MAT482: Miller-Rabin Primality Test

Andrew Feng

November 2021

1 Introduction

This is an expository project on the Miller-Rabin Primality Test[Rab80] written as part of the course *Algorithms and Number Theory and Algebra* at the University of Toronto during fall 2021, taught by professor Swastik Kopparty.

The Miller-Rabin primality test is a randomized probabilistic primality test such that if the input is prime, the test always says prime, and if the input is composite, the test finds out with high probability in time polynomial in the number of input bits. On input n, the test works by randomly generating numbers from 1 to n and testing if they are certificates to the compositeness (this will be made more precise later) of n. If any of the numbers generated is a certificate, then the algorithm will halt and assert n as composite. It turns out that for composite numbers, the number of such certificate is large. So it is unlikely for the randomly generated numbers to all not be certificates.

In the second section, we will introduce the algorithm and analyze its running time. Then, in the third section, we will give a proof of the efficiency of the algorithm. Finally, we will see some benchmarks of the algorithm running in practice in the fourth section.

2 Algorithm

Let $p(n)$ be a polynomial in n.

TEST(N)

1 Pick $1 < b_1, \dots, b_{p(\log N)} < N$ uniformly random
2 **for** $i = 1, \dots, p(\log N)$
3 Check if $(b_i)^{N-1} \not\equiv 1 \mod N$, return NO if so
4 **for** $j = e, \dots, 0$ where e is the multiplicity of 2 in $N-1$'s factorization
5 Check if $1 < ((b_i)^{(N-1)/2^j} - 1, N) < N$, return NO if so
6 Return YES

3 Analysis

First, we will look at the running time of TEST(N). For this, our complexity measure will be the number of bit operations.

Line 1 randomly draws $p(\log N)$ numbers that are at most $\log N$ bits, which can be done in $p(\log N) \log N$ time. Line 3 can be done by the method of repeated squaring in $\log N$ steps on $\log N$ bit integers, taking up $\log N(\log N)^2 = (\log N)^3$ time using elementary school integer multiplication.

On line 4, the number e can be calculated once using at most $\log N$ bitwise right shift operations, taking $(\log N)^2$ steps at most. Notice that once e has been calculated, line 5 only involves squaring a number of $\log N$ bits after the first iteration and the gcd algorithm on $\log N$ bit integers. At most $(\log N)^3$ steps are required to calculate $(b_i)^{(N-1)/2^e}$, $\log N$ steps to calculate the gcd, and $e \leq \log N$. So the inner loops takes up at most $(\log N)^3$ operations.

Thus, each iteration of the outer loop runs for $(\log N)^3$ steps at most, and so the algorithm runs in $O(p(\log N)(\log N)^3)$ steps[1]. If we choose p to be a constant k, then the running time becomes $k(\log N)^3$.

Hence, the Miller-Rabin primality test is a polynomial time algorithm in the number of input bits.

Now, we will investigate the reliability of the primality test by looking at the probability the algorithm is correct on any input. Let n be the integer whose primality is in question (rather than N above).

Definition 1. Let n be an integer. An integer $1 \leq b < n$ is called compositeness certificate of n if

$$b^{n-1} \not\equiv 1 \mod n, \text{ or} \tag{1}$$

$$\exists i : 2^i | (n-1) \text{ and } 1 < (b^{(n-1)/2^i} - 1, n) < n. \tag{2}$$

Let E_n be the multiplication group mod n.

Lemma 1. *Let $n, m_1, \ldots, m_k$ be integers such that each $m_i | n$ and the m_i's are pairwise coprime. Then, for every choice of $s_1 \in E_{m_1}, \ldots, s_k \in E_{m_k}$, the number of b's such that $b \in E_n$ and $b \equiv s_i \mod m_i$ for all i is the same.*

Proof. Given any $b \in E_n$, we know that $b \bmod m_i \in E_{m_i}$. Consider the map $f : E_n \to E_{m_1} \times \ldots \times E_{m_k}$ by $f(b) = (b \bmod m_1, \ldots, b \bmod m_k)$. This is a homomorphism of groups because it is multiplicative. Now, it only remains to show that f is surjective since the preimage of any element of a surjective homomorphism has the same size.

