PRIMALITY PROVING USING ELLIPTIC CURVES: AN UPDATE

F. MORAIN

ABSTRACT. In 1986, following the work of Schoof on point counting on elliptic curves over finite fields, new algorithms for primality proving emerged, due to Goldwasser and Kilian on the one hand, and Atkin on the other. The latter algorithm uses the theory of complex multiplication. The algorithm, now called ECPP, has been used for nearly ten years. The purpose of this paper is to give an account of the recent theoretical as well as practical improvements of ECPP, as well as new benchmarks for integers of various sizes.

1. INTRODUCTION

The last ten years have shown the power of the theory of elliptic curves in many areas of number theory and cryptography. Fast algorithms for integer factorization [12], primality proving [5, 1] and point counting over finite fields [30] were discovered and optimized (see [16, 15, 14] for a bibliography on the topic).

Even though one could dream of using Schoof's algorithm for primality proving – as Goldwasser and Kilian did [5] – the approach due to Atkin, using complex multiplication, is still computationally faster. This algorithm, popularized as ECPP [1, 23], has been used for nearly 10 years. Due to continuous work of the author, new theoretical results emerged, related mostly to the computation of character sums. Moreover, it is customary to say that a program being five year old has to be rewritten. Thus, it appeared desirable to update the whole program, to meet the current technological trends (in brief, more memory, bad processor arithmetic, poor I/O's). As a result, a new version of the package, realized by the author, is now available (see the comments at the end of this article). The resulting program is much faster than the old version, and was able to prove the primality of the 2196 decimal digit cofactor of $2^{7331} - 1$ (work realized by E. Mayer and the author). Actually, it was about time, since the Jacobi Sums test seems to wake up from a long sleep; for the new developments of this test and a new primality record beating that of ECPP, see [19] and the announcements [18, 20].

Section 2 contains a brief review of elliptic curves and ECPP. One of the major improvements concerns the factorization routines used and Section 3 is devoted to this. Section 4 contains the recent improvements to the proving part of ECPP. These results can also be used to build CM curves quickly and are of independent interest (for cryptographic applications, say). Section 5 contains the benchmarks for our implementation: we give the timings for proving the primality of integers having less than 512 bits on some platforms. We also give the time needed to check the primality certificates. As a typical result, proving the primality of a 512-bit number on an Alpha 125 MHz takes only 35 seconds.

Date: March 18, 1998.

Key words and phrases. Primality proving, elliptic curves, complex multiplication.

The author is on leave from the French Department of Defense, Délégation Générale pour l'Armement.

2. An overview of ECPP

The reader wishing to learn about elliptic curves and their cryptographic applications is referred to [17]. A good reference for elliptic curves in general is [31].

We will work on elliptic curves E of equation

$$Y^2Z = X^3 + aXZ^2 + bZ^3$$

over fields or rings $\mathbb{Z}/N\mathbb{Z}$. When N is prime, the set of points of such a curve (i.e., solutions of the above equation in the projective plane) forms an abelian group; the law is denoted by $+$ and the neutral element is the point at infinity, $O_E = (0, 1, 0)$ and equations for this are given in [31]. When N is composite, we do as if N were prime and wait for some a Z having a non trivial gcd with N.

Let us first recall the primality theorem [5]:

Theorem 2.1. *Let N be an integer prime to 6, E an elliptic curve over $\mathbb{Z}/N\mathbb{Z}$, together with a point P on E and m and s two integers with $s \mid m$. For each prime divisor q of s, we put $(m/q)P = (x_q, y_q, z_q)$. We assume that $mP = O_E$ and $\gcd(z_q, N) = 1$ for all q. Then, if p is a prime divisor of N, one has $\#E(\mathbb{F}_p) \equiv 0 \bmod s$.*

We have also:

Corollary 2.1. *With the same conditions, if $s > (\sqrt[4]{N} + 1)^2$, then N is prime.*

The following description of ECPP comes from [1]. Remember that there are basically two phases in the ECPP algorithm. In the first one, a decreasing sequence of probable primes $N_1 = N > N_2 > \ldots > N_k$ is built, the primality of N_{i+1} implying that of N_i. In brief, N_{i+1} is the largest (probable) prime factor of the order m_i of some given elliptic curve E_i. In the second phase, the curve E_i is built and the primality theorem is used to prove the primality of N_i. More formally

function ECPP(N): boolean;

1. if $N < 1000$ then check the primality of N directly and return the answer.
2. Find an imaginary quadratic field $\mathbb{K} = \mathbb{Q}(\sqrt{-D})$ $(D > 0)$, for which the equation

$$4N = x^2 + Dy^2 \qquad (1)$$

has solutions in rational integers x and y.

