmod{r}$, denoted as $o_r(a)$.

Also, $\phi(r)$ is Euler's totient function which gives the number of natural numbers less than r that are relatively prime to r, or in other words, all positive integers less than r that are co-prime with r. For example:

$\phi(1) = 0$ (no positive integers strictly less than 1)
$\phi(2) = 1$ (only 1 is co-prime with 2)
$\phi(5) = 4$ (1, 2, 3, 4 are all co-prime with 5)

An interesting relationship between order of a modulo r and Euler's Totient function is that $o_r(a) | \phi(r)$ for all a that are co-prime to r. (The vertical line means that the RHS is divisible by the LHS, so $\phi(r)$ is divisible by $o_r(a)$). This is because it is the result of Euler's Totient Theorem, which states that if x and y are co-prime, then $x^{\phi(y)} \equiv 1 \mod y$. Here, we know that a is co-prime with r, which means that by Euler's Totient Theorem, $a^{\phi(r)} \equiv 1 \mod r$. Thus, it must be the case that $o_r(a)$ divides $\phi(r)$ because $\phi(r)$ is the smallest power that allows us to achieve $a^{\phi(r)} \equiv 1 \mod r$ (we raise a to the power of $\phi(r)$, and so we must be able to divide $o_r(a)$, which includes $a^{\phi(r)}$, by $\phi(r)$. This is also as a result of Euler'e Totient Theorem, and by definition of $o_r(a)$).

(Note that base 2 is used for log here).

Lemma 2:

Let LCM(m) represent the lowest common multiple of the first m numbers. Then,

$$\text{for } m \geqslant 7, \text{LCM}(m) \geqslant 2^m.$$

Lemma 3:

There exists an $r \leqslant \max\{3, \lceil \log(n)^5 \rceil\}$ such that $o_r(n) > \log(n)^2$.

Proof:
(Notice this is trivially true when $n = 2$, as we can choose $r = 3$ to satisfy this. Now consider $n > 2$. Then, $\lceil \log(n)^5 \rceil > 10$ and so we can apply Lemma 2 here. Notice that the largest value of k for any number of the form $m^k \leqslant B = \lceil \log(n)^5, m \geqslant 2$ is $\lfloor \log(B) \rfloor$.

Now let us consider the smallest number r that cannot divide this expression: $(n^{\lfloor \log(B) \rfloor}) \cdot \prod_{i=1}^{\lfloor \log(n)^2 \rfloor} (n^i - 1)$.

Observe that (r, n) cannot be divisible by all the prime divisors of r since if that was not the case, then r will divide $(n^{\lfloor \log(B) \rfloor})$ ((r, n) is all the factors that divide r and n, so if both r and n are divisible by all prime divisors of r, then we should also be able to divide n by those prime divisors). Thus, $\frac{r}{(r,n)}$ will also not divide the above expression (since it's just a factor of (r, n). So this gives us the fact that $(r, n) = 1$ (since both r and n are not divisible by ALL the prime divisors of r, so it must be the case that the only number that divides both is 1). Also, since r cannot divide any of the $(n^i - 1)$'s in the above expression, we get that:

$$(n^{\lfloor \log(B) \rfloor}) \cdot \prod_{i=1}^{\lfloor \log(n)^2 \rfloor} (n^i - 1) < n^{\lfloor \log(B) \rfloor + \frac{1}{2} \cdot \log(n)^2 \cdot (\log(n)^2 - 1)} \tag{10}$$

$$\leqslant n^{\log(n)^4} \tag{11}$$

$$\leqslant 2^{\log(n)^5} \tag{12}$$

$$\leqslant 2^B \text{ (since } B \text{ takes the ceiling)} \tag{13}$$

So by Lemma 2, the LCM of the first B numbers is at least 2^B, and so $r \leqslant B$, as needed. □

Lemma 4:

For a polynomial $f(X)$ and any $m \in \mathbb{N}$, we define m to be introspective for $f(X)$ if $f(X)^m = f(X^m) \pmod{X^r - 1, p}$.

So we can see that both $\frac{n}{p}$ and p are introspective for $X + a$ inside the loop in Line 5 of AKS (which means that $0 \leqslant a \leqslant l$). This is shown by equations (15) and (16) above. Let's do a basic example of introspective numbers:
Let $f(X) = X + 1$, $m = 1$, $p = 3$, and $r = 5$. So

$$f(X)^m = X + 1 = f(X^m)$$

Thus, since $f(X)^m = f(X^m)$, we have that m is introspective for $f(X)$ since $f(X)^m = f(X^m) \pmod{X^r - 1, p}$.

We now introduce another lemma, Lemma 5, which will show that introspective numbers are closed under multiplication:

Lemma 5:

If m and $m^{'}$ are introspective numbers for $f(X)$, then so is the product $m \cdot m^{'}$.

