Primality Testing : A Review

Debdas Paul

April 26, 2012

Abstract

Primality testing was one of the greatest computational challenges for the theoretical computer scientists and mathematicians until 2004 when Agrawal, Kayal and Saxena have proved that the problem belongs to complexity class P. This famous algorithm is named after the above three authors and now called The AKS Primality Testing having run time complexity $\tilde{O}(\log^{10.5}(n))$. Further improvement has been done by Carl Pomerance and H. W. Lenstra Jr. by showing a variant of AKS primality test has running time $\tilde{O}(\log^{7.5}(n))$. In this review, we discuss the gradual improvements in primality testing methods which lead to the breakthrough. We also discuss further improvements by Pomerance and Lenstra.

1 Introduction

"The problem of distinguishing prime numbers from a composite numbers and of resolving the latter into their prime factors is known to be one of the most important and useful in arithmetic. It has engaged the industry and wisdom of ancient and modern geometers to such and extent that its would be superfluous to discuss the problem at length . . . Further, the dignity of the science itself seems to require that every possible means be explored for the solution of a problem so elegant and so celebrated." - Johann Carl Friedrich Gauss, 1781

A number is a mathematical object which is used to count and measure any quantifiable object. The numbers which are developed to quantify natural objects are called Natural Numbers. For example, $1, 2, 3, 4, \ldots$. Further, we can partition the set of natural

numbers into three disjoint sets : {1}, set of {prime numbers} and set of {composite numbers}. A Prime number has factors only 1 and itself. The set of prime numbers has great importance in theoretical computer science and mathematics especially in the field of computer security and cryptography [1]. The properties of prime numbers were studied extensively in ancient Greece. In about 300 BC, Greek mathematician Euclid shows that there are infinitely many prime numbers. He also establishes a proof of the theorem which states that : Every integer can be written as a product of prime numbers uniquely. This theorem is also known as the Fundamental Theorem of Arithmetic. Later, in 200 BC another Greek mathematician Eratosthenes devises an algorithm to find all the primes up to a number n. This method is called the Sieve of Eratosthenes.

The idea is simple. First, we list all the integers from 2 to n. Then starting from 2 cancel out the multiples of 2 one by one upto n. In each iteration choose the number which is not marked yet. Finally the numbers that are remain unmarked will be the prime numbers up to n. The time complexity of the algorithm is $O(n \log \log n)$ in a random access machine model. An optimized version of Sieve of Eratosthenes is the Sieve of Atkin which marks the multiples of the squares of each prime rather than the simple multiples of the same (`http://en.wikipedia.org/wiki/Sieve_of_Eratosthenes`). But, those algorithms are inefficient in determining whether a given number n is prime or not as they take $\Omega(\sqrt{n})$ steps [1]. Some tests like ECPP (Elliptic Curve Primality Procedure) does a faster computation in testing primes that the Sieve of Eratosthenes, but compared to the composite-based primality test, it is inefficient dorsey

2 Evolution of primality testing methods

The Sieve of Eratosthenes is based on the definition of prime numbers. The algorithm is easy to implement but not efficient when numbers are of 100 digits or larger. On the other hand, these numbers are essential in the field of cryptography because prime factorization of very large number is still intractable. This fact ensures the privacy of decryption key in RSA algorithm. There using Eratosthenes method, it is practically

impossible to criticize RSA. Therefore, we have to find a better way to do that.

In real life, we always look for patterns in objects. There are many situations where we distinguish between two objects or predict a third object (without label) based on similarities in their patterns. With this, an obvious question will come to mind, "Is there any pattern among Prime numbers?". If we can find a pattern, then given an unknown number n, we can easily come to conclusion whether or not the number is prime. Unfortunately, there is no fixed pattern of prime numbers. In the word of number theorist Don Zagier, primes "grow like weeds among the natural numbers, seeming to obey no other law than that of chance [but also] exhibit stunning regularity [and] that there are laws governing their behavior, and that they obey these laws with almost military precision"(`http://en.wikipedia.org/wiki/Prime_number#Distribution`). Let us consider an elimination approach. Eliminate composite numbers. To eliminate composite numbers, we have to find a pattern in composite numbers. Now, we are going to discuss the methods based on unique patterns in composite numbers. Certainly, those methods contain loopholes like existence of pseudo primes. But, overall accuracy and efficiency of these methods are useful for many applications [6].