3. For each pair (U, V) of solutions of (1), try to factor $m = ((U-2)^2+DV^2)/4 = N+1-U$. If one of these can be written as $F \times \varpi$ where F is completely factored and ϖ a probable prime, then go to step 4 else go to step 2.
4. Find the equation of the curve E having m points modulo N and a point P on it. If the primality condition is satisfied for (N, E, P), then return ECPP(ϖ). Otherwise, return composite.
5. end.

The justification of this algorithm relies on the fact that if N is really prime, step 2 ensures that N splits as the product of principal ideals in $\mathbb{K}$, and therefore is the norm of the algebraic integer $\pi = (x + y\sqrt{-D})/2$. In that case, m is precisely the norm of $\pi - 1$ and the theory of complex multiplication asserts that E has indeed m points modulo N. It can be shown that the probability for a prime to split as a product of principal ideals in $\mathbb{K}$ is $1/(2h)$ where h is the class number of $\mathbb{K}$. Of particular interest are the nine fields for which $h = 1$, corresponding to $D \in \{3, 4, 7, 8, 11, 19, 43, 67, 163\}$.

Let us look at the work involved in this algorithm. For each discriminant D tried in step 2, we have to extract a square root modulo N and use a gcd-like computation (the so-called Cornacchia algorithm [1, p. 54] and [28]). Tricks for combining probabilistic primality proving of N and square root extractions modulo N are given in [1, p. 54] and make this phase very fast.

Step 3 requires finding small factors of m, plus a probabilistic primality test. Finding small factors of a given number is speeded up by a particular sieve, also explained in [1, p. 55]; new tricks for speeding this sieve are explained in Section 3. The running time of the whole algorithm is dominated by this phase and so the parameters should be chosen very carefully. We will come back to this task in section 5.

Finally, Step 4 requires building CM curves, and some progress has been made in that direction, see Section 4. Checking the conditions of the primality theorem also requires computing multiples of a point on an elliptic curve. Since we are working one curve at a time, the homogeneous form of the group law is preferable, and is even more profitable when using Montgomery's arithmetic [21].

3. Improving the factorization stage

The integer N being given as $4N = U^2 + DV^2$ (D a fundamental discriminant > 0), we hope that $N + 1 \pm U$ can be factored easily. For this, we can try several methods, already mentioned in [1].

3.1. Improving the sieving part.

As explained in [1], we begin our factorization stage by a sieve. We precompute $r_p = (N+1) \bmod p$ for all small $p \leq P_{\max}$. Once we have this table, we can test whether $p \mid N \pm 1$ (by testing whether $r_p = 0$ or $r_p = 2$) or whether $p \mid N+1 \pm U$, by testing whether $U \bmod p = \mp r_p \bmod p$, which is rather economical, since $|U| \leq \sqrt{p}$.

3.1.1. *Special division routines.* The basic operation we have to perform is thus the computation of $U \bmod p$ for many small p's. It is rather embarrassing to see that modern processors are performing badly as far as integer division is concerned. This is particularly true for the DecAlpha. This suggests to use special tricks for speeding up the division process. Note that we used the BigNum package for our implementation.

A first trick is to divide U by blocks of primes. Suppose we are working with a base B representation of large integers (typically $B = 2^{64}$ on a DecAlpha). We gather primes p_i's in such a way that $c = p_{i_1} p_{i_2} \cdots p_{i_k} < B$ and perform one long division of U by c, followed by division operations by the p_{i_j}'s.

For the DecAlpha only, a second trick is to use a special division routine inspired by [6, Figure 4.1] for computing the quotient of U by c, with $c < 2^{32}$. We made numerous experiments for integers with less than 512 bits, comparing a special routine written in BigNum for division by small integers * with the algorithm of Granlund and Montgomery. We obtained a speedup of at least 2 resulting in a 10% savings in the whole first stage of ECPP. Note that this trick can be (actually *is*) combined with the preceding one.