Proof:
Since m is introspective for $f(X)$, by Lemma 4 we get:

$$f(X)^{m \cdot m^{'}} = (f(X^{m}))^{m^{'}} (\mathrm{mod} X^{r} - 1, p) \tag{14}$$

Also, $m^{'}$ is introspective for $f(X)$, applying Lemma 4 and replacing X with X^{m} in the introspective equation for $m^{'}$, we get:

$$\begin{aligned} (f(X^{m}))^{m^{'}} &= f(X^{m \cdot m^{'}})(\mathrm{mod} X^{m \cdot r} - 1, p) \\ &= f(X^{m \cdot m^{'}})(\mathrm{mod} X^{r} - 1, p))(\text{since } X^{r} - 1 \text{ divides } X^{m \cdot r} - 1) \end{aligned} \tag{15}$$

Now we combine the equations (17) and (18) to give us that:

$$f(X)^{m \cdot m^{'}} = f(X^{m \cdot m^{'}})(\mathrm{mod} X^{r} - 1, p)$$

□

Let us now define another lemma, as follows:

Lemma 6:

If m is introspective for $f(X)$ and $g(X)$, then it is also introspective for $f(X) \cdot g(X)$, (So if we have a set of polynomials and m is introspective for all of them, then this is also closed under multiplication).

Proof:

$$\begin{aligned} (f(X) \cdot g(X))^{m} &= f(X)^{m} \cdot g(X)^{m} (\text{raise each function to the power of } m \text{ individually}) \\ &= f(X^{m}) \cdot g(X^{m}) (\mathrm{mod} X^{r} - 1, p) (\text{Applying Lemma 4}) \end{aligned}$$

□

Lemma 7:

$$|G_2| \geqslant \binom{t+l}{t-l} \text{ (choose symbol)}$$

Proof:
Recall from earlier that $h(X)$ is a factor of $Q_T(X)$, which means that X is a primitive rth root of unity in F. So in other words, X is a complex root of Q_T such that when we raise it to the power of r, we get 1.

We want to show that any 2 distinct polynomials in P with degree less than t will map to different elements in G_2. Let's use $f(X)$ and $g(X)$ as these 2 polynomials.

Consider the case where $f(X) = g(X)$ in the field $\mathbb{F}$, and let $m \in I$. This means that we also have $(f(X))^{m} = (g(X))^{m}$ in F. However, we already defined earlier

that m is introspective for both f and g, and that $h(X)$ divides $X^r - 1$, this gives us that $f(X^m) = g(X^m)$ in F. So if we rearrange this equation, we get the polynomial $Q(Y) = f(Y) - g(Y)$ (replaced X^m with Y) which tells us that X^m is a root of $Q(Y)$, for every $m \in G_1$. Since $(m, r) = 1$ as G_1 is a subgroup of Z_r^*, each X^m is a primitive rth root of unity. So there exists $|G_1| = t$ distinct roots of $Q(Y)$ in F. However, we said earlier that the degree of f and g is less than t (that is how we defined it), which means that the degree of $Q(Y)$ (which we got by rearrangement of f and g earlier) is also less than t, which is a contradiction to $Q(Y)$ having t distinct roots, and so $f(X) \neq g(X)$ in F.

Also, notice that $i \neq j$ in F_p for $1 \leqslant i \neq j \leqslant l$ since $l = \lfloor \sqrt{\phi(r)} \cdot \log(n) \rfloor < \sqrt{r} \cdot \log(n) < r$ and $p > r$. So this implies that all elements $X, X+1, X+2, ..., X+l$ in the field $\mathbb{F}$ are distinct. Also, since the degree of h is greater than 1, this means that $X + a \neq 0$ in F for every a where $0 \leqslant a \leqslant l$ (the values of a in the loop in Line 5). So there exists at least $l + 1$ distinct polynomials with degree 1 in G_2, and so there exists at least $\binom{t+l}{t-l}$ distinct polynomials of degree less than t in G_2. $\square$

Lemma 8:

$$\text{If } n \text{ is not a power of } p, \text{ then } |G_2| \leqslant n^{\sqrt{t}}.$$

Proof:
Consider this subset of I, $I' = \{(\frac{n}{p})^i \cdot p^j \mid 0 \leqslant i, j \leqslant \lfloor \sqrt{t} \rfloor\}$.
If n is not a power of p, the I' has $(\lfloor \sqrt{t} \rfloor + 1)^2 > t$ distinct elements. Since $|G_1| = t$, at least 2 numbers in I' must be equal modular r. Let these 2 numbers be $m1$ and $m2$ where $m1 > m2$, which gives us: $X^{m1} = X^{m2} \pmod{X^r - 1}$.
Now, let $f(X)$ be an element in P, then, we get:

$$\begin{aligned}(f(X))^{m1} &= f(X^{m1}) (\mathrm{mod} X^r - 1, p) \\ &= f(X^{m2}) (\mathrm{mod} X^r - 1, p) \\ &= (f(X))^{m2} (\mathrm{mod} X^r - 1, p)\end{aligned}$$

This tells us that $(f(X))^{m1} = (f(X))^{m2}$ in the field $\mathbb{F}$. Thus, $f(X) \in G_2$ is a root of the polynomial $Q'(Y) = Y^{m1} - Y^{m2}$ in the field $\mathbb{F}$. As $f(X)$ is an element of G_2, we get that $Q'(Y)$ has at least $|G_2|$ distinct roots in F. So the degree of $Q'(Y)$ is $m1 \leqslant (\frac{n}{p} \cdot p)^{\lfloor \sqrt{t} \rfloor} \leqslant n^{\sqrt{t}}$. Therefore, $|G_2| \leqslant n^{\sqrt{t}}$. $\square$.

