Efficient Primality-Tests

Seminar Paper

Author: Lars Reimers <reimers.lars@hotmail.de>

Date: 27.02.2019

Lecturer: Prof. Dr. Sebastian Iwanowski

Contents

Abstract

This paper is part of a seminar about mathematical applications. It is supposed to give an overview of efficient primality testing by the example of the so-called AKS primality test. Most information is taken directly from the original work by Agrawal, Kayal and Saxena [1] with the formulas being explained more detailed and steps that have been skipped or merged in the original work being stretched out. Therefore the structure of this paper is oriented by their structure and some formulations may be copied since there was nothing to add to them. Cases where these formulations exceed single sentences have been explicitly marked.

Definition of Primality

The most common definition of primality refers to a number having no divisors besides 1 and the number itself. Formally this can represented as:

Let $p \in \mathbb{N}$, $p > 1$, then p is *prime* if:

$$\forall n, n \in \mathbb{N}, n > 1, n < p : n \nmid p \tag{1}$$

Another possible definiton is, that a prime can not be represented as the product of two natural numbers. This is equivalent to the above definiton but leads us closer to the term *composite number* which refers to any number that is not a prime and therefore composable by multiplying two natural numbers. This term will be commonly used in this paper and referenced papers.

Complexity Classes and Notation

Complexity classes are an abstract measurement to declare the rate of growth of a computation with increasing size of the input. These classes are commonly notated in the*Big O Notation*. The given term gives an estimate of the growth of computation efficiency and is conventionally melted down to the highest factoring subterm since the other subterms do not have an influence with input size nearing infinity. Computation efficiency can here refer to multiple factors, like computation time, used space in memory or used amount of an ‘expensive’ operation.

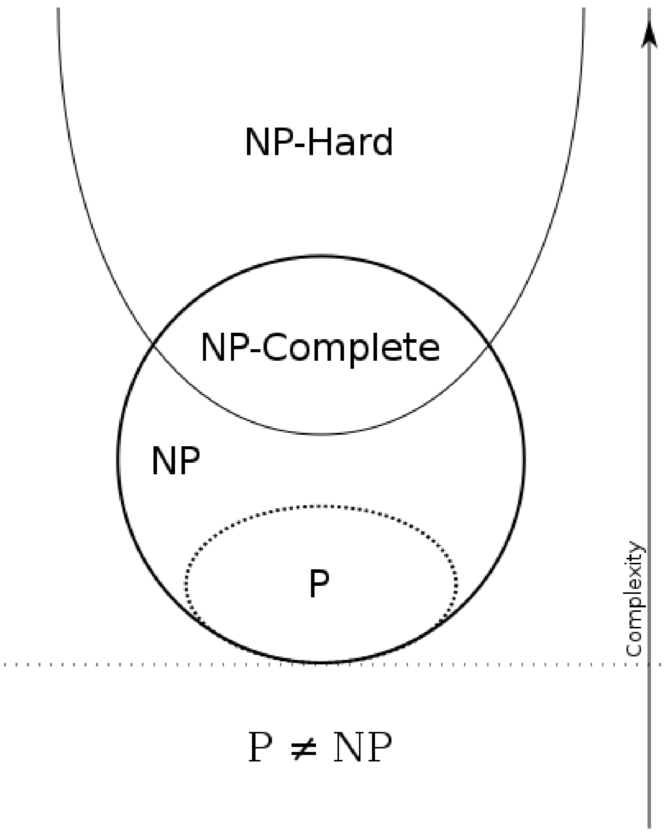
Common time complexities are:

Time Complexity	Big O Notation	Example
constant time	$O(1)$	Find the first item of a list
linear time	$O(n)$	Find the smallest item in an unsorted list
exponential time	$O(2^{p(n)})$	Many games, including Go with the Japanese ko-rules
factorial time	$O(n!)$	Solving the traveling salesman problem

Common complexity classes are:

Name	Description	Example
P	polynomial time	Checking if a number is prime
NP	nondeterministic polynomial time	Finding a Hamiltonian path or cycle
NP-hard	”at least as hard as the hardest problems in NP”	Subset sum problem
NP-complete	intersection between NP and NP-hard	Finding a Hamiltonian path or cycle

The connection between these complexity classes can be shown as follows:



From: `https://en.wikipedia.org/wiki/File:P_np_np-complete_np-hard.svg`

Soft O

In most primality tests the so called *soft-O*-notation is used. This notation ignores logarithmic factors for they become irrelevant with the typical present growth rates. It is shorthand for:

$$\widetilde{O}(g(n)) = O(g(n) * log^k g(n)) \text{ for some } k \tag{2}$$

Trivial Tests

4.1 Wilson's Theorem

Wilson's Theorem states that:

$$(n-1)! \equiv -1(\text{mod } n) \text{ if and only if } n \text{ is } \textit{prime} \tag{3}$$

Using this algorithm needs $(n-2)$ multiplications, leaving us with a time complexity of $O(n)$. Since we want to connect the time complexity to the size of the input, we have to take that into account. For a number n it requires $log_2\, n$ bits to represent it, leading to a time complexity of $O(2^{log_2\, n})$ which is an exponential time complexity.

Proof
The proof is trivial for $n = 2$.

For all $n \geq 3$, assume that there is some prime number q with $2 \leq q \leq n-2$ which divides n. If $(n-1)!$ is congruent to $-1(\text{mod } n)$ then it is also congruent to $-1(\text{mod } q)$. Since $(n-1)!$ contains a term equivalent to q, it is congruent to $0(\text{mod } q)$. This contradiction proves, that the above equation is not true for any composite number.

4.2 Trial Division

Given the original definition we gave in Section 2 the most straightforward primality test is trial division. For this we divide the number to check n with every number q with $2 \leq q \leq n-1$. Using this needs at most $(n-2)$ divisions and leads to the same time complexity as Wilson's Theorem of $O(2^{log_2\, n})$. This algorithm is more efficient for composite numbers, since we can quit, once we found a divisor.

Optimization
The above algorithm can be further optimized by limiting q to $2 \leq q \leq \lceil\sqrt{n}\rceil$ since if n has a divisor d with $d > \lceil\sqrt{n}\rceil$ we would still find the complement of d with n/d which then has to be smaller than $\lceil\sqrt{n}\rceil$.

This leads to at most $\lceil\sqrt{n}\rceil$ divisions, resulting in the same complexity class as before since constant factors are ignored. The average runtime is reduced though.

Fast deterministic Tests

5.1 Preparation: Fermat's Little Theorem

Fermat's little theorem states that:

$$a^p \equiv a(\text{mod } p) \tag{4}$$

It can be proven through induction with the anchor being $0^p \equiv 0(\text{mod } p)$. To prove that the theorem holds for $a = k + 1$ if it's true for $a = k$, we define the following lemma for any integers x and y and prime p:

$$(x + y)^p \equiv x^p + y^p(\text{mod } p) \tag{5}$$

Combined with the binomial theorem, the left half of Equation (5) expands to:

$$\begin{aligned} &\sum_{i=0}^{n} \left(\binom{n}{i} * x^{n-i} * y^i \right) && (\text{mod } p) \\ \Leftrightarrow &\sum_{i=1}^{n-1} \left(\binom{n}{i} * x^{n-i} * y^i \right) + x^p + y^p && (\text{mod } p) \end{aligned} \tag{6}$$

Since all terms of $\binom{n}{i}$ contain a prime factor n if n is prime and are therefore $0(\text{mod } p)$, the term is equivalent to $x^p + y^p(\text{mod } p)$ and the lemma is therefore proven. Further information on why this holds can be found in the section about AKS's lemma 2.1.