3.1.2. *A trick for the case $D = 3$.* In that case, we have 6 solutions to the equation $4N = x^2 + Dy^2$. The first one being (U, V) with $U > 0$, say, the others are:

$$(-U, -V), (\pm(U + 3V)/2, \mp(U - V)/2), (\pm(U - 3V)/2, \pm(U - V)/2).$$

*We divide a 64 bit word integer by a 32 bit one using base 2^{32} arithmetic.

Having computed $u_p = U \bmod p$ and $w_p = 3V \bmod p$, we can check whether p divides any of the numbers $N+1-W$ for $W \in \{\pm U, \pm(U \pm 3V)/2\}$ using linear combinations of r_p, u_p and w_p, thus saving one third of the division operations.

3.2. **Modifying the ρ method for $D = 3$ and $D = 4$.** Traditionally, one uses Pollard's ρ method with a degree 2 function. When one knows that the prime factors p of an integer m are congruent to 1 modulo a number k, it is recommended [2] to use a degree k polynomial in Pollard's ρ method. The number of iterations of the method being reduced by a factor $\sqrt{k-1}$.

There are two cases in ECPP where we know such a thing. When $D = 3$ (resp. $D = 4$), we know that each prime factor of our m's are congruent to 1 modulo D. In the case $D = 3$ (resp. $D = 4$), one can use $f_3(x) = x^3 + 1$ (resp. $f_4(x) = x^4 + 1$). In tables 1 and 2, we indicate the number of modular multiplications and modular squarings needed to find factors $p \leq 10^8$. We used Montgomery's MCF routine [22] for f_4, f_3 † and for $f_2(x) = x^2 + 3$ and $x_0 = 1$. We list those primes that are champions, namely those for which the number of iterations is larger than all the preceding primes.

	2	3	$\times$	$\square$
$\leq 10^6$		830017	8535	4242
	874771		5240	5188
$\leq 10^7$	7784389		19418	19357
		9992053	26851	13397
$\leq 5 \cdot 10^7$		48749479	66313	33125
	48909031		46665	46598
$\leq 10^8$		93490387	100980	50457
	95507539		67403	67336

TABLE 1. Comparisons of the variants of ρ for $D = 3$.

For instance, finding all primes p congruent to 1 modulo 3 that are $\leq 10^7$ requires 26851 modular multiplications and 13397 squarings using $f_3(x)$, compared to respectively 19418 and 19357. If a modular multiplication requires M operations, a squaring S, then the gain τ_3 for $D = 3$ (resp. τ_4 for $D = 4$) for $p \leq 10^8$ is

$$\tau_3 = \frac{100980M + 50457S}{67403M + 67336S}, \tau_4 = \frac{61148M + 61081S}{35418M + 70707S}.$$

Note that from a practical point of view, such optimizations are rather difficult to appreciate precisely, but they are nice from a theoretical point of view.

4. BUILDING CM CURVES OF PRESCRIBED CARDINALITY

In the proving part, one has a (probable) prime p and a putative number of points m of a curve E having complex multiplication by the ring of integers of $\mathbb{Q}(\sqrt{-D})$. Once we have E, we find a point P on it and we have to check the primality conditions.

To find E, one has to find a root of the so-called Weber polynomial, which gives the invariant of the curve. From this, we have to compute the coefficients of E. There are up to 6 (classes of) curves having the same invariant and we have to find the one having m

†Though it cannot be applied *a priori* with $f_3(x)$, it is trivial to modify MCF in this particular case.

	2	4	$\times$	$\square$
$\leq 10^6$		968729	3428	6757
	994393		5173	5121
$\leq 10^7$	7784389		19418	19357
		8854621	11667	23217
$\leq 5 \cdot 10^7$		48659461	24217	48311
	49684241		48180	48113
$\leq 10^8$		92188529	35418	70707

TABLE 2. Comparisons of the variants of ρ for $D = 4$.

points. We can certainly try all of them before finding the right one. However, any gain is worthwhile.

The most heavily used primality tests are the $N \pm 1$ test and the tests corresponding to $D = 3$ or 4, as indicated in Table 3 where we give the statistics for numbers of b bits. For these values, it is natural to speed up the construction of E.

$D \backslash b$	128	192	256	320	384	448	512
-1	0.59	0.50	0.41	0.33	0.33	0.30	0.26
1	0.16	0.12	0.12	0.14	0.13	0.10	0.09
3	0.12	0.16	0.23	0.20	0.19	0.19	0.20
4	0.05	0.08	0.05	0.09	0.08	0.08	0.08

TABLE 3. Frequencies of discriminants used in ECPP.