Fix $s_1 \in E_{m_1}, \ldots, s_k \in E_{m_k}$. We have two cases.

1. Each primes dividing n divides some m_i.

   In this case, the Chinese Remainder Theorem gives us some $b < n$ such that $b \bmod m_i = s_i$ for all $1 \leq i \leq k$. Note that this means b is coprime to each m_i. Thus, b is coprime to n and so $b \in E_n$.

2. Some primes dividing n does not divide any m_i.

   Then, let t be the product of all prime factors of n that do not divide any m_i's. Then, by the Chinese Remainder Theorem again, we can find $b < n$ such that $b \bmod m_i = s_i$ for all $1 \leq i \leq k$ and $b \bmod t = 1$. Similar to above, this means b is coprime to all prime factors of n, so $b \in E_n$.

This completes the proof. □

[1]To be pedantic, one should count in the time taken to compute $p(\log N)$. But it should be clear that any polynomial can be evaluated in time polynomial to the number of input bits ($\log \log N$ in this case). So it is negligible.

Lemma 2. *Let $p_1 \neq p_2$ be primes, $q_1 = p_1^{k_1}$, $q_2 = p_2^{k_2}$, and both q_1, q_2 divide n. Define $t_i = (\phi(q_i), n-1)$ and $m_i = \phi(q_i)/t_i$. Then, at most $\phi(q_i)/m_1m_2$ of the integers in E_n are not compositeness certificates of n. Furthermore, if t_1 is even, then at most $\phi(n)/2m_1m_2$ of integers in E_n are not compositeness certificates.*

Proof. To prove the first assertion, we will considers $b \in E_n$ such that $b^{n-1} \equiv 1 \mod n$. Since E_{q_1}, E_{q_2} are cyclic, let a_1, a_2 be the generators for E_{q_1}, E_{q_2} respectively. Write $b \equiv a_1^{r_1} \bmod q_1$ and $b \equiv a_2^{r_2} \bmod q_2$ (take $r_1, r_2 \geq 0$ as small as possible). Then, we see that

$$\begin{aligned} & b^{n-1} \equiv 1 \mod n \\ & \implies \phi(q_i) | r_i(n-1) \\ & \implies m_i = \left.\frac{\phi(q_i)}{(\phi(q_i), n-1)}\right| r_i \quad \text{for } i = 1, 2. \end{aligned}$$

This means at most

$$\phi(q_1)\phi(q_2)\frac{1}{m_1m_2} = t_1t_2$$

out of all pairs of $r_1 < \phi(q_1), r_2 < \phi(q_2)$ have some $b \in E_n$ with

$$\begin{aligned} & b \equiv a_1^{r_1} \mod q_1, \\ & b \equiv a_2^{r_2} \mod q_2, \text{ and} \\ & b^{n-1} \equiv 1 \mod n. \end{aligned}$$

Note that there is a bijection between powers r_i and remainders s_i in the previous lemma. So by the previous lemma we can conclude there are at most

$$t_1t_2\frac{\phi(n)}{\phi(q_1)\phi(q_2)} = \phi(n)/m_1m_2$$

integers $b \in E_n$ satisfying $b^{n-1} \equiv 1 \mod n$ (i.e are not compositeness certificates).

Now, suppose t_1 is even (note that the problem is symmetric in q_1 and q_2). We will show that even among the $b \in E_n$ such that $b^{n-1} \equiv 1 \mod n$, half of them are still satisfy

