CS 787: **Advanced Algorithms**

Topic: Primality Testing **Presenters**: Maheswaran and Ramachandran

17.9.1 Introduction to Primality Testing

Primality test is a test to determine whether a given number is prime or not. These tests can be either deterministic or probabilistic. Deterministic tests determine absolutely whether a given number is prime or not. Probabilistic tests may, with some small probability, identify a composite number as a prime, although not vice-versa.

17.9.2 Motivation

Prime numbers have applications in many fields:

- **Cryptography**
  Prime numbers are used to compute the cipher text of given plain text through encryption.
  $$Cipher\ Text\ = (Plain\ Text^{e})mod\ n$$
  where n is the product of two large distinct prime numbers and e is the public key.

- **Hash functions**
  Prime numbers are used in hash functions to compute hash value of a data.
  $$h_{a,\,b}(x) = [(ax + b)\ mod\ p)]\ mod\ n$$
  where p is a prime number.

- **Pseudo Random Number Generators (PRNG)**
  Linear congruential generator is an algorithm to generate pseudo random numbers. The generator has the following recurrence form:
  $$X_{n+1} = \left(aX_{n} + c\right) mod\ m$$
  where c and m are relatively prime, which implies that distinct prime numbers can be chosen for c and m.

17.9.3 Desired Characteristics of Primality Tests

Generality

There are many fast primality tests but they work for numbers with only certain properties. For example, the Lucas–Lehmer test for Mersenne numbers can only be applied for Mersenne numbers. Pépin's test works only for Fermat numbers. We would like an algorithm that can be used to test any general number for primality.

Polynomial time in the input size

The maximum running time of the algorithm can be expressed as a polynomial over the number of digits in the target number. Certain algorithms can deterministically determine whether a given number is prime or not, but their running time is not polynomial for all inputs. Algorithms like ECPP (Elliptic Curve Primality Proving) and APR (Adleman-Pomerance-Rumely) deterministically prove primality, but they do not have polynomial running time for all inputs. We would like an algorithm that runs in polynomial time for all possible inputs.

Deterministic

The algorithm guarantees to deterministically distinguish whether the target number is prime or composite. There are certain algorithms like Solovay-Strassen and Miller-Rabin (which are the topics of our study) that are randomized algorithms, and can test any given number for primality in polynomial time, but they may produce some false positives also. We would like an algorithm that can "prove" or "disprove" primality with certainty.

Unconditional

The Miller test for primality is fully deterministic and runs in polynomial time over all inputs, but its correctness depends on the truth of the yet-unproven generalized Riemann hypothesis. We would like an algorithm that is not based on any unproven hypothesis.

17.9.4 Certain Kinds of Algorithms and Complexity Classes

- **Monte Carlo**

  For any x which does not belong to the language L, the algorithms that belong to this class will conclusively say that $x \notin L$. For any x which belongs to the language L, these algorithms say that $x \in L$ with atleast 0.5 probabiltiy. This type of behavior is termed as L is accepted with one-sided error. The class of languages with polynomial time Monte Carlo recognition algorithms is termed as RP.

- **Atlantic City**

  For any x, algorithms that belong to this class will say either $x \in L$ or $x \notin L$ with a probability of 0.75. This is termed as x is accepted with two-sided error. The class of languages with polynomial time Atlantic City recognition algorithms is termed as BPP.

- **Las Vegas**

  It can be defined as a combination of two algorithms: a Monte Carlo algorithm for L and a Monte Carlo algorithm for $\bar{L}$ (complement of L). These classes of algorithms conclusively say whether $x \notin L$ or $x \in L$ but the running time is probabilistic. The class of languages with polynomial time Las Vegas algorithms is termed as ZPP.

The focus of our study is on Monte Carlo algorithms. We present two Monte Carlo tests for primality. Hence, primes belong to RP.

17.9.5 Some Definitions and Lemmas

In this section, we present certain definitions and lemmas (without proofs) which are used in later sections.