2.1 Chinese Primality Test

This method is an attempt to improve the efficiency of the method of Eratosthenes. Consider the following proposition [6]

Proposition 2.1. *Let n be a positive integer greater than 1. If 2^n is not congruent to $2(mod\, n)$, then n is composite.*

We can view this proposition in another way, if p is prime, then $2^p - 2$ is divisible by p. Primality testing method is essentially a filter. We design filters to avoid unwanted noise (in case of signal processing) or unwanted elements like virus or bacteria (biological filters). There are some numbers which are "noise" with respect to prime numbers. They are called pseudo-primes. A pseudo prime has dual nature. It acquires certain properties of prime although there are composite in nature. Therefore for a primality testing method

(filter), it is a challenging task to correctly identify those numbers (noise). There is no general definition of pseudo primes as the set varies with the primality testing methods. For example, Fermat pseudo primes are associated with Fermat's primality test which means that the Fermat's primality testing fails in case of those numbers. In this context, we should mention that there is no set of pseudo primes defined for the AKS primality test as this test is so robust that it works for all numbers. Therefore, existence of pseudo primes may be used to measure the efficiency of a primality test.

The Chinese Primality testing also has pseudo primes called base-2 pseudo primes. Now consider the following definition

Definition 2.2. *Let n be a composite number. If n divides $2^n - 2$, then n is a called a base-2 pseudoprime.*

Examples are: $341, 561, 645, 1105, 1729, 1905$. These are pseudo primes below 2000. $2^{341} \equiv 2(mod\, 341)$ but $341 = 11 \times 31$. The number of pseudo primes is actually small. As base-2 pseudo primes pass the Chinese Primality test, we can include them in the following theorem.

Theorem 2.3. *(Chinese Primality Theorem) Let n be an integer , $n > 1$. If 2^n is congruent to $2(modn)$, then n is either a prime or a base-2 pseudo prime.*

Proof. The assertion follows directly from Proposition 2.1 and from the definition 2.2. □

The theorem above is a special case of Fermat's little theorem we will discuss in a short while. The theorem actually comes from an expression (`http://journal.taiwanmathsoc.org.tw/index.php/TJM/article/view/622`).

If $[antilog(x \log 2) - 2]/x$ is an integer, then x is prime [1] number

2.2 Fermat's Primality Test

In the 17^{th} century, the work of Pierre de Fermat marks a milestone in the evolution of primality testing methods. We can view it as a rediscovery of the previous Chinese

[1]The function $f(x) = b^x$ is the inverse function of $log_b(x)$ and it is called the antilogarithm

Primality Theorem and a direct consequence of the process of generalization of base of pseudoprimes. Fermat's primality testing is based on the famous Fermat's little theorem.

Theorem 2.4. *If p is a prime number, then for any integer a, $a^p - a$ will be evenly divisible by p.*

In mathematical expression it can be written as,

$$a^p \equiv a(mod\, p)$$

We will not discuss the proof of the theorem here because it is beyond the scope of this discussion rather we will see how the theorem creates the basis for Fermat's primality testing. But, the reader should be aware of the fact that Fermat himself did not give the proof of his theorem. The formal proof was first given by mathematician Leibniz.

Fermat's primality test is probabilistic in nature. It will determine if a number is a probable prime or not. Probable primes are nothing but pseudo primes. The concept of the test is easy to understand. If we want to test a number say p is prime, we can pick a random number a between 1 and p and check if $a^{p-1} \equiv 1(mod\, p)$ holds. If a the equality does not hold for a value of a, then p is composite. Therefore, the test simply relies upon the choice of a. It may so happen that, we do not pick an a for which the equality fails. But, if we know previously that p is composite, then p will be called Fermat's pseudo prime of base a. For example, the smallest base-2 Fermat's pseudo prime is 341 as it satisfies Fermat's little theorem [6].