The induction step assumes that $k^p \equiv k(\text{mod } p)$. Now we consider $k + 1$:

$$(k + 1)^p \equiv k^p + 1^p(\text{mod } p) \tag{7}$$

Given the induction hypothesis, $k^p \equiv k(\text{mod } p)$ and $1^p \equiv 1(\text{mod } p)$:

$$\begin{aligned} &(k + 1)^p \equiv k^p + 1^p(\text{mod } p) \\ \Leftrightarrow &(k + 1)^p \equiv k + 1(\text{mod } p) \end{aligned} \tag{8}$$

This is the resulting statement for $a = k + 1$ and therefore concludes the proof for Fermat's little theorem.

5.2 Agrawal-Kayal-Saxena Test

This test was proposed by Manindra Agrawal, Neeraj Kayal and Nitin Saxena (from now on referred to as AKS) in their paper *PRIMES is in P.* [1]

AKS's lemma 2.1 states:

Let $a \in \mathbb{Z}, n \in \mathbb{N}, n \geq 2$ and $(a, n) = 1$. Then p is prime if and only if:

$$(X + a)^n = X^n + a(\text{mod } n) \tag{9}$$

First to explain some of the used symbols: $(a, n) = 1$ represents the greatest common divisor of a and n being 1, showing that a and n are coprime. X is to be seen as a variable.

The proof bases on first moving the right term to the left side of the equation:

$$\begin{aligned} &(X + a)^n - (X^n + a) &&= 0(\text{mod } n) \\ \Leftrightarrow\ &(X + a)^n - X^n - a &&= 0(\text{mod } n) \end{aligned} \tag{10}$$

The minuend can be expanded to:

$$\begin{aligned}&(X+a)^n\\ \Leftrightarrow&\sum_{i=0}^{n}\left(\binom{n}{i}*a^{n-i}*X^i\right)\\ \Leftrightarrow&X^n+a^n+\sum_{i=1}^{n-1}\left(\binom{n}{i}*a^{n-i}*X^i\right)\end{aligned}\tag{11}$$

Combined with the subtrahent we have:

$$\begin{aligned}&X^n+a^n+\sum_{i=1}^{n-1}\left(\binom{n}{i}*a^{n-i}*X^i\right)-X^n-a\\ \Leftrightarrow\quad&(a^n-a)+\sum_{i=1}^{n-1}\left(\binom{n}{i}*a^{n-i}*X^i\right)\end{aligned}\tag{12}$$

Because of modular arithmetic we know that:

$$\begin{aligned}&a+b\equiv 0(\text{mod } k)\\ \Leftrightarrow\quad&a\equiv 0(\text{mod } k)\wedge b\equiv 0(\text{mod } k)\end{aligned}\tag{13}$$

and therefore we can view the summands of Equation (12) separately.

There are two scenarios: either n is prime or n is composite.

n is prime
The left summand is trivially proven with *Fermat's little theorem* since

$$a^n - a = 0(\text{mod } n) \Leftrightarrow \quad a^n = a(\text{mod } n) \tag{14}$$

which is true if n is prime.

For the right summand we can focus on the coefficents, namely $\binom{n}{i}$ with $0 < i < n-1$.

$$\begin{aligned} &\binom{n}{i} \\ =\ & \frac{n * (n-1)(n-2)\cdots(n-(k-1))}{k(k-1)(k-2)\cdots 1} \\ =\ & n * \frac{(n-1)(n-2)\cdots(n-(k-1))}{k(k-1)(k-2)\cdots 1} \end{aligned} \tag{15}$$

Since n is presumabely prime, the fraction will still result in a natural number and therefore we know that $\binom{n}{i} \equiv 0(\text{mod } n)$. Given that

$$\begin{aligned} & a \equiv 0(\text{mod } n) \vee b \equiv 0(\text{mod } n) \\ \rightarrow\ & a * b \equiv 0(\text{mod } n) \end{aligned} \tag{16}$$

we know that the whole term for every part of the sum, $\left(\binom{n}{i} * a^{n-i} * X^i\right)$, is 0 for all values of i. Together with Equation (14) we now have a sum of terms that are all $\equiv 0(\text{mod } n)$ and according to Equation (13) that leads to the whole Equation (12) being $\equiv 0(\text{mod } n)$. With this we've shown that lemma 2.1 proposed by AKS holds if n is prime.

n is composite
Assume there is a prime q and an integer k with q^k dividing n where k is as high as possible, then $n = \lambda * q^k$ with $\lambda \in \mathbb{N}$. It is enough to take a look at the term for X^q which is $\binom{n}{q} * a^{n-q} * X^q$.

