Miller–Rabin primality test

From Wikipedia, the free encyclopedia

The **Miller–Rabin primality test** or **Rabin–Miller primality test** is a primality test: an algorithm which determines whether a given number is prime, similar to the Fermat primality test and the Solovay–Strassen primality test. Its original version, due to Gary L. Miller, is deterministic, but the determinism relies on the unproven Extended Riemann hypothesis;[1] Michael O. Rabin modified it to obtain an unconditional probabilistic algorithm.[2]

Contents

Concepts

Just like the Fermat and Solovay–Strassen tests, the Miller–Rabin test relies on an equality or set of equalities that hold true for prime values, then checks whether or not they hold for a number that we want to test for primality.

First, a lemma about square roots of unity in the finite field $\mathbf{Z}/p\mathbf{Z}$, where p is prime and $p > 2$. Certainly 1 and −1 always yield 1 when squared modulo p; call these trivial square roots of 1. There are no *nontrivial* square roots of 1 modulo p (a special case of the result that, in a field, a polynomial has no more zeroes than its degree). To show this, suppose that x is a square root of 1 modulo p. Then:

$$x^2 \equiv 1 \pmod p$$
$$(x-1)(x+1) \equiv 0 \pmod p.$$

In other words, prime p divides the product $(x − 1)(x + 1)$. By Euclid's lemma it divides one of the factors $x − 1$ or $x + 1$, implying that x is congruent to either 1 or −1 modulo p.

Now, let n be prime with $n > 2$. It follows that $n − 1$ is even and we can write it as $2^s\cdot d$, where s and d are positive integers and d is odd. For each a in $(\mathbf{Z}/n\mathbf{Z})^*$, either

$$a^d \equiv 1 \pmod n$$

or

$$a^{2^r\cdot d} \equiv -1 \pmod n$$

for some $0 \le r \le s − 1$.

To show that one of these must be true, recall Fermat's little theorem, that for a prime number n:

$$a^{n-1} \equiv 1 \pmod n.$$

By the lemma above, if we keep taking square roots of $a^{n−1}$, we will get either 1 or −1. If we get −1 then the second equality holds and it is done. If we never get −1, then when we have taken out every power of 2, we are left with the first equality.

The Miller–Rabin primality test is based on the contrapositive of the above claim. That is, if we can find an a such that

$$a^d \not\equiv 1 \pmod n$$

and

$$a^{2^r d} \not\equiv -1 \pmod n$$

for all $0 \le r \le s − 1$, then n is not prime. We call a a witness for the compositeness of n (sometimes misleadingly called a *strong witness*, although it is a certain proof of this fact). Otherwise a is called a *strong liar*, and n is a strong probable prime to base a. The term "strong liar" refers to the case where n is composite but nevertheless the equations hold as they would for a prime.

Every odd composite n has many witnesses a, however, no simple way of generating such an a is known. The solution is to make the test probabilistic: we choose a non-zero a in $\mathbf{Z}/n\mathbf{Z}$ randomly, and check whether or not it is a witness for the compositeness of n. If n is composite, most of the choices for a will be witnesses, and the test will detect n as composite with high probability. There is, nevertheless, a small chance that we are unlucky and hit an a which is a strong liar for n. We may reduce the probability of such error by repeating the test for several independently chosen a.

For testing large numbers, it is common to choose random bases a, as, a priori, we don't know the distribution of witnesses and liars among the numbers 1, 2, ..., $n − 1$. In particular, Arnault [3] gave a 397-digit composite number for which all bases a less than 307 are strong liars. As expected this number was reported to be prime by the Maple `isprime()` function, which implemented the Miller–Rabin test by checking the specific bases 2,3,5,7, and 11. However, selection of a few specific small bases can guarantee identification of composites for n less than some maximum determined by said bases. This maximum is generally quite large compared to the bases. As random bases lack such determinism for small n, specific bases are better in some circumstances.

Example

Suppose we wish to determine if $n = 221$ is prime. We write $n − 1 = 220$ as $2^2\cdot 55$, so that we have $s = 2$ and $d = 55$. We randomly select a number a such that $1 < a < n - 1$, say $a = 174$. We proceed to compute:

- $a^{2^0\cdot d} \bmod n = 174^{55} \bmod 221 = 47 \neq 1, n − 1$
- $a^{2^1\cdot d} \bmod n = 174^{110} \bmod 221 = 220 = n − 1$.