Algorithm 2.5. *FermatPrimalityTest(n):*

Pick $a \in \{1, 2, \ldots, n-1\}$ uniformly at random
If $a^{n-1}1(mod\, n)$, return ***composite****;*
Else return ***probably prime****.*

The run time of the above test is $O(M \times \log^2 n \log\log n \times \log\log\log n)$, where M is the number of times we pick the value of a and n is the number we want to test.

Proof. First, we have to considers the following facts (These will be also helpful while doing the runtime analysis of the AKS primality testing later in this discussion.)

Fact 2.6. *Let $m, n \in \mathbb{N}$ with at most k bits each. Then*

1. *m and n can be multiplied with $O(k(\log\ k)(\log\log\ k)) = \tilde{O}(k)$ bit operations.*
2. *n div m and n mod m can be computed using $O(k(\log\ k)(\log\log\ k)) = \tilde{O}(k)$ bit operations.*
3. *Multiplication of two polynomials of degree d with coefficients at most m bits in size cane be done in $\tilde{O}(d \times m)$ bit operations.*

Another important fact is that the time complexity of a fast modular exponentiation is $O(\log\ exponent)$.

Now, if the number n is represented by k bits then $k = O(\log\ n)$. The modular exponentiation step a^{n-1} will take $O(\log\ n)$ time. The step to verify the modulo will take $\tilde{O}(\log\ n)$ time. Now if we pick a for M times, then the total time will be the same as above. □

Now the question remains, how hard is to find an a, i.e., the witness for Fermat's primality testing. Here, we will go through a series of lemmas to find the answer. Another interesting point is that, after these lemmas, we will be able to define a new trait of numbers called *Carmichael numbers*. These numbers pass the Fermat's Test. To be more precise, given a Carmichael number, the above algorithm of Fermat's Test will return composite with probablity less than 1/2 and therefore, it is not easily distinguishable from the actual prime numbers. This shortcoming leads to a stronger test called the Miller-Rabin test or Strong Pseudoprimality test which we will discuss in the next section.

The primary goal of these lemmas is to show that $a^{n-1} \equiv 1 (mod\, n)$ has many solutions and therefore the choice of a is prevalent.

Lemma 2.7. *Let a, b be any two integers and let $d = gcd(a, b)$. The set $a\mathbb{Z} + b\mathbb{Z} = \{ar + bs : r, s \in \mathbb{Z}\}$ is equal to $d\mathbb{Z}$ where $d = gcd(a, b)$*

Proof. The set $a\mathbb{Z} + b\mathbb{Z}$ is closed under addition and subtraction, so $a\mathbb{Z} + b\mathbb{Z} = c\mathbb{Z}$ for some integer c. If $d = gcd(a, b)$, then every element of $a\mathbb{Z} + b\mathbb{Z}$ is divisible by d, so $d|c$. But, $a, b \in c\mathbb{Z}$ and hence divsiible by c which implies that c is a common divisior or a and b, so $c|d$.It follows that $c = d$. □

Lemma 2.8. *If $gcd(a, n) = 1$ then there is an integer a^{-1} such that $a.a^{-1} \equiv 1(mod\, n)$.*

Proof. By Lemma 2.7, $a\mathbb{Z} + n\mathbb{Z} = \mathbb{Z}$. This implies that $\exists r, s \in \mathbb{Z}$ such that $ar + ns = 1$. Therefore, $a.r \equiv 1(mod\, n)$. □

Lemma 2.9. *If $b, c, n \in \mathbb{Z}^+$ such that $gcd(c, n) = 1$ and the congruence $a^b \equiv c(mod\, n)$ has $k > 0$ solutions, then the congruence $a^b \equiv 1(mod\, n)$ also has k solutions.*

Proof. Let a_0 be a solution of $a^b \equiv c(mod\, n)$. Then $gcd(a_0, n) = 1$ since otherwise $gcd(a_0^b, n) = gcd(c, n)$ would be greater than 1, contradicting our hypothesis. By Lemma 2.8 we can have a a_0^{-1} such that $a_0.a_0^{-1} \equiv 1(mod\, n)$ A one-to-one correspondence between the solution sets of $a^b \equiv c(mod\, n)$ and of $a^b \equiv 1(mod\, n)$ is given by the mapping $y \rightarrow y.a_0^{-1}$ □