We can show that $q^k \nmid \binom{n}{q}$ via contradiction by assuming there is a $r \in \mathbb{N}$ with $r * q^k = \binom{n}{q}$:

$$\begin{aligned} r * q^k &= \binom{n}{q} \\ \Leftrightarrow r * q^k &= \frac{n * (n-1)(n-2)\cdots(n-(q-1))}{q(q-1)(q-2)\cdots 1} \\ & \text{divide both sides by } q^k \\ \Leftrightarrow r &= \frac{\lambda * (n-1)(n-2)\cdots(n-(q-1))}{q(q-1)(q-2)\cdots 1} \end{aligned} \tag{17}$$

By definition $q \nmid \lambda$ and $!\exists q$ with $q|x$ and x in $[n-1 \cdots n-q+1]$. Therefore the fraction can not result in an integer and $r \notin \mathbb{N}$ and we've shown that $q^k \nmid \binom{n}{q}$

We can also show that $q^k \nmid a^{n-q}$ since $q \nmid a \rightarrow q \nmid a^{n-q} \rightarrow q^k \nmid a^{n-q}$ and *qnmida* can be trivially shown since:

$$q|n \text{ and } (a, n) = 1 \rightarrow q \nmid a \tag{18}$$

Therefore $\binom{n}{q} * a^{n-q} * X^q$ will not be $0(\text{mod } n)$ and Equation (9) is not fulfilled by composite numbers, as stated above.

Calculating this would take at least exponential time and is therefore not the solution to determine primality in a fast manner.

Adaption of the equation

AKS propose a modified version of Fermat's little theorem for polynoms which is calculatable in polynomial time but can not detect pseudoprimes:

Let $a \in \mathbb{Z}, n \in \mathbb{N}, n \geq 2$ and $(a, n) = 1$. Then p is prime if and only if:

$$\begin{aligned} &(X+a)^n &&= X^n + a (\text{mod } X^r - 1, n) \\ \Leftrightarrow &(X+a)^n - X^n - a &&= 0 (\text{mod } X^r - 1, n) \\ \Leftrightarrow &(X+a)^n - X^n - a &&= (0 (\text{mod } X^r - 1))(\text{mod } n) \end{aligned} \tag{19}$$

This is proven for primes the same way as above. As stated, some composites will also fulfill this equation and therefore additional checks need to be added to the algorithm.

The Algorithm

As a note, all references to *log* mean the logarithm to base 2 and all references to *ln* mean the natural logarithm (to base e).

```
Input: integer $n > 1$.

1. If $n = a^b$ for $a \in \mathbb{N}$ and $b > 1$, output COMPOSITE.
2. Find the smallest $r$ such that $o_r(n) > log^2 n$.
3. If $1 < (a, n) < n$ for some $a \leq r$, output COMPOSITE.
4. If $n \leq r$, output PRIME.
5. For $a = 1$ to $\lfloor \sqrt{\phi(r)} log\ n \rfloor$ do
        if $((X+a)^n \neq X^n + a (\text{mod } X^r - 1, n))$, output COMPOSITE;
6. Output PRIME.
```

where

$o_r(a)$ describes the smallest number k such that $a^k = 1 (\text{mod } r)$ with $a \in \mathbb{Z}, r \in \mathbb{N}, (a, r) = 1$ and

$\phi(r)$ for $r \in \mathbb{N}$ describes the euler totient function which is equal to the number of numbers less than r that are coprime to r.