Lemma 9
There exists constants $c > 0$ and n_0 such that for all $x \geqslant n_0$:

$$|\{q \mid q \leqslant x \text{ is prime and } P(q-1) > q^{\frac{2}{3}}\}| \geqslant c\frac{x}{\ln x}$$

This lemma is known to hold for exponents up to 0.6683, and we can use this lemma to speed up AKS, as shown below:

As mentioned earlier, if we can find a high density of primes q such that $P(q - 1) > q^{\frac{2}{3}}$, we can find a prime $r = O(\log(n)^2)$ (approximately) such that $o_r(n) >$

$\log(n)^2$. This fact, as we have combined it with Lemma 9, allows us to bring the runtime of AKS down to approximately $O(\log(n)^{\frac{15}{2}})$. □

Cyclotomic Polynomials: Cyclotomic polynomials are defined as follows: Given a positive integer n, the n-th cyclotomic polynomial is a polynomial $\Phi_n(x)$, such that its first root is equal to 1, and the rest of its roots are the complex non-trivial n-th roots of unity, which simply means that they are complex numbers such that when we raise each of them to the power of n, it gives us 1 for each of them. So the roots can be factored in a way where its roots are evenly spaced out around the unit circle in the complex plane. So $\Phi_n(x) = (x - c_1)(x - c_2)...(x - c_n)$, where $c_1 = 1$ and c_2, ..., c_n are the complex non-trivial n-th roots of unity of Φ, meaning if raise each of c_2, ..., c_n to the power of n, we get 1.

Another way to define Φ is that the n-th cyclotomic polynomial, for any positive integer n, is a unique non-factorable polynomial with integer coefficients (usually its coefficients are over a field, integers are an example of a field), that is a divisor of $x^n - 1$ and is not a divisor of $x^k - 1$ for all $k < n$. Its roots are all n-th roots of unity $e^{2i\pi\frac{k}{n}}$, so more generally speaking, Φ is defined as:

$$\Phi_n(x) = \prod_{1 \leqslant k \leqslant n} (x - e^{2i\pi\frac{k}{n}})$$

, where gcd(k, n) = 1.
So for example, the first 5 cyclotomic polynomials are:

$$\Phi_1(x) = x - 1$$
$$\Phi_2(x) = x + 1$$
$$\Phi_3(x) = x^2 + x + 1$$
$$\Phi_4(x) = x^2 + 1$$
$$\Phi_5(x) = x^4 + x^3 + x^2 + x + 1$$

(So for example when $n = 1$, the only value for k is 1, so gcd(k, n) = 1, and $x - -e^{2i\pi\frac{1}{1}} = x - 1 = \Phi_1(x)$).

Some properties of cyclotomic polynomials are that if n is a prime number, then:

$$\Phi_n(x) = 1 + x + x^2 + ... + x^{n-1} = \sum_{k=0}^{n-1} x^k$$

Also, if $n = 2p$ where p is an odd prime number, then:

$$\Phi_{2p}(x) = 1 - x + x^2 - x^3 + x^4 - ... + x^{p-1} = \sum_{k=0}^{p-1} (-x)^k$$

Groups: A group is defined as:

If G is a non-empty set and "*" is a binary operation defined on G such that the following 4 laws below are satsisfied, then $(G, *)$ is called a group.

Four laws needed for a group:

Closure Law	for $a, b \in G$, we have that $a * b \in G$
Associative Law	$\forall a, b, c \in G : a * (b * c) = (a * b) * c$
Identity element	$\exists e \in G$ s.t $\forall a \in G : a * e = e * a$, where e is the identity element (it is a unique element in G)
Inverse law	$\forall a \in G$, $\exists b \in G$ s.t $a * b = b * a = e$, where $b = a^{-1}$ is the inverse element of a

So that is how Groups are defined. Let us consider some examples of Groups and non-Groups below:

$(\mathbb{R}, +)$ Group: Set of all real numbers with binary operation of addition
$(\mathbb{Z}, +)$ Group : Set of all integers with binary operation of addition
$(\mathbb{N}, +)$: Not a group since it doesn't satisfy Inverse law (e.g $3 \in \mathbb{N}$ has no inverse element (which would be -3 in $\mathbb{N}$ with respect to "+". In other words, we can't add something from $\mathbb{N}$ to 3 to get -3)

Fields: A field is defined as:

A set $\mathbb{F}$ with 2 binary operations on it called addition $(+)$ and multiplicatioon $(\cdot)$, which satisfies the following 11 field axioms below.