Now we can see, If $gcd(a, n) = 1$ and a is Fermat's witness for n, then $a^{n-1} \equiv c(mod\, n)$ for some $c \neq 1$ which satisfy $gcd(c, n) = 1$. From Lemma 2.9, we prove the one to one correspondence. Therefore, there will be as many Fermat's witnesses as Fermat's liars. Fermat's liars are composite numbers. If q is a Fermat's liar then $n \in \mathbb{N} : n^q \equiv n(mod\, q)$. Now we are ready to define Carmichael numbers.

Definition 2.10. *If q is a Carmichael number then $\forall n \in \mathbb{N} : n^q \equiv n(mod\, q)$*

To handle these numbers efficiently rather separate them out from the set of primes, Miller and Rabin propose a probabilistic test. The original test due to Miller is deterministic but the determinism lies on the generalized Riemann hypothesis which is unproven. Therefore, Miller modified it and presented an unconditional probabilistic algorithm. Before going into detail of Miller-Rabin test, we have to know another test called Solovey-Strassen test. The reason is that, this test is superior than Fermat's test

and inferior than Miller-Rabin test in terms of percentage of pseudo primes they produce. Remember, the lesser is the number of pseudo primes produce by a test, the more deterministic it is. A deterministic test has no pseudo primes (Example :Lucas Test)

2.3 The Solovey-Strassen Primality test

The Solovey-Strassen primality was popularized due to advent of public-key cryptography, especially RSA cryptosystem. This test is based on Euler witness and Jacobi symbol.

Definition 2.11. *For any integer a and any positive integer n the Jacobi symbol is defined as the product of Legendre symbols corresponding to the prime factor n*

$$(\frac{a}{n}) = (\frac{a}{p_1})^{\alpha_1}(\frac{a}{p_2})^{\alpha_2}\dots(\frac{a}{p_k})^{\alpha_k},$$

where $n = p_1^{\alpha_1}p_2^{\alpha_2}\dots p_k^{\alpha_k}$.
$(\frac{a}{p})$ *is a Legendre symbol defined for all integers and all odd primes p by*

$$(\frac{a}{n}) = \begin{cases} 0 & \text{if } a \equiv 0(mod\, p) \\ +1 & \text{if } a(mod\, p) \neq 0(mod\, p) \text{ and for some integer } x, a \equiv x^2(mod\, p) \\ -1 & \text{if there is no such } x \end{cases}$$

Jacobi symbol is only defined when the numerator is an integer and denominator is a positive odd integer (`http://en.wikipedia.org/wiki/Jacobi_symbol`*).*

The Solovey-Strassen test is based on the following fact.

Fact 2.12. *(Euler criterion) Let n be an odd prime. Then,* $a^{(n-1)/2} \equiv (\frac{a}{n})(mod\, n)$ *for all integers a which satisfy* $gcd(a, n) = 1$.

Therefore the overall idea of the test is if we have a value p and want to determine if it is prime, we can check many random values of a and make sure the above equality holds. If it does not hold for some a, we know that p must not be prime.

2.4 The Miller-Rabin Probabilistic Primality Test

In Fermat's test we prove that a number n is composite in the following way :

1. Exhibit a Fermat's witness for n, i.e, a number x satisfying $x^{n-1}(mod\, n)$.

The Miller-Rabin test is based on the following thing :

1. Exhibit a "Fake square root of $1 mod n$ " i.e, a number x satisfying $x^2(mod\, n)1(mod\, n)$ but $x(mod\, n) \neq \pm 1(mod\, n)$.

The follow lemma explains the reasons.

Lemma 2.13. *If x, n are positive integers such that $x^2(mod\, n)1(mod\, n)$ but $x(mod\, n) \neq \pm 1(mod\, n)$, then n is composite.*

Proof. The hypotheses of the lemma imply that n is a divisor of $x^2 1 = (x+1)(x1)$, but n divides neither $x+1$ nor $x1$. This is impossible when n is prime. □

The idea of the test is to pick a random number x in $1, 2, ..., n1$ and use it to try finding either a Fermat witness or a fake square root of $1 mod\, n$.