The proof bases on two lemmas:

- *If n is prime, the algorithm returns PRIME*
- *If the algorithm returns PRIME, n is prime*

If n is prime, the algorithm returns PRIME

- In step 1, if n is prime, the algorithm doesn't output COMPOSITE.
- In step 2, the algorithm doesn't output anything.
- In step 3, if n is prime, the algorithm doesn't output COMPOSITE.
- In step 5, if n is prime, the algorithm doesn't output COMPOSITE.

So if n is prime the algorithm outputs PRIME in either step 4 or in step 6.

If the algorithm returns PRIME, n is prime

“If n is prime then steps 1 and 3 can never return COMPOSITE. By [Equation (10)], the for loop also cannot return COMPOSITE. Therefore the algorithm will identify n as PRIME either in step 4 or in step 6.” [1]

If step 4 outputs PRIME then $n \leq r$ so all possible factors would've been found in step 3 since all integers $\leq r$ are tested for coprimality to n.

The only remaining step in which PRIME can be output is step 6. For the proof it is assumed that step 6 outputs PRIME and that n is a composite number since the proof is based on building a contradiction.

As Nair has already shown in [2], if $LCM(m)$ denotes the least common multiple of the first m natural numbers:

$$LCM(m) \geq 2^m \text{ for } m \geq 7 \tag{20}$$

The original work defines an upper bound for the r defined in the algorithm:

$$\exists r \text{ with } r \leq max\{3, \lceil log^5 n \rceil\} \text{ such that } o_r(n) > log^2 n \tag{21}$$

This is trivially shown for $n = 2$ since $r = 3$ satisfies all conditions. Therefore we can assume $n > 2$ meaning $\lceil log^5 n \rceil > 10$ and the lemma in Equation (20) always applies. We define $B = \lceil log^5 n \rceil$ and by that definiton the largest value of k for $m^k \leq B$ with $m \geq 2$ is $\lfloor logB \rfloor$.

To prove the upper bound for r consider $r_1, r_2, \cdots, r_t$ being all integers that either fulfill $o_{r_i}(n) \leq log^2 n$ or divide n. All of these numbers divide the product:

$$n * \prod_{i=1}^{\lfloor log^2 n \rfloor} (n^i - 1) \text{ which is } \leq n^{log^4 n} \leq 2^{log^5 n} \tag{22}$$

Taking into account that the lcm of the first $\lceil log^5 n \rceil$ numbers is atleast $2^{\lceil log^5 n \rceil}$ and the above product is smaller than $2^{\lceil log^5 n \rceil}$ there must exist a number $s \leq \lceil log^5 n$ that is not equal to any of $\{r_1, r_2, \cdots, r_t\}$.

If that s is coprime to n then $o_s(n) > log^2 n$ since otherwise it would've been a factor of Equation (22) and therefore we found a value fulfilling our demands and define $r = s$.

If s and n are not coprime, there exists a number $r = \frac{s}{(s,n)}$ that is not in $\{r_1, r_2, \cdots, r_t\}$ since s is by definition not a factor of n and so (s, n) must be in $\{r_1, r_2, \cdots, r_t\}$.

With this we've proven the existence of a number r that satisfies the given borders.
Additionally we will define a prime divisor p of n such that $o_r(p) > 1$. p must exist since $o_r(n) > 1$ and must be greater than r since otherwise step 3 or 4 would've determined already whether n is prime. Since both n and p are coprime to r, we know that both numbers are in Z_r^*. For the rest of this proof we will assume r and p to be fixed and additionally define $\ell = \lfloor \sqrt{\phi(r)} log\ n \rfloor$.

AKS define the term *introspective* for a polynomial $f(X)$ and a number $m \in \mathbb{N}$ such that:

$$[f(X)]^m = f(X^m)(\text{mod } X^r - 1, p) \tag{23}$$

For introspectiveness two rules are defined:
"If m and m' are introspective numbers for $f(X)$ then so is $m * m'$"
"If m is introspective for $f(X)$ and $g(X)$ then it is also introspective for $f(X) * g(X)$" (both [1])