Field axioms of $\mathbb{F}$:

Closure under addition	$\forall x, y \in \mathbb{F}$, $x + y \in \mathbb{F}$
Closure under multiplication	$\forall x, y \in \mathbb{F}$, $x \cdot y \in \mathbb{F}$
Commutativity of Addition	$\forall x, y \in \mathbb{F}$, $x + y = y + x$
Associativity of Addition	$\forall x, y, z \in \mathbb{F}$, $(x + y) + z = x + (y + z)$
Additive Identity	$\exists 0 \in \mathbb{F}$, s.t $x + 0 = 0 + x = x \forall x \in \mathbb{F}$
Additive Inverses	$\forall x \in \mathbb{F}$, $\exists y \in \mathbb{F}$ s.t $x + y = y + x = 0$. So y is called the "additive incerse of x and is written as $-x$.
Commutativity of Multiplication	$\forall x, y \in \mathbb{F}$, $x \cdot y = y \cdot x$
Associativity of Multiplication	$\forall x, y, z \in \mathbb{F}$, $(x \cdot y) \cdot z = x \cdot (y \cdot z)$
Multiplicative Identity	$\exists 1 \in \mathbb{F}$ s.t $x \cdot 1 = 1 \cdot x = x \forall x \in \mathbb{F}$
Multiplicative Inverses	$\forall x \in \mathbb{Z}$ s.t $x \neq 0$, $\exists y \in \mathbb{F}$, s.t $x \cdot y = y \cdot x = 1$. So y is called the "multiplicative inverse of x and is written as x^{-1} or $\frac{1}{x}$.
Distributivity of Multiplication over Addition	$\forall x, y, z \in \mathbb{F}$, $x \cdot (y + z) = x \cdot y + x \cdot z$
Distinct Additive and Multiplicative Identities	$1 \neq 0$ (so this excludes 1-element sets from being fields).

Now, let's consider some examples of fields:

Real numbers: $\mathbb{R}$
Integers: $\mathbb{Z}$
Rational numbers: $\mathbb{Q}$
Complex numbers: $\mathbb{C}$
All 3×3 matrices with entries in $\mathbb{R}$
Polynomials $\mathbb{Q}[x]$ (so polynomials where x is the variable and it has rational number coefficients)

3 The AKS algorithm

In this section we will now present the AKS algorithm.

Input: $n \in \mathbb{Z}$, $n > 1$
Output: Return PRIME if n is a prime number, otherwise Return COMPOSITE.

AKS

1. If ($n = a^b$ for some $a \in \mathbb{N}$ and $b > 1$) $\longrightarrow$ Return COMPOSITE
2. Find the smallest $r \in \mathbb{N}$, $r > 0$, such that $o_r(n) > \log(n)^2$
3. If $1 < \text{GCD}(a, n) < n$ for some $a \leqslant r$ (try all natural numbers, starting from 2, up to and including r, check if any of those numbers is NOT co-prime with n) $\longrightarrow$ Return COMPOSITE.
4. If $n \leqslant r \longrightarrow$ Return PRIME
5. For $a = 1, ..., \lfloor \sqrt{\phi(r)} \cdot \log(n) \rfloor$:
   if $((X + a)^n \neq X^n + a(\text{ mod } X^r - 1, n))$, $\longrightarrow$ Return COMPOSITE
6. Return PRIME

4 Proof of correctness

In this section we will prove the correctness of AKS. The proof of correctnes of AKS involves proving a biconditional statement, which we will discuss below. One of the sides of the biconditional (② below) involves two further sub-cases to consider, with the latter sub-case being the part of the proof of correctness that is much more involved and heavy. So let's move now to see the biconditional statement we need to prove to show that AKS runs correctly.

We will prove the correctness of the algorithm by proving the following claim:

Given input $n \in \mathbb{Z}$, $n > 1$, AKS returns PRIME iff n is prime.

Let's prove each part of the biconditional, indicated by ① and ② respectively.

① : Assume n is prime, show that AKS returns PRIME.
This part is trivial. If n is prime, then the if statements in Lines 1 and 3 will be false. By Lemma 1, the if statement in the for loop in Line 5 will also be false for every

iteration. Thus, this leaves Lines 4 and 6 as the output of the algorithm, which will be PRIME in either case, as needed. □

② : Assume that AKS returns PRIME, show that n is prime.
If AKS returns PRIME, then there are 2 sub-cases to consider. Either it returns PRIME in Line 4, or in Line 6. Let us consider each sub-case individually.

(Sub-case 1): Assume AKS returns PRIME in step 4. This means that n must be prime as well since if that was not the case, then AKS would have found that $n = a^b$ for some $a \in \mathbb{N}$ and $b > 1$, which is definitely not the case here. Or, it would have found that a and n are NOT co-prime to each other. In other words, Line 3 would have found that a and n share a factor or divisor other than 1, which again is not the case here. Hence, the only possibility in this sub-case is that AKS returns PRIME and so n is prime, as needed. □

(Sub-case 2): Assume AKS returns PRIME in step 6. In this sub-case, we will closely examine the 2 major steps performed in AKS, which is Lines 2 and 5. This part of the proof of correctness is much more involved and heavy as mentioned, so we will need to make use of Lemmas 3-8 that we presented in the section "Background information". So at this point, read through Lemma 3 to examine the first major step in this sub-case which is Line 2, and then come back here to continue with examining the second major step in this sub-case which is Line 5 of AKS. (Lemma 3 examines a property of r in relation to Line 2 of AKS, which is further used in Lemmas 4-8 then which prove some properties needed to examine Line 5 of AKS, which we will do below).