2.5 A comparison between Fermat, Solovey-Strassen and Miller-Rabin

The following fact describes the relation between Fermat liars, Euler liars and strong liars.

Fact 2.14. *if a is an Euler liar for n, then it is also a Fermat liar for n.*
If a is a strong liar for n, then it is also an Euler liar for n.

The less the number of liars, the more accurate the test is. It is evident from the above facts that Miller-Rabin is never worse than Solovey-Strassen test. However, in some cases Miller-Rabin and Solovey-Strassen perform equally.

Fact 2.15. *If $n \equiv 3(mod\, 4)$, then a is an Euler liar for n iff it is a strong liar for n.*

Regarding the computational costs, Miller-Rabin test is better than Fermat's test because it requires at worst the same amount of computation of modular multiplication as Fermat's test. Also Miller-Rabin takes less amount of time than Solovey-Strassen due to absence of Jacobi symbol computation. Moreover the error probability of Solovey-Strassen is bounded above by $\frac{1}{2^t}$ whereas in case of Miller-Rabin it is bounded above by $\frac{1}{4^t}$ [2].

2.6 Lucas Primality Test

The Lucas Primality Tes is one of the most complicated tests among the tests discussed before. It is based on the notion of Lucas sequences. Let D be the discriminant of the quadratic equation $x^2 - ax + bx = 0$. Also let α and β are the two roots of the equation. Then we have

$$\alpha = (a + \sqrt{D}D)/2; \beta = (a - \sqrt{D})/2$$

. Lucas sequence $U(a, b)$ and $V(a, b)$ can be defined as follows: Let

$$U_k(a, b) = \frac{\alpha^k - \beta^k}{\alpha - \beta}; V_k(a, b) = \alpha^k + \beta^k$$

then Lucas sequence of the pair (a, b) are the sequences,

$$U(a, b) = (U_0(a, b), U_1(a, b), U_2(a, b), \dots); V(a, b) = (V_0(a, b), V_1(a, b), V_2(a, b), \dots)$$

A simple example of Lucas sequence is the sequence of Fibonacci numbers with $a = 1, b = -1$ and produced by $U(a, b)$. Now to apply Lucas sequence to primality testing, we have to replace the term a^{n-1} in the Miller-Rabin Strong test with a Lucas sequence.

Theorem 2.16. *(Lucas Theorem) Let $a, b, DandU_k$ be defined as above. If p is an odd prime, if $gcd(b, p) = 1$ and if $(\frac{D}{p}) = -1$, then p divides U_{p+1}*

Theorem 2.17. *(Lucas Test) Let n be an odd positive integer. If n does not divide U_{n+1}., the n is composite.*

Lucas test also has its set of pseudo primes called Lucas pseudo primes. Here is a formal definition. If n is composite, if the largest common divisor of n and b is equal to 1, if $(\frac{D}{n}) = -1$, and if n divides U_{n+1}, then n is called a Lucas pseudo-primes with parameter a and b.

2.7 The Elliptic Curve Primality Test

Solovey-Strassen and Miller-Rabin primality testing methods produce a proof or witness with a high probability if the input number is composite. Those algorithms fail to produce a witness or proof of compositeness for a prime input. They give only a probabilistic support to the answer but no definitive proof. As we mention earlier that Miller-Rabin can be a deterministic polynomial time algorithm for primality test provided the generalized Riemann hypothesis is proved. Therefore, all the previous composite based algorithms produce proofs or witnesses for composites. The question arise whether or not we can produce proofs or witnesses for primes. The answer for this question leads to the Elliptic Curve Primality Test. It is a group theoretical approach to produce short proof for primes. The basis of this test lies in the effort of primality testing using cyclic groups [4].