Definition 17.9.5.1 Legendre Symbol

For any prime number p and a positive integer b, the $Legendre$ Symbol is defined by

$$\left[\frac{b}{p}\right] = \left\{ \begin{array}{rl} 0 & if\ b\ and\ p\ are\ not\ relatively\ prime \\ 1 & if\ b\ is\ a\ quadratic\ residue\ mod\ p \\ -1 & if\ b\ is\ a\ quadratic\ non-residue\ mod\ p \end{array} \right\}$$

If p is 2, the value of $Legendre$ symbol is one for any odd b and 0 otherwise.

Definition 17.9.5.2 Jacobi Symbol

Legendre symbol holds good when the denominator is prime. Jacobi symbol extends it to denominators other than primes.

For any positive integer n, Jacobi symbol is defined by prime decomposition of n.

$If\ n = \prod_{i=1}^{r} p_i^{\alpha_i}$, then,

$$\left(\frac{m}{n}\right) = \prod_{i=1}^{r} \left[\frac{m}{p_i}\right]^{\alpha_i}$$

Lemma 17.9.5.3 For any odd prime p,

$$\left[\frac{2}{p}\right] = (-1)^{\frac{(p^2-1)}{8}} = \left\{ \begin{array}{rl} 1 & if\ p\ =\ \pm 1\ (mod\ 8) \\ -1 & if\ p\ =\ \pm 3\ (mod\ 8) \end{array} \right\}$$

Lemma 17.9.5.4 Law of Quadratic Reciprocity

For p and q odd primes,

$$\left[\frac{p}{q}\right]\left[\frac{q}{p}\right] = (-1)^{\frac{(p-1)(q-1)}{4}} = \left\{ \begin{matrix} -1 & if\ p = q = 3\ (mod\ 4) \\ 1 & otherwise \end{matrix} \right\}$$

Lemma 17.9.5.5 For integers m and n and factorization of $n = n_1 n_2$ and $m = m_1 m_2$, the Jacobi symbol obeys the following:

$$\left(\frac{m_1 m_2}{n}\right) = \left(\frac{m_1}{n}\right)\left(\frac{m_2}{n}\right)$$

and

$$\left(\frac{m}{n_1 n_2}\right) = \left(\frac{m}{n_1}\right)\left(\frac{m}{n_2}\right)$$

Lemma 17.9.5.6 For m and n relatively prime and odd,

$$\left(\frac{n}{m}\right)\left(\frac{m}{n}\right) = (-1)^{\frac{(m-1)(n-1)}{4}}$$

Lemma 17.9.5.7 For any odd integer n,

$$\left(\frac{2}{n}\right) = (-1)^{\frac{(n^2-1)}{8}}$$

Lemma 17.9.5.8 Suppose $n - 1 = q^k r$, where q is a prime and q does not divide r.

If there exists a such that $a^{n-1} \equiv 1\ (mod\ n)$ and gcd$\left(a^{\frac{n-1}{q}} - 1, n\right) = 1$, then for every prime $p|n$ we have $p \equiv 1\ (mod\ q^k)$.

Lemma 17.9.5.9 If n is Carmichael, then it is odd, squarefree, and divisible by atleast 3 distinct primes.

17.9.6 The Three Theorems and Corresponding Tests

17.9.6.1 Fermat's little theorem

We state, without proving, a fundamental theorem in number theory:

For any prime number p and an integer a which is coprime to p, $a^{p-1} - 1$ will be divisible by p.

$$a^{p-1} \equiv 1 \qquad (mod\ p)$$

Carmichael number is a composite number n which satisfies the following criterion

$$b^{n-1} \equiv 1 \qquad (mod\ n)$$

for all numbers b which are relatively prime to n. These numbers are also called Fermat's absolute pseudoprimes.

For testing whether a number p is prime, we pick a number a which is relatively prime to p and see if the equality holds. If it fails for any value of a, then p is composite. If the equality holds for many values of a, then we can say that p is probably prime. The presence of Carmichael numbers prevents us from conclusively determining that p is a prime. There could be cases where the number $'a'$ that we pick doesn't fail the equality for a composite number n. Then, a is called a $Fermat\ Liar$ for the number n. If we pick a number a such that it fails the equality for a number n, then we can deterministically say that n is a composite number and such an a is called a $Fermat\ witness$ for the compositeness of n.

We usually don't use Fermat's theorem for primality testing because

a) The test is not useful for Carmichael numbers.
b) We don't have any bounds on the number of Fermat liars for a given composite number.