Let's examine Line 5 of AKS.
Since $o_r(n) > 1$, there must be a prime divisor p or n such that $o_r(p) > 1$. Also, $p > r$ since if this was not the case, then either Line 3 or Line 4 in AKS would be executed. So we know that $(n, r) = 1$ here (again, since otherwise Line 3 or 4 would be executed), where $p, n \in Z_r^*$, where p, n are fixed, and we let $l = \lfloor \sqrt{\phi(r)} \cdot \log(n) \rfloor$. So Line 5 verifies l equations, and in this sub-case we have that AKS does NOT output COMPOSITE, which means $((X + a)^n = X^n + a(\bmod X^r - 1, n))$ for every iteration of the loop. This means that

$$((X + a)^n = X^n + a(\mathrm{mod} X^r - 1, p)) \tag{16}$$

and applying Lemma 1 to this gives:

$$((X + a)^p = X^p + a(\mathrm{mod} X^r - 1, p)) \tag{17}$$

$((X+a)^p = X^p + a(\bmod X^r - 1, p))$, and combining equations (2) and (3) above, we get:

$$((X + a)^{\frac{n}{p}} = X^{\frac{n}{p}} + a(\mathrm{mod} X^r - 1, p)) \tag{18}$$

for every iteration of the loop in Line 5. This shows that both n and $\frac{n}{p}$ behave like the prime number p in this equation we just derived. This allows us to define

Lemma 4 (see Lemmas 4,5, and 6 in section "Background Information", and then come back here to continue on).

Now, consider the set $I = \{(\frac{n}{p})^i \cdot p^j \mid i, j \geqslant 0\}$, and the set of polynomials $P = \{\prod_{a=0}^{l}(X+a)^{e_a} \mid e_a \geqslant 0\}$. From Lemmas 5 and 6, we get that every number in I is introspective for every polynomial in the set P. We will now define 2 groups based on these sets I and P:

Group 1:
This is the set of all residues of numbers in I mod r. By residue we mean remainder, so whatever remainder you get by dividing an element in I with r, that is the residue. We will call this group G_1. G_1 is a subgroup of Z_r^* since we know have shown that $(n, r) = (p, r) = 1$. Let G_1 be the first group so that $|G_1| = t$. G is generated by n and p mod r since $o_r(n) > \log(n)^2$ and $t > \log(n)^2$.

To define Group 2, we will use the definition of Cyclotomic polynomials (see it in section "Background information", where we provide a definition for it).
Group 2:
Let $Q_T(X)$ be the rth cyclotomic polynomial over $\mathbb{F}_p$ (so the coefficients of $Q_T(X)$ are from the field $\mathbb{F}_p$). So $Q_T(X)$ divides $X^r - 1$ (by definition of cyclotomic polynomials) and factors into irreducible factors of degree $o_T(p)$. Let $h(X)$ be 1 such irreducible factor. Since $o_T(p) > 1$, the degree of $h(X)$ must also be greater than 1. This tells us that the second group, denoted by G_2, is the set of all residues of polynomials in P mod $h(X)$ and p. We generate G_2 by elements $X, X+1, X+2, ..., X+l$ in the field $\mathbb{F} = \mathbb{F}_p[X]/(h(X))$, and G_2 is a subgroup of the multiplicative group of $\mathbb{F}$.

We will now make use of Lemma 7, which defines a property of the cardinality of G_2, so refer to Lemma 7 in section "Background information", and then come back here to continue reading).

We will make use of a final lemma, Lemma 8, which will allow us to complete the proof of correctness (we will be able to prove the 2nd sub-case in ②). This lemma comes into play if n is not a power of p, in which case there is an upper bound for the number of elements in G_2. Again, refer to Lemma 8 in section "Background information", and then come back here to continue reading).

Now at this point we have all lemmas needed, some of which give us a lower and upper bound on the size of G, which means that we can finally prove the correctness of AKS in the 2nd sub-case of ②:

Suppose that AKS returns PRIME, as mentioned earlier in the beginning of the 2nd sub-case, we show that n must also be prime. Lemma 7 shows that for $|G_1| = t$, and $l = \lfloor\sqrt{\phi(r)} \cdot \log(n)\rfloor$, we have that:

$$
\begin{aligned}
|G_2| &\geqslant \binom{t+l}{t-l} \\
&\geqslant \binom{l+1+\lfloor\sqrt{t}\cdot\log(n)\rfloor}{\lfloor\sqrt{t}\cdot\log(n)\rfloor} \\
&\geqslant \binom{2\lfloor\sqrt{t}\cdot\log(n)\rfloor+1}{\lfloor\sqrt{t}\cdot\log(n)\rfloor} (\text{since } l \geqslant \lfloor\sqrt{t}\cdot\log(n)\rfloor) \\
&> 2^{\lfloor\sqrt{t}\cdot\log(n)\rfloor+1} (\text{since } \lfloor\sqrt{t}\cdot\log(n)\rfloor > \lfloor(\log(n))^2\rfloor \geqslant 1) \\
&\geqslant n^{\sqrt{t}}
\end{aligned}
$$

