Miller–Rabin test

The algorithm can be written in pseudocode as follows. The parameter k determines the accuracy of the test. The greater the number of rounds, the more accurate the result.[6]

```
Input #1: $n > 2$, an odd integer to be tested for primality
Input #2: $k$, the number of rounds of testing to perform
Output: “composite” if $n$ is found to be composite, “probably prime” otherwise

let $s > 0$ and $d$ odd $> 0$ such that $n - 1 = 2^s d$  # by factoring out powers of 2 from $n - 1$
repeat $k$ times:
    $a \leftarrow$ random(2, $n - 2$)  # $n$ is always a probable prime to base 1 and $n - 1$
    $x \leftarrow a^d$ mod $n$
    repeat $s$ times:
        $y \leftarrow x^2$ mod $n$
        if $y = 1$ and $x \neq 1$ and $x \neq n - 1$ then # nontrivial square root of 1 modulo $n$
            return “composite”
        $x \leftarrow y$
    if $y \neq 1$ then
        return “composite”
return “probably prime”
```

Complexity

Using repeated squaring, the running time of this algorithm is $O(k \log^3 n)$, where n is the number tested for primality, and k is the number of rounds performed; thus this is an efficient, polynomial-time algorithm. FFT-based multiplication (Harvey-Hoeven algorithm) can decrease the running time to $O(k \log^2 n \log \log n) = \tilde{O}(k \log^2 n)$.

Accuracy

The error made by the primality test is measured by the probability that a composite number is declared probably prime. The more bases a are tried, the better the accuracy of the test. It can be shown that if n is composite, then at most $\frac{1}{4}$ of the bases a are strong liars for n.[2][7] As a consequence, if n is composite then running k iterations of the Miller–Rabin test will declare n probably prime with a probability at most 4^{-k}.

This is an improvement over the Solovay–Strassen test, whose worst-case error bound is 2^{-k}. Moreover, the Miller–Rabin test is strictly stronger than the Solovay–Strassen 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.

In addition, for large values of n, the probability for a composite number to be declared probably prime is often significantly smaller than 4^{-k}. For instance, for most numbers n, this probability is bounded by 8^{-k}; the proportion of numbers n which invalidate this upper bound vanishes as we consider larger values of n.[8] Hence the *average* case has a much better accuracy than 4^{-k}, a fact which can be exploited for *generating* probable primes (see below). However, such improved error bounds should not be relied upon to *verify* primes whose probability distribution is not controlled, since a cryptographic adversary might send a carefully chosen pseudoprime in order to defeat the primality test.[c] In such contexts, only the *worst-case* error bound of 4^{-k} can be relied upon.

The above error measure is the probability for a composite number to be declared as a strong probable prime after k rounds of testing; in mathematical words, it is the conditional probability $\Pr(MR_k \mid \neg P)$ where P is the event that the number being tested is prime, and MR_k is the event that it passes the Miller–Rabin test with k rounds. We are often interested instead in the inverse conditional probability $\Pr(\neg P \mid MR_k)$: the probability that a number which has been declared as a strong probable prime is in fact composite. These two probabilities are related by Bayes' law:

$$\begin{aligned}\Pr(\neg P \mid MR_k) &= \frac{\Pr(\neg P \land MR_k)}{\Pr(\neg P \land MR_k) + \Pr(P \land MR_k)} \\ &= \frac{1}{1 + \frac{\Pr(MR_k|P)}{\Pr(MR_k|\neg P)} \frac{\Pr(P)}{\Pr(\neg P)}} \\ &= \frac{1}{1 + \frac{1}{\Pr(MR_k|\neg P)} \frac{\Pr(P)}{1-\Pr(P)}}\end{aligned}$$

In the last equation, we simplified the expression using the fact that all prime numbers are correctly reported as strong probable primes (the test has no false negative). By dropping the left part of the denominator, we derive a simple upper bound:

$$\Pr(\neg P \mid MR_k) < \Pr(MR_k \mid \neg P) \left(\tfrac{1}{\Pr(P)} - 1\right)$$

Hence this conditional probability is related not only to the error measure discussed above —which is bounded by 4^{-k}— but also to the probability distribution of the input number. In the general case, as said earlier, this distribution is controlled by a cryptographic adversary, thus unknown, so we cannot deduce much about $\Pr(\neg P \mid MR_k)$. However, in the case when we use the Miller–Rabin test to *generate* primes (see below), the distribution is chosen by the generator itself, so we can exploit this result.