17.9.6.2 Euler's Criterion

For any odd prime number p and any positive integer b,

$$\left[\frac{b}{p}\right] = b^{\frac{p-1}{2}} \ (mod\ p)$$

If, for a composite number p, there exists a positive integer b that satisfies Euler's criterion, b is called an *Euler Liar* of the composite number p.

We can use Euler's criterion for primality testing because

a) There is no composite number that satisfies Euler's criterion always.

b) We have a bound on the number of Euler liars for a composite number.

Proof:

If $(b,p) \neq 1$, then $b = 0\ (mod\ p)$ and the equality follows, We henceforth consider integers b relatively prime to p. It can be seen that the set,

$$A = \{\, a \in (\boldsymbol{Z}/p\boldsymbol{Z})^{*} \mid a^{\frac{p-1}{2}} = 1\ (mod\ p)$$

form a subgroup of $(\boldsymbol{Z}/p\boldsymbol{Z})^{*}$**.** Any quadratic residue $b = a^{2}$ is in A, since

$$b^{\frac{p-1}{2}} = a^{\frac{2(p-1)}{2}} = 1\ (mod\ p)$$

by Fermat's little theorem. Half the elements of $(\boldsymbol{Z}/p\boldsymbol{Z})^{*}$ are quadratic residues, hence A is of size either $(p-1)$ and $\frac{p-1}{2}$ (the order of a subgroup must divide the order of the group). The subgroup A does not include any generator of the group, hence its size is not $(p-1)$. Hence, A contains exactly the quadratic residues.

From Fermat's little theorem, the square of $b^{\frac{p-1}{2}}$ is 1, hence for those integers relatively prime to p but outside of A, the power must evaluate to the only other root of one, that is -1.

17.9.6.3 The Third Criterion

Let $n - 1 = 2^{s}.d$, where d is odd, and let $S(n)$ be the set of strong liars. $S(n)$ is defined as follows:

$$S(n) = \{a \in (\boldsymbol{Z}/n\boldsymbol{Z})^{*} : a^{d} \equiv 1\ (mod\ n) \quad or$$

$$a^{2^{r}.d} \equiv -1\ (mod\ n) for\ some\ r, 0 \leq r < s\}$$

Let n be an odd integer ≥ 3. Then n is prime iff $S(n) = (\boldsymbol{Z}/n\boldsymbol{Z})^{*}$. If n is composite, then $|S(n)| \leq \frac{n-1}{4}$

A nonzero element of $(\boldsymbol{Z}/n\boldsymbol{Z})^{*} - S(n)$ is a *strong witness* to the compositeness of n.

The proof of this criterion is in Section 17.9.8.

We can use this criterion for primality testing because

a) There is no composite number that satisfies this criterion always.
b) We have a bound on the number of strong liars for a composite number.

17.9.7 The Solovay-Strassen Test

This test is based on the Euler's criterion (Section 17.9.6.2). Let n be the number that we want to test for primality. Solovay-Strassen algorithm works by selecting random integers and computing large powers of them in the ring $\boldsymbol{Z}/n\boldsymbol{Z}$**.** In addition to this, the algorithm also computes the Jacobi symbol for these numbers. If the results from the above two calculations don't match, then the number is composite. As we have seen earlier, had the number n been prime, the Jacobi symbol would have been in fact the Legendre symbol. For the Legendre symbol, results from both the calculations should be equal [theorem 17.9.6.2].

As there are some Euler liars, it is not necessary for the two calculations to agree when n is composite. In fact, among those numbers which are relatively prime to n in the set $\{1,2,..,n-1\}$, at most 50% of them are Euler liars (We will prove this theorem). For any given integer n, the algorithm will pick k integers u.a.r. and do the calculations. The probability that all the k numbers picked are Euler liars is atmost $\frac{1}{2^k}$ which is almost zero for sufficiently large k.