Now applying Lemma 8, we know that $|G_2| \leqslant n^{\sqrt{t}}$ if n is not a power of p. So $n = p^k$ for some $k > 0$. However, if we consider $k > 1$ (both ranges overlap for k), then Line 1 in AKS would have returned COMPOSITE, and we know that isn't the case here, which means that the only possibility here is that $k = 1$ and so $n = p$ thus showing that n is prime, as needed. So this completes the proof of sub-case 2 in ② and thus the overall proof of correctness of AKS, as needed. □

5 Show the algorithm is in P

In this section we will show how the AKS algorithm is in the class P, by analyzing its runtime. Long story short, the runtime of AKS is logarithmic, which is polynomial, which indicates that AKS is in P. To show how and why this is the case, we will prove that the runtime of AKS is $O(\log(n)^{\frac{21}{2}})$. So for a given input $n \in \mathbb{Z}$, $n > 1$, let's look at each line of the AKS algorithm and analyze its runtime, to come up with a final answer for the total runtime.

Line 1, which essentially determines if the input n is a perfect power or not, takes time $O(\log(n)^3)$.

In Line 2, we try to find an r such that $o_r(n) > \log(n)^2$. To do this, we could just try $r = 0, 1, 2, ...$ and check if $n^k \neq 1 \pmod r$ for every $k \leqslant \log(n)^2$. We are trying each r ($r = 0, 1, ...$ until we hit the particular value of r that we are working with) until we satisfy the equation we just mentioned, which means we will perform this check at most $\log(r)$ times, and for each value of r we try, checking if $n^k \neq 1 \pmod r$ will take at most $\log(n)^2$ multiplications modulo r, and so we get a runtime of $O(\log(n)^2 \cdot \log(r))$. However, recall from Lemma 3, we know that $r \leqslant \max\{3, \lceil\log(n)^5\rceil\}$, which means that the number of different values of r that we use to check if $n^k \neq 1 \pmod r$ is simply $\log(n)^5$). So we get the runtime of Line 2 to be:

$$
\begin{aligned}
O(\log(n)^2 \cdot \log(r)) &= O(\log(n)^2 \cdot \log(n)^5) \\
&= O(\log(n)^7)
\end{aligned}
$$

For Line 3, we are checking the GCD of n and every natural number up to and including r, so we check GCD r times, and checking GCD takes time $O(\log(n))$. So

Line 3 then takes time $O(r \cdot \log(n))$, and again by applying Lemma 3 just like we did for Line 2 previously, we get that the runtime for Line 3 simplifies to $O(\log(n)^6)$.

For Line 4, it takes $O(\log(n))$ time (the official proof of AKS states this, although some people agree that this operation usually takes constant time. In this case however, wether it takes time $\log(n)$ or constant time, the runtime of Line 5 which we will discuss below will dominate the runtime of Line 4 anyways).

For Line 5, the loop runs $\lfloor \sqrt{\phi(r)} \cdot \log(n) \rfloor$ times. For each iteration of the loop, every equation in it requires you to multiply degree r polynomials who have coefficients of size $\log(n)$, $\log(n)$ times. So every equation in the loop can be verified in $r \cdot \log(n)^2)$ steps, which means that the overall runtime of Line 5 is: $O(r \cdot \sqrt{\phi(r)} \cdot \log(n)^3)) = O(r^{\frac{3}{2}} \cdot \log(n)^2))$, and we apply Lemma 3 here, (in this case, $r^{\frac{3}{2}} \leqslant \max\{3^{\frac{3}{2}}, \lceil (\log(n)^5)^{\frac{3}{2}} \rceil\}$, so we replace $r^{\frac{3}{2}}$ with $(\log(n)^5)^{\frac{3}{2}})$, which gives us a runtime for Line 5 of:

$$O((\log(n)^5)^{\frac{3}{2}} \cdot \log(n)^3)) = O(\log(n)^{\frac{21}{2}}))$$

Notice that this runtime, for Line 5, is greater than the runtimes of all other steps, and so the runtime of AKS is $O(\log(n)^{\frac{21}{2}}))$ in the worst case. Since the runtime is polynomial, this clearly indicates that AKS is in P, as needed. □

6 Tracing output of specific number

In this section we will trace the output of AKS on a small number ($n = 2$) and a larger number ($n = 31$), so that you can get practical idea of how this algorithm really works.

Input: $2 \in \mathbb{N}$, $31 > 1$
Line 1: 2 is not a perfect power of any number $a \in \mathbb{N}$ and $b > 1$, so we go to line 2 of AKS.