The introspective equation for m and $f(X)$ is:

$$\begin{aligned} [f(X)]^m &= f(X^m)(\text{mod } X^r - 1, p) \\ \Leftrightarrow [f(X)]^{m*m'} &= [f(X^m)]^{m'}(\text{mod } X^r - 1, p) \end{aligned} \tag{24}$$

by replacing X with X^m in the introspective equation for m' we get:

$$[f(X^m)]^{m'} = f(X^{m*m'})(\text{mod } X^{m*r} - 1, p) \tag{25}$$

and since $X^r - 1$ divides $X^{m*r} - 1$:

$$[f(X^m)]^{m'} = f(X^{m*m'})(\text{mod } X^r - 1, p) \tag{26}$$

Putting the two equations together we get:

$$[f(X))]^{m*m'} = f(X^{m*m'})(\text{mod } X^r - 1, p) \tag{27}$$

proving the first rule.
The second rule is trivially shown since:

$$[f(X) * g(X)]^m = [f(X)]^m * [g(X)]^m = f(X^m) * g(X^m)(\text{mod } X^r - 1, p) \tag{28}$$

Under the assumption that step 5 of the algorithm doesn't output COMPOSITE we can deduct:

$$(X + a)^n = X^n + a(\text{mod } X^r - 1, n) \tag{29}$$

for $0 \leq a \leq \ell$. Since p divides n, this implies (all following equations are valid for $0 \leq a \leq \ell$):

$$(X + a)^n = X^n + a(\text{mod } X^r - 1, p) \tag{30}$$

Because of Equation (19) this also implies:

$$(X + a)^p = X^p + a(\text{mod } X^r - 1, p) \tag{31}$$

and

$$(X + a)^{\frac{n}{p}} = X^{\frac{n}{p}} + a(\text{mod } X^r - 1, p) \tag{32}$$

These equations lead to the conclusion that both $\frac{n}{p}$ and p are introspective for $X + a$ with the bounds for a being $0 \leq a \leq \ell$.

With the defined properties of introspectiveness we can define two sets for the further proof; a set I with:

$$I = \left\{ \left(\frac{n}{p}\right)^i * p^j \middle| i, j \geq 0 \right\} \tag{33}$$

and a set P with:

$$P = \left\{ \prod_{a=0}^{\ell} (X + a)^{e_a} \middle| e_a \geq 0 \right\} \tag{34}$$

and state for every polynomial P_i in P that every number in I is introspective to P_i. This can be seen easily with the above rules for introspectiveness being transformed into (let $\multimap$ denote introspectiveness):

$$m \multimap f(X) \rightarrow m^k \multimap f(X) \text{ for any } k \in \mathbb{N} \tag{35}$$

and

$$m \multimap f(X) \rightarrow m \multimap f(X)^k \text{ for any } k \in \mathbb{N} \tag{36}$$

Out of the above sets we construct two groups. The first group is G and contains the set of all residues of numbers in I modulo r. We define the size of G as t. Because we know that $o_r(n) > log^2 n$, t also has to be greater than $log^2 n$ since otherwise it wouldn't contain an element which is $1 (\text{mod } r)$.
For the second group we need to introduce cyclotomic polynomials. "The nth cyclotomic polynomial, for any positive integer n, is the unique irreducible polynomial with integer coefficients that is a divisor of $x^n - 1$ and is not a divisor of $x^k - 1$ for any $k < n$." (From: `https://en.wikipedia.org/wiki/Cyclotomic_polynomial`)

This leads to the nth cyclotomic polynomial being:

$$\Phi_n(x) = \prod_{1 \leq k \leq n \text{ with } gcd(k,n)=1} \left(x - e^{2i\pi \frac{k}{n}} \right) \tag{37}$$

AKS use the r^{th} cyclotomic polynomial over F_p and call it $Q_r(X)$. Combined with the work of [3] they also define $h(X)$ being an irreducible factor of $Q_r(X)$ of degree $o_r(p)$ and therefore greater than one.