4.1. **The cases $D = 3$ and $D = 4$.** For these two cases, algorithms are given in [1, p. 58]. Slightly more efficient approaches can be found in the literature [9]. We combine these theorems with the use of known values of quartic and cubic symbols computed in [33].

The methods to be described yield a 40% savings in the proving part for small numbers (less than 512 bits). We begin with $D = 4$ to describe the philosophy, giving less details for $D = 3$. In the sequel, we let (a/p) denote the Legendre symbol.

4.1.1. *The case $D = 4$.*

Theorem 4.1. *Let $p \equiv 1 \bmod 4$ and write $p = x^2 + 4y^2$ with $x \equiv 1 \bmod 4$. The quartic symbol is $(k/p)_4 \equiv k^{(p-1)/4} \bmod p$. If $a \not\equiv 0 \bmod p$, then the curve $Y^2 = X^3 + aX$ has cardinality $p + 1 - t$ where*

$$t = \begin{cases} 2x & \text{if } (a/p)_4 = 1, \\ -2x & \text{if } (a/p)_4 = -1, \\ -4y & \text{otherwise where } y \text{ is chosen uniquely by } 2y(a/p)_4 = x. \end{cases}$$

We will use the following result of [33][‡]

[‡]Lienen's notations are not ours at this point.

Theorem 4.2. *Write $p = s^2 + 4y^2$ with $s = 2y + 1 \bmod 4$ and let $i = 2y/s \bmod p$ so that $i^2 \equiv -1 \bmod p$. Then*

$$(2/p)_4 = \begin{cases} 1 & \text{if } y \equiv 0 \bmod 4, \\ i & \text{if } y \equiv 3 \bmod 4, \\ -1 & \text{if } y \equiv 2 \bmod 4, \\ -i & \text{if } y \equiv 1 \bmod 4. \end{cases}$$

When $y \equiv 0 \bmod 2$, we have

$$(3/p)_4 = \begin{cases} 1 & \text{if } 3 \mid y, \\ i & \text{if } 3 \mid x + 2y, \\ -1 & \text{if } 3 \mid x, \\ -i & \text{if } 12 \mid x - 2y. \end{cases}$$

Suppose that $p = u^2 + v^2$ and we want a curve of cardinality $p + 1 - 2u$, the first thing we have to do is recover x and y from u and v. Then we proceed as follows, finding a satisfying any of the three cases of Katre's theorem[§]:

1. If $2u = 2x$, it is enough to find an a such that $(a/p)_4 = 1$ and we can take $a = 1$.
2. If $2u = -2x$, then any a with $(a/p)_4 = -1$ will do: If $p \equiv 5 \bmod 8$, we take $a = -1$. If $p \equiv 1 \bmod 8$, according to Theorem 4.2, we take $a = 4$ when $y \equiv 1 \bmod 2$ and $a = 2$ when $y \equiv 2 \bmod 4$; when $y \equiv 0 \bmod 4$ we can take $a = 3$ when $3 \mid x$, and 9 if $3 \mid x \pm 2y$; otherwise, we do an exhaustive search, starting at $a = 5$.
3. When $2u \not\equiv \pm 2x$, we choose the sign of y such that $2u = -4y$. Then we must find a such that $(a/p)_4 = x/(2y) \in \{\pm i\}$ (we cannot have $x = \pm 2y$). If y is odd, then with the notations of Theorem 4.2, one has $s = -x$; if $y \equiv 3 \bmod 4$, we take $a = 2$ and if $y \equiv 1 \bmod 4$, we take $a = 1/2$. When y is even, then $s = x$; if $y \equiv 2 \bmod 4$, we let w be a square root of $2 \bmod p$ and take for a the value w or $1/w$ such that $(a/p)_4 = x/(2y)$; if $y \equiv 0 \bmod 4$, we take $a = 1/3$ when $3 \mid x + 2y$, $a = 3$ when $3 \mid x - 2y$, w or $1/w$ with $w^2 \equiv 3 \bmod p$ when $3 \mid x$ and we use exhaustive search beginning at $a = 5$ in the last case where $6 \mid y$.