Line 2: We need to find the smallest $r \in \mathbb{N}$, $r > 0$, such that $o_r(2) > \log(3)^2$. So let's try $r = 1, 2, 3$ (for each value of r we need to find the smallest positive integer k such that $2^k \equiv 1 \mod r$:

$r = 1$: Modular arithmetic is not well-defined in this case, so we discard this
$r = 2$: No value of k satisfies $2^k \equiv 1 \mod 2$ because every power of 2 is even (meaning $2^k \mod 2 = 0$ for all positive integers k) and $1 \mod 2 = 1$
$r = 3$: $k = 1$ doesn't work because $2^1 \not\equiv 1 \mod 3$, $k = 2$ also doesn't work since $2^2 \not\equiv 1 \mod 3$. However, $k = 3$ works as $2^3 \equiv 1 \mod 3$

So we get that $o_r(2) = o_3(2) = 3$ which is greater than $\log(2)^2 \approx 2.51$. So our smallest r is 3 for Line 2. Let's move onto Line 3 of AKS.

Line 3: So now for all values $a \leqslant r$, $a \geqslant 1$, we check to see if a and n are co-prime. In other words, check gcd(a, n) and if it is greater than 1 or less than n, we output COMPOSITE in AKS. So $r = 3$ is what we got earlier, which means the values of a we need to check are $a = 1, 2, 3$. So let's compute the GCD of each a with n:

$$\gcd(1, 2) = 1$$
$$\gcd(2, 2) = 2$$
$$\gcd(3, 2) = 1$$

All of the gcd values above are not in the range indicated in Line 3 of AKS (which is $1 < (a, 2) < 2$) which means that we don't output COMPOSITE in Line 3. So we move onto Line 4 of AKS.

Line 4: Here we can see that $n = 2$ and $r = 3$, and clearly $n \leqslant r$, which means that we output PRIME in AKS and terminate the algorithm. Hence, $n = 2$ is prime, as needed.

Input: $3 \in \mathbb{N}$, $31 > 1$ We will now demonstrate a quick example of running AKS on an input, which we will choose to be $n = 31$. Let us trace through each step of the algorithm:

Input: $31 \in \mathbb{N}$, $31 > 1$.

Line 1: Check if 31 is a perfect power, so if $31 = a^b for some a \in \mathbb{N}$ and $b > 1$. This is clearly not the case, so we move to Line 2.

Line 2: We calculate $o_r(31)$. By lemma 3, $r \leqslant \max\{3, \lceil \log(31)^5 \rceil = 2985\} = 2985$. So for each natural number up to and including $r = 2985$, we try to find the smallest $k > 0$ such that $31^k = 1 \pmod r$. This will take a very long time of course, however, we eventually realize that $r = 29$ should be chosen. Notice how ($r = 29$, $n = 31$) $= 1$. Now we want to find the smallest number k such that $31^k = 1 \pmod{29}$. So we try $k = 1, 2, 3, ...$ which again will take a long time, so we eventually get to $k = 28$ meaning $31^2 8 = 1 \pmod{29}$. So $o_2 9(31) = 28$, and this is greater than $\log(31)^2 = 25$). So we have chosen $r = 29$, and we move to Line 3 of AKS.

Line 3: Now for each $a = 2, ..., 29$, we check if $1 < (a, 31) < 31$. So in other words, compute the GCD of a and $n = 31$, and if you get a number that is greater than 1 and less than $n = 31$, then we return COMPOSITE. So we try all values of a, and realize that none of those values share a factor with 31, hence we continue to Line 4 of AKS.

Line 4: Now we check if $n \leqslant r$, in this case it is obviously false as 31 is not less than or equal to 29, so we move on to Line 5 of AKS.

Line 5: First for the number of for loop iterations, let us compute the value of $\lfloor \sqrt{\phi(r)} \cdot \log(n) \rfloor = \lfloor \sqrt{\phi(29)} \cdot \log(31) \rfloor$. So $\phi(29) = 28$, because all numbers from 1 to 28 (both inclusive) are co-prime with 29. So for the number of loop iterations, we get: $\lfloor \sqrt{\phi(29)} \cdot \log(31) \rfloor = 26$. So the loop runs for $a = 1, 2, ..., 26$. Now we

focus on the body of the loop. So for each iteration, $(X + a)^3 1$ is a very large polynomial when expanded (can be expanded via Binomial Theorem), same for $X^3 1$. Now we compute $(X^3 1 + a \bmod X^2 9 - 1) \bmod 31$, which is very time consuming to compute, so we will just put the answer here, which is $a^3 1 + X^2$. Then we compute $X^3 1 + a \bmod X^2 9 - 1$ which is $a + X^2$. Now we subtract this from the earlier calculation, so we do: $a^3 1 + X^2$ - $(a + X^2) = a^3 1 - a$. So we check if this value is equal to 0 in each iteration of the loop, which it is not. Hence, Line 5 never returns COMPOSITE, so we move on to Line 6 of AKS.

Line 6: If we made it to this line, that means $n = 31$ is prime, and thus, AKS returns PRIME, as needed.