$$\exists i : 2^i | (n-1) \text{ and } 1 < (b^{(n-1)/2^i} - 1, n) < n.$$

(<u>Case 1</u>): t_1 and t_2 have the same number of 2 in their factorization. Then, let $i \in \mathbb{N}$ be such that $(n-1)/2^i$ is an integer, $t_j \nmid (n-1)/2^i$, but $t_j | (n-1)/2^{i-1}$, $j = 1, 2$. The set of valid powers r_j above in E_{q_j} is from 0 to $\phi(q_j) - 1$, but only multiples of m_j are allowed because $b^{n-1} \equiv 1 \mod n$. Thus, the r_j/m_j can take on is $0, 1, \ldots, t_j - 1$. By the previous lemma again, this means for any $b \in E_n$ such that $b^{n-1} \equiv 1 \mod n$, there is probability 1/4 such that r_1/m_1 is odd and r_2/m_2 is even (1/2 for one quotient to be even and the other odd). We argue in this case that

$$2^i | (n-1) \text{ and } 1 < (b^{(n-1)/2^i} - 1, n) < n$$

is satisfied.

Indeed, suppose WLOG that r_1/m_1 is odd but r_2/m_2 is even. Then,

$$\begin{aligned} & t_1 \nmid (n-1)/2^i \quad \text{and} \quad t_1 | (n-1)/2^{i-1} \\ & \implies t_1 \nmid \frac{r_1}{m_1}(n-1)/2^i \\ & \implies \phi(q_1) \nmid \frac{r_1}{m_1}\frac{\phi(q_1)}{t_1}(n-1)/2^i = r_1(n-1)/2^i. \end{aligned}$$

And

$$
\begin{aligned}
&t_2|(n-1)/2^{i-1} \\
&\implies t_2|\frac{r_2}{m_2}(n-1)/2^i \quad \text{because } \frac{r_2}{m_2} > 0 \text{ is even} \\
&\implies \phi(q_2)|r_2(n-1)/2^i.
\end{aligned}
$$

This gives us

$$
b^{(n-1)/2^i} \not\equiv 1 \mod q_1 \tag{3}
$$

$$
b^{(n-1)/2^i} \equiv 1 \mod q_2. \tag{4}
$$

Thus, we know that $q_2 \leq (b^{(n-1)/2^i} - 1, n) < n$.

(CASE 2): t_1 has more factors of 2 than t_2. We can choose i such that $t_2|(n-1)/2^i$ but $t_1 \nmid (n-1)/2^i$, and choose the minimal j such that $t_1|(n-1)/2^{i-j}$. Then, by the same argument above, we have $b^{(n-1)/2^i} \equiv 1 \mod q_2$. However, $b^{(n-1)/2^i} \equiv 1 \mod q_1$ happens only if $\phi(q_1)|r_1(n-1)/2^i$. This implies $2^j|r_1/m_1$. This is only true for at most $\phi(n)/(2^j m_1 m_2)$ of $b \in E_n$. Note that $j \geq 1$. □

Theorem 1. *If $n > 4$ is composite, then the number of compositeness certificates of n is at least $3(n-1)/4$.*

Proof. Since for $b < n$ outside of E_n, we already have $b^{n-1} \not\equiv 1 \mod n$, we only need to show at most $1/4$ of numbers in E_n are not compositeness certificate.

The first case is $n = p^k$ for some prime p. Then, we have $\phi(n) = p^{k-1}(p-1)$. Since $p \nmid n-1$, the gcd $(\phi(n), n-1)$ must divide $p-1$. Set $m = \phi(n)/((\phi(n), n-1))$, we have $p^{k-1} \leq m$. Using the same argument as the beginning of previous lemma, we can show that at most $\phi(n)/m$ of $b \in E^n$ are not compositeness certificates. One can check manually that the theorem holds for $n \leq 9$. As for $n > 9$, we must have $k > 2$ or $p \geq 5$. In either case, we have $m \geq 4$ and thus $\phi(n)/m \leq \phi(n)/4$.

Now, suppose n has at least two different prime divisors. Write $n = p_1^{k_1} \dots p_r^{k_r}$, $r \geq 2$ and p_i are primes. There are three subcases to consider.