Since $220 \equiv −1 \bmod n$, either 221 is prime, or 174 is a strong liar for 221. We try another random a, this time choosing $a = 137$:

- $a^{2^0\cdot d} \bmod n = 137^{55} \bmod 221 = 188 \neq 1, n − 1$
- $a^{2^1\cdot d} \bmod n = 137^{110} \bmod 221 = 205 \neq n − 1$.

Hence 137 is a witness for the compositeness of 221, and 174 was in fact a strong liar. Note that this tells us nothing about the factors of 221 (which are 13 and 17). However, the example with 341 in the next section shows how these calculations can sometimes produce a factor of n.

Computational complexity

The algorithm can be written in pseudocode as follows:

```
Input #1: n > 3, an odd integer to be tested for primality;
Input #2: k, a parameter that determines the accuracy of the test
Output: composite if n is composite, otherwise probably prime
```

```
write n − 1 as 2^r·d with d odd by factoring powers of 2 from n − 1
WitnessLoop: repeat k times:
   pick a random integer a in the range [2, n − 2]
   x ← a^d mod n
   if x = 1 or x = n − 1 then
      continue WitnessLoop
   repeat r − 1 times:
      x ← x^2 mod n
      if x = n − 1 then
         continue WitnessLoop
   return composite
return probably prime
```

Using modular exponentiation by repeated squaring, the running time of this algorithm is $O(k \log^3 n)$, where k is the number of different values of a that we test; thus this is an efficient, polynomial-time algorithm. FFT-based multiplication can push the running time down to $O(k \log^2 n \log\log n \log\log\log n) = \tilde{O}(k \log^2 n)$.

If we insert Greatest common divisor calculations into the above algorithm, we can sometimes obtain a factor of n instead of merely determining that n is composite. In particular, if n is a probable prime base a but not a strong probable prime base a, then either $\mathrm{GCD}((a^d \bmod n) - 1, n)$ or (for some r in the above range) $\mathrm{GCD}((a^{2^r\cdot d} \bmod n) - 1, n)$ will produce a (not necessarily prime) factor of n.[4]:1402 If factoring is a goal, these GCDs can be inserted into the above algorithm at little additional computational cost.

For example, consider $n = 341$. We have $n - 1 = 85\cdot 4$. Then $2^{85} \bmod 341 = 32$. This tells us that n is not a strong probable prime base 2, so we know n is composite. If we take a GCD at this stage, we can get a factor of 341: $\mathrm{GCD}((2^{85} \bmod 341) - 1, 341) = 31$. This works because 341 is a pseudoprime base 2, but is not a strong pseudoprime base 2.

In the case that the algorithm returns "composite" because $x = 1$, it has also discovered that $d2^r$ is (an odd multiple of) the order of a—a fact which can (as in Shor's algorithm) be used to factorize n, since n then divides

$$a^{d2^r} - 1 = (a^{d2^{r-1}} - 1)(a^{d2^{r-1}} + 1)$$

but not either factor by itself. The reason Miller–Rabin does *not* yield a probabilistic factorization algorithm is that if

$$a^{n-1} \not\equiv 1 \pmod n$$

(i.e., n is not a pseudoprime to base a) then no such information is obtained about the period of a, and the second "return composite" is taken.

Accuracy

The more bases a we test, the better the accuracy of the test. It can be shown that for any odd composite n, at least 3/4 of the bases a are witnesses for the compositeness of n.[2][5] The Miller–Rabin test is strictly stronger than the Solovay–Strassen primality test in the sense that for every composite n, the set of strong liars for n is a subset of the set of Euler liars for n, and for many n, the subset is proper. If n is composite then the Miller–Rabin primality test declares n probably prime with a probability at most $4^{−k}$. On the other hand, the Solovay–Strassen primality test declares n probably prime with a probability at most $2^{−k}$.

It is important to note that in many common applications of this algorithm, we are not interested in the error bound described above. The above error bound is the probability of a composite number being declared as a probable prime after k rounds of testing. We are often instead interested in the probability that, after passing k rounds of testing, the number being tested is actually a composite number. Formally, if we call the event of declaring n a probable prime after k rounds of Miller–Rabin Y_k, and we call the event that n is composite X (and denote the event that n is prime $\overline{X}$), then the above bound gives us $P(Y_k|X)$, whereas we are interested in $P(X|Y_k)$. Bayes' theorem gives us a way to relate these two conditional probabilities, namely

$$P(X|Y_k) = \frac{P(Y_k|X)P(X)}{P(Y_k|X)P(X) + P(Y_k|\overline{X})P(\overline{X})}.$$

This tells us that the probability that we are often interested in is related not just to the $4^{−k}$ bound above, but also probabilities related to the density of prime numbers in the region near n.