Algorithm

```
$SOLOVAY-STRASSEN(n)$
$choose\ a\ at\ random\ from\ \{1,2,\ldots.,n-1\}$
$if\ gcd(a,n)\ \neq 1$
    $then\ return\ `composite'.$
$elseif$
    $\left(\frac{a}{n}\right) \neq a^{\frac{(n-1)}{2}}\ (mod\ n)$
        $then\ return\ `composite'.$
    $else$
        $return\ `prime'.$
```

Note that lemmas 17.9.5.3 to 17.9.5.7 provide efficient ways to compute the Jacobi symbol. Lemma 17.9.5.6 is used in a way similar to Euclid's algorithm to reduce the magnitude of the numbers being computed. When used with other lemmas from 17.9.5.3 to 17.9.5.7, we can compute Jacobi symbol efficiently.

Lemma 17.9.7.1 For any prime p, there is a generator for $\left(\boldsymbol{Z}/p^2\boldsymbol{Z}\right)^*$

Proof:

For any prime p, there is a generator g for $(\boldsymbol{Z}/p\boldsymbol{Z})^*$. Let h be equal to g or $g(1+p)$ depending on whether or not

$$g^{p-1} = ?\,1\ (mod\ p^2)$$

If $g^{p-1} = 1\ \left(mod\ p^2\right)$, then

$$\left(g(1+p)\right)^{p-1} = 1 + (p-1)p + p^2 w = 1 + p(p-1)\quad (mod\ p^2)$$

where w is some integer $\Rightarrow h$ can be chosen such that $h^{p-1}\ \left(mod\ p^2\right)$ is not one. Since in either case, $h = g\ (mod\ p), h$ is a generator of $(\boldsymbol{Z}/p\boldsymbol{Z})^*$.

Next we show that h is a generator of $\left(\boldsymbol{Z}/p^2\boldsymbol{Z}\right)^*$. Let $h^j = 1\ \left(mod\ p^2\right)$. This congruence remains true modulo p.

$\therefore\ (p-1)|j$. We can write j as $j = (p-1)j'$. But j must also divide the order of the group, $j|p(p-1)$, hence $j'|p$. Since h^{p-1} is not one, j' cannot be one, so it must be p.

Hence, the order of h is the size of the group.

Lemma 17.9.7.2 For any odd composite number n, there is a b relatively prime to n such that,

$$\left(\frac{b}{n}\right) \neq b^{\frac{(n\ -1)}{2}}\ \ (mod\ n)$$

Proof:

If $p^2\,|n$, p being prime, let g generate $\left(\boldsymbol{Z}/p^2\boldsymbol{Z}\right)^*$. Select a b such that $b = g\left(mod\ p^2\right)$ and $b = 1\ (mod\ q)$ for any other distinct prime q dividing n. The Chinese Remainder theorem assures the existence of b. If the equation were true, then

$$b^{n-1} = 1\ (mod\ n)$$

which being a congruence remaining true in $\left(\boldsymbol{Z}/p^2\boldsymbol{Z}\right)^*$, would imply that $p(p-1)|(n-1)$. However, then p would divide both n and $n-1$.

So, we can suppose n is square free. Let g be a quadratic non-residue in some $(\boldsymbol{Z}/p\boldsymbol{Z})^*$ where $p|n$. Select a b, again, by Chinese Remainder theorem, such that $b = g\ (mod\ p)$ and $b =$

$1\ (mod\ q)$ for any other prime q that divides n. Then it is impossible for $b^{\frac{n-1}{2}} = -1\ (mod\ n)$ else this would be true in $(\boldsymbol{Z}/q\boldsymbol{Z})^*$ for those primes where b is one.

However, the Jacobi symbol gives

$$\left(\frac{b}{n}\right) = \left(\frac{b}{p}\right)\left(\prod_{q|n}\left(\frac{b}{q}\right)\right) = -1$$

We used the fact that n is odd in two places, that there are quadratic non-residues and that $-1\ \neq 1\ (mod\ q)$

Theorem 17.9.7.3 If n is odd composite, then for at least half of the integers b relatively prime to n in the interval $[1, n-1]$,

$$\left(\frac{b}{n}\right) \neq b^{\frac{(n\ -1)}{2}}\ (mod\ n)$$

Proof:

Let A be the set of a for which $(a, n) =\ 1$ and the equality holds. Since there is a b which is relatively prime to n for which the equality does not hold, take any $a\ \in A$ and consider ab. It is again relatively prime to n and,

$$\left(\frac{a}{n}\right)\left(\frac{b}{n}\right) \neq a^{\frac{(n\ -1)}{2}}\ b^{\frac{(n\ -1)}{2}}\ (mod\ n)$$

This is because we are inside the group $(\boldsymbol{Z}/n\boldsymbol{Z})^*$. Hence, we have that all of bA does not satisfy the equality, and hence A cannot account for more than half the elements in $[1, n\ -1]$.