4.2. **The case $D = 3$.** First of all, we have Katre's result:

Theorem 4.3. *Let $p \equiv 1 \bmod 3$ and write $4p = L^2 + 27M'^2$, where $L \equiv 1 \bmod 3$. The cubic symbol is denoted $(k/p)_3 \equiv k^{(p-1)/3} \bmod p$. If $b \not\equiv 0 \bmod p$, then the curve $Y^2 = X^3 + b$ has cardinality $p + 1 - t$ where*

$$t = \begin{cases} -(b/p)L & \text{if } (4b/p)_3 = 1, \\ \frac{1}{2}(b/p)(L + 9M') & \text{otherwise where } M' \text{ is chosen uniquely by} \\ & \qquad (L - 9M')(4b/p)_3 = (L + 9M'). \end{cases}$$

Let us normalize things as follows: p is a prime number $\equiv 1 \bmod 3$, so that $4p = U^2 + 3V^2 = L^2 + 27M^2$, with $L \equiv 1 \bmod 3$ and $M > 0$. We want a curve $E : Y^2 = X^3 + b$ having $p + 1 - U$ points over $\mathbb{F}_p$. Following [33], we can also write $p = \alpha^2 - \alpha\beta + \beta^2$ with $\beta = 3M$ and $\alpha = (L + 3M)/2$ for which $\alpha \equiv 2 \bmod 3$.

Proposition 4.1. *Let $X^3 - 1 = (X - 1)(X - v_p)(X - w_p) \bmod p$. Then $v_p \equiv (L + 9M)/(L - 9M) \bmod p$, $w_p \equiv 1/v_p \bmod p$.*

[§]Though tedious to implement, this procedure is very fast, resorting to the Riemann hypothesis in as few cases as possible.

With all these notations, we have:

Theorem 4.4.

$$\left(\frac{2}{p}\right)_3 = \begin{cases} 1 & \text{if } \beta \equiv 0 \bmod 6, \\ v_p & \text{if } \beta \equiv 3 \bmod 6, \alpha \equiv 2 \bmod 6, \\ w_p & \text{if } \beta \equiv 3 \bmod 6, \alpha \equiv 5 \bmod 6, \end{cases}$$

$$\left(\frac{3}{p}\right)_3 = \begin{cases} 1 & \text{if } \beta \equiv 0 \bmod 9, \\ v_p & \text{if } \beta \equiv 6 \bmod 9, \\ w_p & \text{if } \beta \equiv 3 \bmod 9 \end{cases}$$

The algorithm used to find b such that $E : Y^2 = X^3 + b$ has $p+1-U$ points is as follows: We first search for the cubic residue and then correct the value of the Legendre symbol, remarking that if γ is a quadratic non residue, then the value of $(b/p)_3$ is unchanged when b is replaced by $b\gamma^3$. The algorithm is:

1. Find (L, M) from (U, V).

2. If $|U| = |L|$, we let $\varepsilon = U/L$ and we must find b such that $(4b/p)_3 = 1$ and $(b/p) = -\varepsilon$. In the case $\varepsilon = -1$, $b = 2^4$ is convenient; when $\varepsilon = +1$, we take γ such that $(\gamma/p) = -1$ and take $b = \gamma^3/4$.

3. If $U = \pm(L+9\theta M)/2$ with $\theta \in \{\pm 1\}$, we let $\varepsilon = (2U)/(L+9\theta M)$ and we must find b such that $(4b/p)_3 = (L+9\theta M)/(L-9\theta M)$ and $(b/p) = \varepsilon$. For this, we first look for b' such that $(b'/p)_3 = (L+9\theta M)/(L-9\theta M)$ and then set $b = b'/4$. If $M \equiv 0 \bmod 6$, we look for b' by enumeration, starting from 5, since 2 and 3 are cubic residues by Theorem 4.4. Otherwise, we take for b' a suitable power of 2 or 3, using Theorem 4.4.

4.3. **Other cases.** All cases for $h = 1$ are now fully proved (see [8, 29] and [32]).