Algorithm 2.18. *PrimalityUsingCyclicGroups($n > 2$):*

Guess a generator g, where $1 < g < n$
Verify that $g^{n-1} \equiv 1 (mod\, n)$ if not halt without accepting.
Guess a multi set of positive integers $p_1, \ldots, p_k$ and verify that $p_1, \ldots, p_k = n - 1$. If not, halt without accepting.
Recursively, verify that each $p_i, 1 \leq i \leq k$, is a prime number. If not all are prime, halt without accepting.
For each $1 \leq i \leq k$, verify that $g^{(n-1)/p_i} \neq 1 (mod\, n)$. If not halt without accepting.
Accept.

In the above algorithm we use the order of the cyclic group $(\mathbb{Z}/p\mathbb{Z})^{\times}$ to determine the primes. The run time of the above algorithm is $O(log^5 n)$. In the Elliptic Curve Method, we use the order of the curve (The order of the curve is the number of distinct points on

the curve, including the zero point) to determine the prime. Now let us define the elliptic curves.

Definition 2.19. *Let $E(a,b)/p$ denote the* ***elliptic curve*** *over finite field F_p (p is prime) whose element are pairs (x,y) of non-negative integers less than p satisfying $y^2 \equiv x^3 + ax + b (mod\, p)$, together with point at infinity. Given (x_1, y_1) in $E(a,b)$, we can define $(x_i, y_i) \equiv (x_1, y_1)^i (mod\, p)$* The theorem below gives us the the formula for counting points on an elliptic curve over a finite field.

Theorem 2.20. *Let $|E(a,b)/p|$ denote the number of points on elliptic curve $E(a,b)$ over F_p. Then $|E(a,b)/p| = p + 1 + \Sigma_{x_p}(\frac{x^3+ax+b}{p})$*

Now let us define the Goldwasser-Kilian ECPP (Elliptic Curve Primality Proving) Theorem based on which we will formulate an algorithm for primality testing.

Theorem 2.21. *Let $n > 1$ be an integer such that $(n, 6) = 1$. Let $E(a,b)/\mathbb{Z}_n$ be a pseudo curve(a curve over $\mathbb{Z}/n\mathbb{Z}$ instead of $\mathbb{Z}/p\mathbb{Z}$) and let $s, m \in \mathbb{Z}^+, s|m$. Assume that there exists a point $P \in E(a,b)$ such that $mP = O$, where O is the point at infinity and suppose that for every prime q dividing s, we have $[m/q]P \neq O$. Then for every prime p dividing n, we have the order $|E(a,b)/F_p| \equiv 0 (mod\, s)$. More over if $s > (n^{1/4} + 1)^2$ then n is prime.*

The idea of the algorithm is to find curves with the orders that have sufficiently large probable prime factors and we recurs on the idea that thee factors are provably prime. In each recursive step, we use the ECPP theorem stated above with s as the probable prime factor of the curve order. We continue this process for smaller and smaller probable primes until we get a value of s that is so small that we can prove it is prime using any naive primality test. The slowest part of the algorithm is to asses the order of curves. Even the quickest method take $O(log^6 n)$ steps to execute (`http://www.math.dartmouth.edu/~thompson/ecc.pdf`).

2.8 The Breakthrough - AKS primality testing

Folker Bornemann in an article addresses this break through as a "break through for *Everyman*" [3] . The word "Everyman" implies the underlying simplicity of the approach. The approach is short, elegant and easily understandable. Let us examine what is "so simple".

Instead of going for heavy mathematical formulation, let us discuss the underlying idea in simple manner. The key idea of the AKS algorithm lies in the Pascal's Triangle (`http://en.wikipedia.org/wiki/Pascal's_triangle`). The Pascal's Triangle has an interesting property. If n is prime, then every number in the n^{th} row of the triangle (except 1) is a multiple of n. If n is composite this does not hold. Using this fact and Fermat's little theorem, we can write the following corollary.

Corollary 2.22. *For any a that is relatively prime to n,*

$$(X + a)^n mod\, n = (X^n + a) mod\, n$$

for any values of X iff n is prime.