17.9.8 The Miller-Rabin Test

This is the second probabilistic primality test. It is based on the concept of strong pseudoprime (Section 17.9.6.3)

Algorithm

```
$MILLER - RABIN(n)$

$choose\ a\ at\ random\ from\ \{1,2, \ldots., n-1\}$

$express\ n-1\ =\ 2^s * d, d\ odd$
```

```
compute successively $a_0 = a^d \ (mod\ n), a_1 = a_0^2 \ (mod\ n), \ldots, a_k = a_{k-1}^2 \ (mod\ n)$
until $k = s$ or $a_k \equiv 1 \ (mod\ n)$
if $( k = s)$ and $a_k \neq 1 \ (mod\ n)$ then return 'composite'
    elseif $(k = 0)$ then return 'prime'
        else if $a_{k-1} \neq -1$, then return 'composite'
            else return 'prime'
```

Similar to the Solovay Strassen test, this test has to be run k times. If it replies 'composite' in any of the iterations, the number is indeed composite. If it replies 'not composite' in all the k iterations, the probability of the number being composite is atmost $(1/4)^k$.

Proof of criterion in 17.9.6.3:

Assume n is prime. Then, if $a \in (\boldsymbol{Z}/n\boldsymbol{Z})^*$, we have $a^{n-1} \equiv 1 \ (mod\ n)$

Hence, either $a^{\frac{n-1}{2}} \equiv -1 \ (mod\ n)$ or $a^{\frac{n-1}{2}} \equiv 1 \ (mod\ n)$. In the former case, we are done. In the latter case, we again have $a^{\frac{n-1}{4}} \equiv \pm 1 \ (mod\ n)$. Proceeding like this, eventually either $a^d \equiv 1 \ (mod\ n), or\ a^{2^r.d} \equiv -1 \ (mod\ n) for\ some\ r, 0 \leq r < s$.

Now, assume n is composite. We know that $n - 1 = 2^s.d$, d being odd. Express $n = p_1^{e_1} \ldots . p_j^{e_j}$. Let k be the largest integer such that there exists at least one $b \in (\boldsymbol{Z}/n\boldsymbol{Z})^*$ with $b^{2^k} \equiv -1 \ (mod\ n)$. k is well defined, which is clear from $(-1)^{2^0} \equiv -1 \ (mod\ n)$, so $k \geq 0$.

By lemma 17.9.5.8, $p_i \equiv 1 \left(mod\ 2^{k+1}\right) \forall i, 1 \leq i \leq j$.

So, $n \equiv 1 \left(mod\ 2^{k+1}\right)$.

Let $m = 2^k.d$. Then, $2m | n - 1$. Now, consider the following chain of subgroups:

$$\begin{tikzcd}
(\boldsymbol{Z}/n\boldsymbol{Z})^* \arrow[d, dash] \\
J = \{a \in (\boldsymbol{Z}/n\boldsymbol{Z})^* \ : a^{n-1} \equiv 1\ (mod\ n)\} \arrow[d, dash] \\
K = \{a \in (\boldsymbol{Z}/n\boldsymbol{Z})^* \ : a^{m} \equiv \pm 1 \left(mod\ p_i^{e_i}\right) \forall i\} \arrow[d, dash] \\
L = \{a \in (\boldsymbol{Z}/n\boldsymbol{Z})^* \ : a^{m} \equiv \pm 1\ (mod\ n)\} \arrow[d, dash] \\
M = \{a \in (\boldsymbol{Z}/n\boldsymbol{Z})^* \ : a^{m} \equiv 1\ (mod\ n)\}
\end{tikzcd}$$

Each of these sets is actually a group. Also, every strong liar in $(\boldsymbol{Z}/n\boldsymbol{Z})^*$ is a member of L, since if $a^d \equiv 1\ (mod\ n)$, then $a^m \equiv 1\ (mod\ n)$, while if $a^{2^t.d} \equiv -1\ (mod\ n)$ for some t, then $t \leq k$ by the definition of k. We will show that, provided $n \neq 9$, L is a subgroup of index atleast 4 in $(\boldsymbol{Z}/n\boldsymbol{Z})^*$. From this it will follow that $|S(n)| \leq \frac{n-1}{4}$