Stark's results are also of interest for building curves with prescribed properties, as used in [24, 11, 3]. We give here the main theorem of Stark. Remember the definition of Weber's functions in terms of the modular invariant $j(z)$:

$$\gamma_2(z) = \sqrt[3]{j(z)}, \quad \gamma_3(z) = \sqrt{j(z) - 1728}.$$

Stark's theorem is:

Theorem 4.5. *Suppose $(D, 6) = 1$ and $4p = U^2 + DV^2$. Put $g_2 = \gamma_2((-3+\sqrt{-D})/2)$ and $g_3 = \gamma_3((-3+\sqrt{-D})/2)$. Let c be an element of $\mathbb{Z}/p\mathbb{Z}$. Then the curve $E_c : y^2 = x^3 + c^2 D\frac{g_2}{48} - c^3 D\frac{g_3\sqrt{-D}}{864}$ has cardinality*

$$p + 1 - \begin{cases} \left(\frac{c}{p}\right)\left(\frac{2u}{D}\right) u & \text{if } D \equiv 7 \bmod 8, \\ \left(\frac{-c}{p}\right)\left(\frac{2u}{D}\right) u & \text{if } D \equiv 3 \bmod 8. \end{cases}$$

Some cases are also known for $h = 2$, which are not comprised in Stark's theorem. For $D = 15$, we can use Rajwade's method as improved in [8] (see [4] for the details):

Theorem 4.6. *Let $p = u^2 + 15v^2$ with $u \equiv 2 \bmod 3$ if $p \equiv 1 \bmod 4$ and $u \equiv 1 \bmod 3$ if $p \equiv 4 \bmod 15$. Let ϖ be any square root of 5 modulo p. Then the curve of equation*

$$y^2 = x^3 - (3675 + 6240\varpi)x - (188650 + 320320\varpi)$$

has $p + 1 + 2u$ points.

For $D = 20$, one can use the following result taken from [13] (see this reference for complementary remarks on the normalizations):

Theorem 4.7. *Suppose that* $p = a^2 + b^2 = u^2 + 5v^2$ *with* $a \equiv 1 \bmod 4$. *We put* $\varepsilon_{20}(p) = +1$ *if* $p \equiv 1 \bmod 20$ *and* -1 *if* $p \equiv 9 \bmod 20$ *(these are the only possible congruences for* $p \bmod 20$*). Let* $\varpi = \sqrt{5} \bmod p$ *and* $Q = 1/\varpi \bmod p$. *Put* $\varepsilon(Q) = (Q/p)_4$ *if* $(Q/p) = 1$ *and* 1 *otherwise. Then the curve of equation*

$$y^2 = x(x^2 + 4x + 2 - \varpi)$$

has $p + 1 - \varepsilon_{20}(p)\varepsilon(Q)\ (2u)$ *points when* u *is normalized as follows: If* $(Q/p) = 1$, *then* $u \equiv a \bmod 5$; *if* $(Q/p) = -1$, *one should take* $b \equiv aQ^{(p-1)/4} \bmod p$ *and* $u \equiv b \bmod 5$.

5. Benchmarks of the program

5.1. **A few words on strategy.** As in [1, p. 52], we use a precomputed list of 2244 discriminants that can be used. This enables us to factor up to 4500 putative m's (corresponding to $D = \pm 1$ and all D such that $h \leq 20$ and for $(h, g) \in \{(32, 16), (24, 8), (48, 16), (32, 8), (64, 16)\}$).

ECPP can be seen as a tree search. At each level, we have to run through a set of possible branches, one per discriminant. In most cases, we have enough discriminants to find a good one and we can go down to next level. In some cases (when we are dealing with a bad number), our stock is not enough and we have the possibility of backtracking one level up. This *backtrack* feature is present in our implementation of ECPP. Another special feature is the *redo* option, which says that before backtracking, we should try our list of discriminants at the same level with higher factorization parameters.

5.2. **Benchmarks.** All benchmarks were done on a `DEC 3000 - M300LX` (processor alpha). For comparisons with [1, p. 61], we managed to find an old DEC 5000 (processor mips).

5.2.1. *Basic arithmetic.* We first give the timings for modular multiplication for our implementation, built on top of the `BigNum` package. We assume our numbers have size between 128 and 512 bit. We give the timings for the naive implementation, as well as the one using Montgomery's arithmetic [21].

b	128	192	256	320	384	448	512
plain	58	81	101	119	144	165	196
Montgomery	28	37	44	54	76	86	102

Table 4. 64-bit arithmetic on DEC 3000 - M300LX in μs

b	128	192	256	320	384	448	512
plain	47	78	176	203	254	297	336
Montgomery	27	78	125	113	137	195	285

Table 5. 32-bit arithmetic on DEC 5000, in μs

The DEC 3000 machine is a 64-bit, the DEC 5000 is 32-bit. On a DEC 3000, Montgomery's arithmetic is up to twice as fast as the plain version for 512 bits. Montgomery's arithmetic is very nice, but some care has to be taken when using it, because all constants are to be normalized before use. Since this can be very cumbersome, we limit the use of this arithmetic to critical parts of the algorithm, such as probabilistic primality proving.

