

Technique for generating provable primes

Preda Mihăilescu

FingerPIN AG & ETH, Institut für wissenschaftliches Rechnen
EMail : mihailes@inf.ethz.ch, develop@fingerpin.com

Abstract. We suggest a technique for generating provable primes for cryptographical use, for the P1363 standard. The method not only provides a certificate for the primes generated, it is also *faster* than similar probabilistic generation algorithms. The security concerns are also covered. Detailed descriptions and analysis may be found in [Mi], [Mi1].

1 Algorithm Description

Algorithm AP (** Primes in Arithmetic Progressions **)

Let $m > 0$ be an integer: the number of bits required for a prime N , which is to be generated. Let $B > 0$ be a fixed bound, such that primes with less than B bits can be generated by extensive trial division - e.g. $B = 32$. Suppose a uniform source of random bits R is known to the algorithm. The following is a pseudo code description of a recursive procedure $N = \mathbf{AP}(m)$.

1. If $m \leq B$ return a random prime with m bits, proved by extensive trial division (and stop).
2. Let m' be an integer with $m/3 < m' < m/3 + m/h$, for a constant k allowing some uncertainty in the length m' - say, $h = 30$. Put $Q = \mathbf{AP}(m')$ (recursive call to \mathbf{AP}).
3. Use the random source R for generating a number t , such that

$$2^{m-1} < 2 \cdot t \cdot Q < 2^m.$$

4. Let D be a trial division bound. Criteria for optimal choices for D are described in [Mi]. For practical purposes, say $D = 2^{16}$. Multiply virtually all primes $p < D$ in products P_i of machine word length - 32 or 64 bits. Build the two tables:

$$T = \{2 \cdot t \cdot Q + 1 \bmod P_i : p < D\} \text{ and } F = \{2 \cdot Q \bmod P_i : p < D\}.$$

5. (Eratosthenes Sieve)

Initialize an array E of length $s \cdot m$ with the value 1, for s a fixed constant, upper bounding the interval search length for a prime, e.g. $s = 10$. For all the i do following:

- (i) Find all the solutions $0 \leq k < m \cdot s$ of the equation

$$T(i) + k \cdot Q(i) = 0 \pmod{P_i}.$$

- (ii) If k is a solution of the above equation, set $E(k) = 0$.

6. (Main prime search loop)

For $0 \leq k < s \cdot m$ do the following:

- (i) If $E(k) = 1$, put $N = 2 \cdot (t + k) \cdot Q + 1$ and check if $2^{N-1} = 1 \pmod{N}$. If this is not the case, goto (v).
- (ii) Find a base b ($b = 2, 3, 5, \dots$), such that $(b^{\frac{N-1}{Q}} - 1, N) = 1$ and $b^{N-1} = 1 \pmod{N}$.
- (iii) Put $f = 2(t+k)$ and $u = f \pmod{Q}$, $v = f/Q$, the integer division.
- (iv) Consider the quadratic discriminant $\delta = u^2 - 4 \cdot v$ and let P be the least bound such that

$$\prod_{p < P; p \text{ prime}} p > 2^m.$$

For primes $2 < p < P$, check if δ is a quadratic non residue modulo p . If this is found to be true for p , return N as the requested prime.

- (v) increase k .

2 Advantages of the procedure

This procedure provides a primality proof and a simple Plaistad certificate (consisting of the list of the intermediate primes Q together with the final prime N generated by the algorithm). This eliminates all security considerations about the necessary number of iterations, common for the probabilistic primality generation schemes. Besides, the performance of the algorithm is also superior to the one of probabilistic schemes.

The time gains stem from:

- I. Less than 2 exponentiations are required, on average, for a primality proof (one for the Fermat test and one for the Pocklington lemma).
- II. The number of Fermat tests is reduced by the use of the Eratosthenes sieve (as compared to search methods based random trials and strong pseudoprime tests).
- III. The recursion time is negligible (roughly $\frac{1}{80}$ of the total time).

In practice this leads to an improvement of 20 – 30% with respect to strong pseudoprime tests based on the same base arithmetic implementation. The algorithm is state of the art and used in different bank software. It has been provided for the public domain packages SSLEay and Crypto3.0 (Wei Dai). As an exercise, the largest random proved prime – a number with 50000 bits – was produced with the simple SSLEay implementation [PP].

3 Security Assessment and Considerations

The algorithm seeks primes in arithmetic progressions $N_k = 2 \cdot (t + k) \cdot Q + 1$, starting at random positions depending on the choice of t . Since the location of the start point is more likely to be drawn inside longer gaps between consecutive primes, the algorithm has a known bias towards primes at the end of longer gaps.

This bias may be very well analyzed using the old and highly accepted conjecture of Hardy – Littlewood concerning the distribution of n – tuples of primes. This has as consequence the fact that the lengths of gaps between primes are Poisson distributed. If $H_U(m), H_A(m)$ are the entropies of uniform distributed primes with m bits, respectively of primes with m bits generated by this algorithm, it is shown [Mi] that

$$H_U(m) - \log(H_U(m)) < H_A(m) < H_U(m),$$

thus the entropy loss for m bit primes is less than $\log(m)$. What more is, any (unknown) algorithm capable of using the distribution bias of AP for breaking some number theoretic problem (factoring, discrete logarithm) may be modified into an algorithm which breaks the same problem for uniform distributed primes in negligibly higher run time. The following fact is proved in [Mil]:

Theorem 1. *If $O(m)$ is an oracle giving the solution of one of the above problems for the biased distribution, then there is a modified oracle $O'(m)$ solving the same problem for uniform distributed primes. The response time of $O'(m)$ is superior to the one of $O(m)$ by less than $f(m)$, for a polynomial f . In practical ranges of magnitude, the value of $f(m)$ can be considered bounded by a small constant < 100 .*

Using a classical conjecture it is thus shown that the only drawback of the algorithm, the distribution bias, which pays for the useful advantages described in the previous section, has negligible and controlled impact. The security of the algorithm is thus equivalent to the one of random uniform distributed strong pseudoprime test based generation algorithms.

The certainty that the algorithm terminates by finding a prime is controlled by the parameter s . A universal proof depends upon the generalized Riemann hypothesis; for practical purposes the proposed value $s = 10$ is sufficient and it did never fail in the current implementations. Note that if no prime is found in the fixed search interval, a new search with different random values will be likely to generate the required prime.

There are no other limitations or disadvantages of this algorithm. If storage is a problem, the size of the factor tables may be reduced without changing the principle of the algorithm. The performance loss will be comparable to the one of any other known algorithm.

4 Intellectual Property Issue

The algorithm is intellectual property of P. Mihailescu and FingerPIN AG. It is state of the art, having been applied for several years, published and distributed in public domain software. No patents are thus existing or possible.

References

- [Mi] P.Mihăilescu, "Fast Generation of Provable Primes Using Search in Arithmetic Progressions", Proceedings CRYPTO94, pp. 282-293.
- [Mi1] P.Mihăilescu, "Measuring the cryptographic relevance of biased distributions in public key generation algorithms.", preprint.
- [PP] Chris Caldwell, "The Prime Page",
<http://www.utm.edu/research/primex/index.html>