The first subcase is when there are $1 \leq i < j \leq r$ with $\phi(p_i^{k_i}) \nmid (n-1)$ and $\phi(p_j^{k_j}) \nmid (n-1)$. In this case, we have $m_i = \phi(p_i^{k_i})/((\phi(p_i^{k_i}), n-1)) \geq 2$ and $m_j = \phi(p_j^{k_j})/((\phi(p_j^{k_j}), n-1)) \geq 2$. By lemma 2, we are done because at most $\phi(n)/m_i m_j$ of $b \in E^n$ are not compositeness certificates.

The second subcase is when some $1 \leq i \leq r$ has $\phi(p_i^{k_i}) \nmid (n-1)$ but for any other $1 \leq j \leq r$, this is false. We still have $m_i \geq 2$. If $p_j \neq 2$, then $t_j = \phi(p_j^{k_j})$ is even. So we are done by the second claim of lemma 2. If $p_j = 2$, then we have the estimate $\phi(n) \leq (n-1)/2$. Then, the bound $\phi(n)/m_1$ implies $(n-1)/4$—so we are done once again.

The third subcase is when for all $1 \leq i \leq r$ has $\phi(p_i^{k_i})|(n-1)$. Then, notice two things. First, each power $r_i = 1$ because $p_i \nmid (n-1)$. Second, $r \geq 3$. This is because if $n = pq$ for primes $p < q$. Then, if $(q-1)|(pq-1)$, we have

$$
\frac{pq-1}{q-1} = \frac{pq-p+p-1}{q-1} = p + \frac{p-1}{q-1}
$$

is an integer, a contradiction since $p < q$.

Thus, we have $n = p_1 \dots p_r$ with $r \geq 3$. In this case, all $m_i = 1$. Let $e(p_i - 1)$ denote the number of factors of 2 in $p_i - 1$. WLOG, suppose $e(p_1 - 1) \leq e(p_2 - 1) \leq e(p_3 - 1)$. If $e(p_1 - 1) = e(p_2 - 1)$, then we can use the argument in lemma 2 (CASE 2) to see that at most half of the b in E_n are not compositeness certificates. Similarly, if $e(p_1 - 1) < e(p_2 - 1)$, we can use argument in lemma

2 (Case 1) to see that at most 1/2 of the b in E_n are not compositeness certificates. This can be similarly done for p_2 and p_3. Multiplying, we get that at most 1/4 of $b \in E_n$ are not compositeness certificates (we can multiply because of lemma 1!). This completes the proof. □

The implication of the proof is clear in regard to the primality test. Since all we are doing in the algorithm is to find a compositeness certificate, the theorem implies that the probability of a set of random points less than n contain no certificate is low for a composite n. More specifically, if k integers are independently sampled from $[1, n-1]$ with uniform distribution with n begin composite, then with $1 - (1/2)^{2n}$ probability the algorithm has a compositeness certificate!

4 Practical Use

The Miller-Rabin primality is easy to implement on a computer as it only requires gcd (this is in fact not required in an improved version of Miller-Rabin) and arithmetic modulo n. I have implemented the algorithm fully using `C++` with the `Boost::multiprecision` library. The library supports integers of arbitrary magnitude. This gives Miller-Rabin an advantage over other algorithms such as AKS which involve arithmetic in polynomial rings rather than simply integers mod n.

Most importantly, the Miller-Rabin primality test runs efficiently on real computers. The algorithm is blazing fast at proving whether an integer is composite:

```
Testing whether 4951760154835678088235319297 is prime...
    Time taken for naive algorithm: 294113 milliseconds
    Time taken for miller-rabin algorithm: 0 milliseconds
```

Figure 1: Captured when testing whether the product $(2^{31} - 1)(2^{61} - 1)$ of Mersenne primes.

To see the source code and learn more about the actual implementation, please check out my Github repository (it also contains an implementation of the FFT algorithm);

References

[Rab80] Michael O Rabin. "Probabilistic algorithm for testing primality". In: *Journal of Number Theory* 12.1 (1980), pp. 128–138. ISSN: 0022-314X. DOI: https://doi.org/10.1016/0022-314X(80)90084-0. URL: https://www.sciencedirect.com/science/article/pii/0022314X80900840.