Generation of probable primes

The Miller–Rabin test can be used to generate strong probable primes, simply by drawing integers at random until one passes the test. This algorithm terminates almost surely (since at each iteration there is a chance to draw a prime number). The pseudocode for generating b-bit strong probable primes (with the most significant bit set) is as follows:

```
Input #1: $b$, the number of bits of the result
Input #2: $k$, the number of rounds of testing to perform
Output: a strong probable prime $n$

while True:
    pick a random odd integer $n$ in the range $[2^{b-1}, 2^b-1]$
    if the Miller-Rabin test with inputs $n$ and $k$ returns “probably prime” then
        return $n$
```

Complexity

Of course the worst-case running time is infinite, since the outer loop may never terminate, but that happens with probability zero. As per the geometric distribution, the expected number of draws is $\frac{1}{\Pr(MR_k)}$ (reusing notations from earlier).

As any prime number passes the test, the probability of being prime gives a coarse lower bound to the probability of passing the test. If we draw odd integers uniformly in the range $[2^{b-1}, 2^b-1]$, then we get:

$$\Pr(MR_k) > \Pr(P) = \frac{\pi\left(2^b\right) - \pi\left(2^{b-1}\right)}{2^{b-2}}$$

where π is the prime-counting function. Using an asymptotic expansion of π (an extension of the prime number theorem), we can approximate this probability when b grows towards infinity. We find:

$$\begin{aligned}\Pr(P) &= \tfrac{2}{\ln 2} b^{-1} + \mathcal{O}\left(b^{-3}\right) \\ \tfrac{1}{\Pr(P)} &= \tfrac{\ln 2}{2} b + \mathcal{O}\left(b^{-1}\right)\end{aligned}$$

Hence we can expect the generator to run no more Miller–Rabin tests than a number proportional to b. Taking into account the worst-case complexity of each Miller–Rabin test (see earlier), the expected running time of the generator with inputs b and k is then bounded by $\mathrm{O}(k\,b^4)$ (or $\tilde{\mathrm{O}}(k\,b^3)$ using FFT-based multiplication).

Accuracy

The error measure of this generator is the probability that it outputs a composite number.

Using the relation between conditional probabilities (shown in an earlier section) and the asymptotic behavior of $\Pr(P)$ (shown just before), this error measure can be given a coarse upper bound:

$$\Pr(\neg P \mid MR_k) < \Pr(MR_k \mid \neg P)\left(\tfrac{1}{\Pr(P)} - 1\right) \leq 4^{-k}\left(\tfrac{\ln 2}{2} b - 1 + \mathcal{O}\left(b^{-1}\right)\right).$$

Hence, for large enough b, this error measure is less than $\frac{\ln 2}{2}4^{-k}b$. However, much better bounds exist.

Using the fact that the Miller–Rabin test itself often has an error bound much smaller than 4^{-k} (see earlier), Damgård, Landrock and Pomerance derived several error bounds for the generator, with various classes of parameters b and k.[8] These error bounds allow an implementor to choose a reasonable k for a desired accuracy.

One of these error bounds is 4^{-k}, which holds for all $b \geq 2$ (the authors only showed it for $b \geq 51$, while Ronald Burthe Jr. completed the proof with the remaining values $2 \leq b \leq 50$[20]). Again this simple bound can be improved for large values of b. For instance, another bound derived by the same authors is:

$$\left(\frac{1}{7}b^{\frac{15}{4}}2^{-\frac{b}{2}}\right)4^{-k}$$

which holds for all $b \geq 21$ and $k \geq b/4$. This bound is smaller than 4^{-k} as soon as $b \geq 32$.