Every element of $G = \{a \in (\boldsymbol{Z}/n\boldsymbol{Z})^* \ : a \equiv \pm 1 \left(mod\ p_i^{e_i}\right) \forall i\}$ is a 2^k-th power; hence an m^{th} power. (Putting $x \equiv b$ or $x \equiv b^2 \left(mod\ p_i^{e_i}\right)$ results in $x^{2^k} \equiv \pm 1 \left(mod\ p_i^{e_i}\right) \forall i$. Hence M has index 2^j in

$$K = \{a \in (\boldsymbol{Z}/n\boldsymbol{Z})^* \ : a^m \in G\}$$

Similarly, M has index 2 in L. Thus $(K{:}\,L) = 2^{j-1}$.

From the diagram,

$$((\boldsymbol{Z}/n\boldsymbol{Z})^*{:}\,L) \geq ((\boldsymbol{Z}/n\boldsymbol{Z})^* : J)\ (K{:}\,L) = 2^{j-1}((\boldsymbol{Z}/n\boldsymbol{Z})^*{:}\,J)$$

If $j \geq 3$, then $((\boldsymbol{Z}/n\boldsymbol{Z})^*{:}\,L) \geq 4$

If $j = 2$, then by lemma 17.9.5.9, n cannot be Carmichael. Thus there exists

$a \in (\boldsymbol{Z}/n\boldsymbol{Z})^*$ with $a^{n-1} \neq 1\ (mod\ n)$; thus J is a proper subgroup of $(\boldsymbol{Z}/n\boldsymbol{Z})^*$ and so $((\boldsymbol{Z}/n\boldsymbol{Z})^* : J) \geq 2$. Thus $((\boldsymbol{Z}/n\boldsymbol{Z})^* : L) \geq 4$.

If $j = 1$, then $n = p^e, e \geq 2$. But then $|J| = p - 1$, so $((\boldsymbol{Z}/n\boldsymbol{Z})^* : J) = p^{e-1} \geq 4$, except when $p^e = 9$. When $n = 9$, there are only 2 strong liars: 1 and -1, and hence the number of strong liars is $\leq \frac{n-1}{4}$

17.9.9 Conclusion

Now that we have discussed two probabilistic primality tests, which one should be used in practice? Both of them run in $O((\log n)^3)$ time. However, one should probably prefer the Miller-Rabin algorithm because

- Jacobi symbol is harder to program. Since the Solovay-Strassen algorithm involves computation using Jacobi symbol, it is harder to program.
- The error probability of Miller-Rabin algorithm is bounded by $\frac{1}{4}$ while that of Solovay-Strassen is bounded by $\frac{1}{2}$. This implies that a specific error bound can be achieved by testing half the number of elements in Miller-Rabin algorithm when compared with the Solovay-Strassen algorithm.
- Any Euler witness is also a strong witness to the compositeness of n.

In 2002, a new algorithm (The AKS algorithm) was found. This is a deterministic, general, polytime, unconditional algorithm that determines if the input number is prime or composite. This proved that primes belong to the complexity class P. Though a deterministic algorithm has been found, one might still want to use probabilistic tests to generate potential candidates for the AKS algorithm. Till date, the best version of this algorithm has been found to have a running time of $O((\log n)^6)$. So the probabilistic tests are much faster and can be used to generate potential candidates for the AKS algorithm to work on.

References

[1] E. Bach, and J. Shallit. Algorithmic Number Theory (Volume I: Efficient Algorithms). Published by *MIT Press*, 1996.

[2] B. Rosenberg. The Solovay Strassen Primality Test. Available online at http://www.cs.miami.edu/~burt/learning/Csc609.011/jacobi.pdf.

[3] Kenneth H. Rosen. Elementary Number Theory and Its Applications. Published by *Addison-Wesley Publishing Company*, 1986.

[4] P. Garrett. Intro to Abstract Algebra. Available online at http://www.math.umn.edu/~garrett/m/intro_algebra/notes.pdf.