From this they define the second group $\mathcal{G}$ being "the set of all residues of polynomials in P modulo $h(X)$ and p". [1]

$\mathcal{G}$ can be described by the elements $X, X+1, X+2, \cdots, X+\ell$ over the Field $F = F_p[X]/(h(X))$ and is therefore a subgroup of the multiplicative group of F.
Hendrik Lenstra Jr. has shown a lower bound on the size of $\mathcal{G}$: [4]

$$|\mathcal{G}| \geq \binom{t+\ell}{t-1} \tag{38}$$

This lower bound is an improvement on a bound defined in an earlier version of AKS's paper. AKS continue to define an upper bound for the size of $\mathcal{G}$ for the case that n is not a power of p:

$$|\mathcal{G}| \leq n^{\sqrt{t}} \text{ if } n \text{ is not a power of } p \tag{39}$$

To prove this we define a subset of I with:

$$\hat{I} = \left\{ \left(\frac{n}{p}\right)^i * p^j \middle| 0 \leq i, j \leq \lfloor \sqrt{t} \rfloor \right\} \tag{40}$$

Assuming n is not a power of p this set has $(\lfloor\sqrt{t}+1\rfloor)^2$ elements which is greater than t. Because we defined t to be the size of $\mathcal{G}$, we know there must be at least two numbers in $\hat{I}$ equal modulo r. We call these m_1 and m_2 with $m_1 < m_2$. With a $f(X) \in P$ we can find that:

$$[f(X)]^{m_1} = [f(X)]^{m_2} \text{ in } F \tag{41}$$

This leads to the conclusion that $f(X) \in \mathcal{G}$ and $f(X)$ is therefore a root of $Q'(Y) = Y^{m_1} - Y^{m_2}$ in F. $Q'(Y)$ has at least $|\mathcal{G}|$ distinct roots in F since $f(X)$ is in $\mathcal{G}$. The degree of $Q'(Y)$ is m_1 and greater than $|\mathcal{G}|$. Since $m_1 \leq \left(\frac{n}{p} * p\right)^{\sqrt{t}} \leq n^{\sqrt{t}}$, $|\mathcal{G}| \leq n^{sqrtt}$. This proof was formulated by [5] and sadly no further explained by AKS.
To now prove the algorithm being correct we still assume that step 6 outputs PRIME. We know the lower bound of $|\mathcal{G}|$ and by replacing ℓ we can transform it:

$$\begin{aligned}|\mathcal{G}| \geq &\binom{t+\ell}{t-1} \\ &\text{since } t > log^2 n \rightarrow t > \sqrt{t}\log n \\ &\geq \binom{(\lfloor\sqrt{t}\log n\rfloor + 1) + \ell}{\lfloor\sqrt{t}\log n\rfloor} = \binom{\ell + \lfloor\sqrt{t}\log n\rfloor + 1}{\lfloor\sqrt{t}\log n\rfloor} \\ &\text{since } \ell = \lfloor\sqrt{\phi(r)}\log n\rfloor \text{ and } \phi(r) \geq t \text{ and therefore } \ell \geq \lfloor\sqrt{t}\log n\rfloor \\ &\geq \binom{2\lfloor\sqrt{t}\log n\rfloor + 1}{\lfloor\sqrt{t}\log n\rfloor} \\ &\text{this step is a little more complicated and will be discussed below} \\ &> 2^{\lfloor\sqrt{t}\log n\rfloor + 1} \\ &\geq n^{\sqrt{t}}\end{aligned} \tag{42}$$

The inbetween step can be explained with some knowledge about binomial coefficients. Following we'll replace $\lfloor\sqrt{t}\log n\rfloor$ with x leading to:

$$\binom{2x+1}{x} \tag{43}$$

Since we know that $1 \leq x \leq 2x+1$ we also know that:

$$\binom{2x+1}{x} = \binom{2x}{x} + \binom{2x}{x-1} \tag{44}$$

Since $\binom{n}{k} \geq \frac{n}{k}^k$:

$$\begin{aligned}\binom{2x+1}{x} &\geq 2^x + \binom{2x}{x-1} \\ &\geq 2^x + \frac{2^x x^x}{(x-1)^x} \\ &> 2^{x+1}\end{aligned} \tag{45}$$