7 Improvements

In this section, we will discuss some improvements that can be made to speed up the AKS algorithm, as suggested by its original authors. One thing to note is that the improvements that we will discuss are not final, as in, they don't work any general input to AKS. There is still work being done towards speeding up AKS, and so the improvements are showing what progress has been made towards speeding up the algorithm, as the improvements are still being refined.

As discussed in section 5, the runtime of AKS is $O(\log(n)^{\frac{21}{2}})$, however, there are ways to improve the runtime, which relies on the value of r that we find in Line 2 of AKS. One of the ways is that if $r = O(\log(n)^2)$, then this would bring the runtime of AKS down to $O(\log(n)^6)$. This is the best case scenario, and there are 2 conjectures we will present below that argue that such a value of r can indeed exist.

Artin's Conjecture:
This conjecture argues that given any $n \in \mathbb{N}$, such that n is NOT a perfect square, the number of primes $q \leqslant m$ (m is any number) where $o_q(n) = q-1$ is $A(n) \cdot \frac{m}{\ln m}$, where $A(n)$ is known as Artin's constant, and $A(n) > 0.35$.

Sophie-Germain Prime Density Conjecture:
This conjecture argues that the number of primes $q \leqslant m$ (again, m is any number) where the number $2q + 1$ is also prime is $\frac{2C_2 m}{(\ln m)^2}$, where C_2 is the twin prime constant which is equal to approximately 0.66. So the number of primes q that have this property are called Sophie-Germain primes.

So essentially, the above 2 conjectures give an estimate for the number of prime numbers in each given scenario. There has been some progress put forth towards proving these 2 conjectures.

If Artin's conjecture can become effective for $m = O(\log(n)^2)$, it can then be proven directly that there exists an $r = O(\log(n)^2)$ such that it satisfies the conditions of the conjecture. Again, there is still progress being made towards proving

this conjecture, and it is also known that it holds under Generalized Riemann Hypothesis.

For the second conjecture, we mention "density of primes", which simply describes the frequency or distribution of prime numbers. If you have a high density of prime numbers, that means you frequently see prime numbers, and if you have a low density of prime numbers, that means you see prime numbers less frequently. So for the second conjecture, if it can be proven to hold true, we can then prove that $r = O(\log(n)^2)$ approximately. Now, by density of Sophie-Germain primes, there must exist at least $\log(n)^2$ such primes between $8\log(n)^2$ and $c\log(n)^2 \cdot (\log(\log(n)))^2$, where c is some constant. So for any such prime q, we have 2 cases for the value of $o_q(n)$: either $o_q(n) \leqslant 2$, or $o_q(n) \geqslant \frac{q-1}{2}$. Any value of q such that $o_q(n) \leqslant 2$ must also divide $n^2 - 1$ evenly in which case the number of such a value of q would be at maximum $O(\log(n))$. This would then imply that there must be a prime $r = O(\log(n)^2)$ (approximately) such that $o_r(n) > \log(n)^2$. This value of r would then allow AKS to have a runtime of roughly $O(\log(n)^6)$. There has been some more progress put forth to proving this conjecture, will we will discuss below.

Let $P(m)$ represent the greatest prime divisor of m. It was proven that primes q with $P(q-1) > q^{\frac{1}{2}+c}$ (c is a constant roughly equal to $\frac{1}{12}$), occurs with positive density. This means we have a certain number of prime numbers, q, with the specified property of $P(q-1)$ that was just mentioned, such that the specified property always holds true when you have a list of numbers with a high frequency of prime numbers. Now, we present another Lemma, Lemma 9, which improves this idea, so see section "Background information" for Lemma 9.

References

- `https://www.cse.iitk.ac.in/users/manindra/algebra/primality_v6.pdf/`
  This link is probably the most useful as it is the official paper of the algorithm. It contains essentially everything needed about the AKS Primality Test, like background info needed to understand algorithm, proof of correctness, runtime, and improvements that can be made to the algorithm to make it run faster.
- `https://en.wikipedia.org/wiki/AKS_primality_test`
  This link also contains some more info about the algorithm, its runtime, and a useful example of tracing the output of the algorithm on a specific input.
- `https://t5k.org/prove/prove4_3.html`
  . This link contains some more info about proving the correctness of the algorithm as well. It also contains explanation of background info needed to understand the algorithm, like Fermat's Little Theorem.

- https://en.wikipedia.org/wiki/Cyclotomic_polynomial
  For definition of cyclomotic polynomials.
- https://testbook.com/maths/fermats-little-theorem
  This link provides a very good high level overview, with many details as well, of Fermat's Little Theorem.
- https://books.google.ca/books?id=fERohM1sPagC&newbks=1&newbks_redir=0&lpg=PP1&pg=PA131#v=onepage&q&f=false
  This source provides more insight into Fermat's Little Theorem, including a proof of it, which is included in this report in the Background Information section.
- https://byjus.com/maths/groups/
  This link is used to define Groups.
- https://dept.math.lsa.umich.edu/~jchw/2015Math110Material/FieldAxioms-Math110-W2015.pdf
  This link is used to define Fields.