In addition, for large values of n, on average the probability that a composite number is declared *probably prime* is significantly smaller than $4^{−k}$. Damgård, Landrock and Pomerance[6] compute some explicit bounds and provide a method to make a reasonable selection for k for a desired error bound. Such bounds can, for example, be used to *generate* probable primes; however, they should not be used to *verify* primes with unknown origin, since in cryptographic applications an adversary might try to send you a pseudoprime in a place where a prime number is required. In such cases, only the error bound of $4^{−k}$ can be relied upon.

Deterministic variants

The Miller–Rabin algorithm can be made deterministic by trying all possible a below a certain limit. The problem in general is to set the limit so that the test is still reliable.

If the tested number n is composite, the strong liars a coprime to n are contained in a proper subgroup of the group $(\mathbf{Z}/n\mathbf{Z})^*$, which means that if we test all a from a set which generates $(\mathbf{Z}/n\mathbf{Z})^*$, one of them must be a witness for the compositeness of n. Assuming the truth of the generalized Riemann hypothesis (GRH), it is known that the group is generated by its elements smaller than $O((\log n)^2)$, which was already noted by Miller.[1] The constant involved in the Big O notation was reduced to 2 by Eric Bach.[7] This leads to the following conditional primality testing algorithm:

```
Input: n > 1, an odd integer to test for primality.
Output: composite if n is composite, otherwise prime
write n−1 as 2^s·d by factoring powers of 2 from n−1
repeat for all a ∈ [2, min(n − 1, ⌊2(ln n)^2⌋)]:
     if a^d ≢ 1 (mod n) and a^(2^r·d) ≢ −1 (mod n) for all r ∈ [0, s − 1] then
        return composite
return prime
```

The running time of the algorithm is $\tilde{O}((\log n)^4)$ (with FFT-based multiplication). The full power of the generalized Riemann hypothesis is not needed to ensure the correctness of the test: as we deal with subgroups of even index, it suffices to assume the validity of GRH for quadratic Dirichlet characters.[5]

The Miller test (the algorithm above) is not used in practice. For most purposes, proper use of the probabilistic Miller-Rabin test or the Baillie-PSW primality test gives sufficient confidence while running much faster. It is also slower in practice than commonly used proof methods such as APR-CL and ECPP which give results that do not rely on unproven assumptions. For theoretical purposes requiring a deterministic polynomial time algorithm, it was superseded by the AKS primality test, which also does not rely on unproven assumptions.

Note that Miller-Rabin pseudoprimes are called strong pseudoprimes.

When the number n to be tested is small, trying all $a < 2(\ln n)^2$ is not necessary, as much smaller sets of potential witnesses are known to suffice. For example, Pomerance, Selfridge and Wagstaff[8] and Jaeschke[9] have verified that

- if n < 2,047, it is enough to test a = 2;
- if n < 1,373,653, it is enough to test a = 2 and 3;
- if n < 9,080,191, it is enough to test a = 31 and 73;
- if n < 25,326,001, it is enough to test a = 2, 3, and 5;
- if n < 3,215,031,751, it is enough to test a = 2, 3, 5, and 7;
- if n < 4,759,123,141, it is enough to test a = 2, 7, and 61;
- if n < 1,122,004,669,633, it is enough to test a = 2, 13, 23, and 1662803;
- if n < 2,152,302,898,747, it is enough to test a = 2, 3, 5, 7, and 11;
- if n < 3,474,749,660,383, it is enough to test a = 2, 3, 5, 7, 11, and 13;
- if n < 341,550,071,728,321, it is enough to test a = 2, 3, 5, 7, 11, 13, and 17.

Using the work of Feitsma and Galway enumerating all base 2 pseudoprimes in 2010, this was extended (see A014233), with the first result later shown using different methods in Jiang and Deng:[10]

- if n < 3,825,123,056,546,413,051, it is enough to test a = 2, 3, 5, 7, 11, 13, 17, 19, and 23.
- if n < 18,446,744,073,709,551,616 = 2^{64}, it is enough to test a = 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, and 37.

Sorenson and Webster[11] verify the above and calculate precise results for these larger than 64-bit results:

- if n < 318,665,857,834,031,151,167,461, it is enough to test a = 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, and 37.
- if n < 3,317,044,064,679,887,385,961,981, it is enough to test a = 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, and 41.

Other criteria of this sort, often more efficient (fewer bases required) than those shown above, exist[12][13][14][15] and these results give very fast deterministic primality tests for numbers in the appropriate range without any assumptions.