Example 2.23. *Let $n = 3$ and $a = 2$, then $(X+2)^3 mod 3 = (X^3+6X^2+12X+8) mod 3 = X^3 + 2$ which is $X^n + a$. Now let $n = 4$ and $a = 3$ then $(X + 3)^4 mod 4 = X^4 + 2X^2 + 3$ which is not equal to $X^4 + 3$ for all values of X.*

From the above examples, it is evident that if we could examine the binomial expansion of $(X + a)^n$, then we would have an infallible test for whether n is prime or not. But, what if n is huge. There will be too many terms to check. Is there any way to speed up the process? The answer is the AKS primality test! The AKS algorithm speed up the binomial test and correctly answer whether or not n is prime (`http://www.scottaaronson.com/writings/prime.pdf`). They take the mod not only with respect to n but also with respect to $X^r - 1$, where r is small prime number. Therefore, instead of computing $(X + a)^n$, we will compute the remainder when $(X + a)^n$ is divided by $X^r - 1$. The remainder has only $r + 1$ terms which is small compare to $n + 1$. For n is prime the

following is equality always true.

$$(X+a)^n mod(n, X^r-1) = (X^n+a) mod(n, X^r-1)$$

but it is not obvious that it is always false when n is composite. It is possible that two different polynomials could have the same remainder when we divide both of them by X^r-1. AKS show us *If n is composite, and if we choose the right value of r, then we need to try only for a small number of a's until we find one such that*

$$(X+a)^n mod(n, X^r-1) \neq (X^n+a) mod(n, X^r-1)$$

. If we find such an a then n is composite. Now, AKS is deterministic in the sense that we can pick a deterministically. Now let us examine the runtime of the algorithm. There are two important parameters in the whole AKS primality testing: one is a and another is r. The run time complexity of the algorithm is depending upon the bounds on these two parameters. If we can prove that the bounds are polynomial then we are done. The AKS trio show that a and r are bounded above by some polynomial in $log(n)$. Here is the lemma which shows the above fact.

Algorithm 2.24. *(TheAKSPrimalityTest($n > 1$):*

1. *if $a, b > 1 \in \mathbb{N}$ such that $n = ab$, then output* ***Composite***
2. *Find the minimal $r \in \mathbb{N}$ such that $o_r(n) > log^2(n)$; $o_r(n)$ is the order of r modulo n is the smallest number k such that $r^k = 1 (mod\, n)$*
3. *For $a = 1$ to r*
   —- if $1 < (a, n) < n$, then output ***composite***
4. *if $r \geq n$, then output Prime*
5. *For $a = 1$ to $\lfloor \sqrt{\phi(r)}.log(n) \rfloor$ ($\sqrt{\phi(r)}$ is Eulers totient function giving the number of numbers less than r that are r) do*

—- *if* $(X+a)^n\,(mod X^r-1,n) \neq X^n+a(mod X^r-1,n)$ *then output* ***Composite***

6. *output* ***Prime***

2.8.1 Runtime Analysis

The most important step in the algorithm is step 5 which dominates over all other previous steps. step 1 will take at most $\tilde{O}(log^3(n))$ bit operations. Step 2 and 3 will take at most $\tilde{O}(log^7(n))$ bit operations. Now, in step 5, we determine the whether the congruence

$$(X+a)^n \equiv (X^n+a)(mod X^r-1,n)$$

holds for $a = 1,2,3,\ldots,\lfloor\sqrt{\phi(r)}.log(n)\rfloor$. Given $a \leq \lfloor\sqrt{\phi(r)}.log(n)\rfloor$, we may calculate $(X+a)^n$ in the ring $\mathbb{Z}/n\mathbb{Z}$ as reducing modulo X^r-1 is trivial. In order to calculate $(X+a)^n$, we must perform $O(log(n))$ multiplications of polynomial of degree less than r with coefficients of size $O(log(n))$ (as the coefficients are in written modulo n; all polynomials are reduced modulo X^r-1 during Fast modular exponentiation) Each congruence therefore takes $\tilde{O}(log^7(n))$ bit operations to verify. Therefore the total time of step 5 is $\tilde{O}(\sqrt{\phi(r)}log(n)log^7(n)) = \tilde{O}(log^{5/2}(n)log(n)log^7(n)) = \tilde{O}(log^{10.5}(n)$. which dominates the time complexity of other steps. Therefore, the overall runtime of AKS Primality testing is $\tilde{O}(log^{10.5}(n)$ which is indeed polynomial.