To show that $2^x + \frac{2^x x^x}{(x-1)^x} > 2^{x+1}$:

$$\begin{aligned} 2^x + & \frac{2^x x^x}{(x-1)^x} > 2^{x+1} \\ \Leftrightarrow 2^x + & \frac{2^x x^x}{(x-1)^x} > 2 * 2^x \\ \Leftrightarrow \quad & \frac{2^x x^x}{(x-1)^x} > 2^x \\ \Leftrightarrow \quad & 2^x x^x > 2^x * (x-1)^x \\ \Leftrightarrow \quad & x^x > (x-1)^x \\ \Leftrightarrow \quad & x > x-1 \\ \Leftrightarrow \quad & 1 > 0 \end{aligned} \tag{46}$$

These steps are all possible since $x = \lfloor \sqrt{t} \log n \rfloor \rightarrow x \geq 1$.
With this we've shown that $|\mathcal{G}| > n^{\sqrt{t}}$ (take into account the single greater-relation that is not a greater-equals-relation in the above transformation). Since we've shown before that $|\mathcal{G}| \leq n^{\sqrt{t}}$ if n is not a power of p, n has to be a power of p for this to not contradict. If this were the case, the algorithm would've already returned COMPOSITE in step 1 and therefore $n = p$ and we've shown that n is indeed prime.

Time complexity
The first step takes asymptotic time $O^\sim(log^3 n)$. Calculation for this can be found in the work of [6].

In step 2 we need to test values for $k \leq log^2 n$ for any r which involves at most $O(log^2 n)$ multiplications modulo r equaling a time of $O^\sim(log^2 n \log r)$. Since we need to try a maximum of $log^5 n$ r's the total time of step 2 evaluates to $O^\sim(log^7 n)$. Note that the '$\log r$'-term is omitted out of insignificance.

The third step of the algorithm computes the greatest common divisors of r numbers. According to [6] each gcd-computation takes $O(\log n)$ and because of the upper bound of r step 3 leads to a worst case time of $O(log^6 n)$.

Step 4 can be done in $O(\log n)$ since it's just one comparison.

The for-loop in step 5 has at most $\lfloor \sqrt{\phi(r)} log\ n \rfloor$ iterations and in each iteration we have to perform $O(\log n)$ multiplications of polynomials of degree r with coefficients of size $O(\log n)$ leading to a total time complexity of $O^\sim(r * log^2 n)$ for each iteration. Multiplied with the amount of iterations we result in a time complexity of $O^\sim(r * \sqrt{\phi r} log^3 n)$ being equal to $O^\sim(r^{\frac{3}{2}} * log^3 n)$ and, taking the upper bound of r into account, $O^\sim(log^{\frac{21}{2}} n)$. This time supercedes the time of all other steps and therefore notates the time complexity of the whole algorithm.
There are multiple proposed conjectures that would lead to a time complexity of down to $O^\sim(log^6 n)$ but none of them have been proven yet. These usually rely on finding a lower upper bound for r.

References

[1] M. Agrawal, N. Kayal, N. Saxena, “PRIMES is in P.” http://annals.math.princeton.edu/wp-content/uploads/annals-v160-n2-p12.pdf. [Accessed: 02.2019].

[2] M. Nair, “On Chebyshev-type inequalities for primes.” Amer. Math. Monthly 89(1982), 126–129.

[3] R. Lidl and H. Niederreiter, “Introduction to Finite Fields and their Applications.” Cambridge Univ. Press, Cambridge, 1986.

[4] H. W. Lenstra, Jr., “Primality testing with cyclotomic rings.” http://cr.yp.to/papers.html#aks. unpublished.

[5] A. Kalai, A. Sahai, M. Sudan, “Notes on primality test and analysis of AKS.” Private communication with the authors of AKS.

[6] J. von zur Gathen and J. Gerhard, “Modern Computer Algebra.” Cambridge Univ. Press, Cambridge, 1999.