There is a small list of potential witnesses for every possible input size (at most n values for n-bit numbers). However, no finite set of bases is sufficient for all composite numbers. Alford, Granville, and Pomerance have shown that there exist infinitely many composite numbers n whose smallest compositeness witness is at least $(\ln n)^{1/(3\ln\ln\ln n)}$.[16] They also argue heuristically that the smallest number w such that every composite number below n has a compositeness witness less than w should be of order $\Theta(\log n \log\log n)$.

Notes

1. Miller, Gary L. (1976), "Riemann's Hypothesis and Tests for Primality", *Journal of Computer and System Sciences*, **13** (3): 300–317, doi:10.1145/800116.803773
2. Rabin, Michael O. (1980), "Probabilistic algorithm for testing primality", *Journal of Number Theory*, **12** (1): 128–138, doi:10.1016/0022-314X(80)90084-0
3. F. Arnault (August 1995). "Constructing Carmichael Numbers Which Are Strong Pseudoprimes to Several Bases". *Journal of Symbolic Computation*. **20** (2): 151–161. doi:10.1006/jsco.1995.1042.
4. Robert Baillie; Samuel S. Wagstaff, Jr. (October 1980). "Lucas Pseudoprimes" (PDF). *Mathematics of Computation*. **35** (152): 1391–1417. doi:10.1090/S0025-5718-1980-0583518-6. MR 583518.
5. Schoof, René (2004), "Four primality testing algorithms", *Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography* (PDF), Cambridge University Press, ISBN 0-521-80854-5
6. Damgård, I.; Landrock, P. & Pomerance, C. (1993), "Average case error estimates for the strong probable prime test" (PDF), *Mathematics of Computation*, **61** (203): 177–194, doi:10.2307/2152945
7. Bach, Eric (1990), "Explicit bounds for primality testing and related problems", *Mathematics of Computation*, **55** (191): 355–380, doi:10.2307/2008811
8. Carl Pomerance; John L. Selfridge; Samuel S. Wagstaff, Jr. (July 1980). "The pseudoprimes to $25\cdot 10^9$" (PDF). *Mathematics of Computation*. **35** (151): 1003–1026. doi:10.1090/S0025-5718-1980-0572872-7.
9. Jaeschke, Gerhard (1993), "On strong pseudoprimes to several bases", *Mathematics of Computation*, **61** (204): 915–926, doi:10.2307/2153262
10. Jiang, Yupeng; Deng, Yingpu (2014). "Strong pseudoprimes to the first eight prime bases". *Mathematics of Computation*. **83** (290): 2915–2924. doi:10.1090/S0025-5718-2014-02830-5.
11. Sorenson, Jonathan; Webster, Jonathan (2015). "Strong Pseudoprimes to Twelve Prime Bases". arXiv:1509.00864 [math.NT].
12. *The Prime Pages*
13. Zhang, Zhenxiang & Tang, Min (2003), "Finding strong pseudoprimes to several bases. II", *Mathematics of Computation*, **72** (44): 2085–2097, doi:10.1090/S0025-5718-03-01545-X
14. "Sloane's A014233 : Smallest odd number for which Miller-Rabin primality test on bases <= n-th prime does not reveal compositeness". *The On-Line Encyclopedia of Integer Sequences*. OEIS Foundation.
15. *SPRP Records*
16. Alford, W. R.; Granville, A.; Pomerance, C. (1994), "On the difficulty of finding reliable witnesses" (PDF), *Lecture Notes in Computer Science*, Springer-Verlag, **877**: 1–16, doi:10.1007/3-540-58691-1_36

External links

- Weisstein, Eric W. "Rabin-Miller Strong Pseudoprime Test". *MathWorld*.
- Interactive Online Implementation of the Deterministic Variant (http://gandraxa.com/miller_rabin_primality_test.xml) (stepping through the algorithm step-by-step)
- Applet (German) (http://www.informationsuebertragung.ch/indexAlgorithmenRabinMiller.html)
- Miller-Rabin primality test in C# (https://stackoverflow.com/questions/4236673/sample-code-for-fast-primality-testing-in-c-sharp/4236870#4236870)
- Miller-Rabin primality test in JavaScript using arbitrary precision arithmetic (http://www.javascripter.net/math/primes/millerrabinprimalitytest.htm)

The Wikibook *Algorithm Implementation* has a page on the topic of: ***Primality testing***

Retrieved from "https://en.wikipedia.org/w/index.php?title=Miller–Rabin_primality_test&oldid=748742058"

Categories: Primality tests | Finite fields

- This page was last modified on 10 November 2016, at 02:50.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.