2.8.2 Improvements by Lenstra and Pomerance

Initial review of the AKS algorithm by mathematicians was predominately positive. The most significant observed flaw was that the order of the algorithm was unacceptably high. As a result, the execution time was far too great in comparison to times of the current deterministic primality tests. There are many people who make improvements to the original algorithm but among them the improvements made by Lenstra and Pomerance is significant one. In their modified version they use only 4 steps [7, 5].

Algorithm 2.25. *AKSWithGaussianPeriods*$(n \in \mathbb{N}, n > 1)$

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* $gcd(a, n) \neq 1$ *for all* $a \leq r$*, output* ***Composite***

4. *For* $a = 1$ *to* $\lfloor \sqrt{(}r)log(n) \rfloor$ *do*
   — *if* $(X + a)^n \equiv X^n + a(mod(f(X), n))$ *,output* ***Prime***

The function $f(x)$ is a monic polynomial of degree r with integer coefficients such that the ring $[x]/(n, f(x))$ becomes a pseudofield. r can be calculated as follows:

1. $q > \lfloor log^2(n) \rfloor$

2. Compute $n^j mod\, q$ for $j = 1, 2, \ldots, [log^2(n)]$

3. If residue equals $1 mod\, q$ then $q = q + 1$

4. Else $r = q$

The value of r after the above modification is reduced by a factor of 4 which significantly reduce the number of iterations required in the final loop. The run time of the modified algorithm is $\tilde{O}(log^7.5(n)$ Therefore, the key to the improvements in the AKS algorithm is to reduce the number of iterations in the final loop. Here is a conjecture which reduces the runtime to $\tilde{O}(log^3(n)$

Conjecture 2.26. *If* r *is a prime number that does not divide* n *and if* $n \neq 0 \, mod\, r$*, then either* n *is prime or* $n^2 = 1 mod\, r$

3 Conclusion

We will conclude this discussion by discussing consequences of AKS Primality Testing [1]. There is a long standing problem still open called Factoring:

given an integer n, find the prime factors of n.

This problem is not expected to lie in $\mathcal{P}$ (otherwise the RSA cryptosystem will no longer exist!) but in $\mathcal{NP}$. By definition a problem is in $\mathcal{NP}$ if we can verify (not solve) the solution in polynomial time. Using any factorization algorithm, we can factor a number and now with the AKS we can very the factors as prime in polynomial time. Therefore, the AKS is an advancement towards the greatest open problem in science (in broader sense)- $\mathcal{P} \neq \mathcal{NP}$. Here is a nice quotation by Paul Leyland on this simple, deterministic discovery :

"Everyone is now wondering what else has been similarly overlooked"

4 Acknowledgments

The author would like to thank Prof. Henry Baird, Department of Computer Science and Engineering, Lehigh University for his thoughtful comments while preparing this report.

References

[1] M. Agrawal, N. Kayal, and N. Saxena. Primes is in p. *Annals of Mathematics*, pages 781–793, 2004.

[2] A. BEKTAŞ. *PROBABILISTIC PRIMALITY TESTS*. PhD thesis, MIDDLE EAST TECHNICAL UNIVERSITY, 2005.

[3] F. Bornemann. Primes is in p: A breakthrough for" everyman". *NOTICES-AMERICAN MATHEMATICAL SOCIETY*, 50(5):545–553, 2003.

[4] S. Goldwasser and J. Kilian. Primality testing using elliptic curves. *Journal of the ACM (JACM)*, 46(4):450–472, 1999.

[5] H. W. Lenstra and C. Pomerance. *Primality testing with Gaussian periods*. Dartmouth, 2009.

[6] Z.S. McGregor-Dorsey. Methods of primality testing. *MIT Undergraduate Journal of Mathematics*, 1:133–141, 1999.

[7] R.G. Salembier and P. Southerington. An implementation of the aks primality test. *Computer Engineering*, 2005.