5.2.2. *ECPP.* The speed of ECPP depends largely on the factorization parameters used. As in [1], a sieve is used, followed by Pollard's ρ if necessary, then $p-1$ is used and finally ECM as a last resort. For numbers up to 512 bits, the sieve is enough. Finding the optimal values involves running the program with a lot of different parameters, starting from $b = 128$ bits, up to 512. The largest prime $P_{\max}$ for the sieve in our implementation is given in the following table:

b	128	192	256	320	384	448	512
$P_{\max}$	10000	30000	30000	50000	50000	50000	80000

Tables 6 and 7 contain the timings for 50 random primes of each size. The first line gives the time for the building phase, the second for the proving phase, the third the total time and the fourth the number of intermediate primes. Optimizing the parameters is a tedious task. We tried to minimize the maximal time spent.

b	min	max	mean	s.d.	b	min	max	mean	s.d.
128	0.34	1.82	0.81	0.32	384	12.39	69.99	28.30	11.47
	0.18	1.36	0.61	0.25		9.26	26.14	16.42	3.66
	0.70	3.01	1.43	0.52		23.54	92.10	44.72	14.26
	4.00	15.00	9.82	1.96		18.00	35.00	25.26	3.39
192	0.98	6.50	2.94	1.15	448	22.56	100.43	46.85	15.70
	0.37	3.03	1.80	0.55		17.33	43.35	27.84	6.56
	1.35	9.52	4.74	1.57		41.43	141.48	74.69	20.84
	9.00	21.00	13.76	2.75		20.00	36.00	29.08	3.16
256	3.15	20.62	7.11	3.09	512	40.23	143.15	79.39	23.32
	2.39	10.51	4.56	1.41		29.35	68.90	43.05	8.33
	5.97	31.13	11.67	4.31		69.58	194.66	122.44	29.05
	12.00	24.00	17.64	2.79		28.00	40.00	32.52	3.01
320	6.54	24.00	13.74	4.19					
	5.91	13.49	8.92	1.81					
	12.90	34.20	22.66	5.34					
	16.00	25.00	20.80	2.16					

TABLE 6. Benchmarks for 50 primes on a DEC 5000

5.2.3. *Checking the ECPP certificates.* We measured the time needed to check a certificate, and this only on the DEC 3000 machine. Results are given in Table 8.

6. PROGRAMS, RECORDS

The first version of ECPP (version 3.4.1) was made available some time around 1991. Several thousands copies were taken since. Continuous (if sporadic) work on the different libraries progressively modified the program and the program continuously used for proving

b	min	max	mean	s.d.	b	min	max	mean	s.d.
128	0.08	0.82	0.26	0.15	384	3.68	27.10	9.17	3.87
	0.05	0.48	0.17	0.09		1.87	8.57	3.56	1.21
	0.15	1.02	0.43	0.21		5.55	32.97	12.73	4.78
	7.00	14.00	9.82	1.88		18.00	33.00	24.08	3.22
192	0.22	2.47	1.06	0.51	448	5.12	31.38	14.67	5.43
	0.12	0.77	0.42	0.13		2.82	14.28	5.93	2.18
	0.33	3.12	1.48	0.59		8.23	45.67	20.60	7.10
	9.00	18.00	13.14	2.27		23.00	34.00	28.42	2.88
256	1.17	6.38	2.95	1.42	512	12.40	45.02	25.63	8.43
	0.38	2.10	0.89	0.31		5.75	27.00	10.18	4.16
	1.70	7.20	3.84	1.61		18.17	68.53	35.81	11.31
	9.00	23.00	16.54	2.75		24.00	39.00	31.08	3.56
320	1.50	8.80	4.74	1.67					
	0.98	3.40	1.84	0.56					
	2.87	10.45	6.58	1.96					
	13.00	28.00	19.92	2.91					

TABLE 7. Benchmarks for 50 primes on a DEC 3000.

b	min	max	mean	s.d.
128	0.16	0.95	0.42	0.15
192	0.29	2.34	1.38	0.43
256	1.48	5.63	3.14	0.78
320	4.25	8.80	6.08	1.03
384	5.11	14.09	10.20	1.98
448	9.44	22.11	16.29	2.92
512	16.78	34.00	24.89	3.91

TABLE 8. Time for checking the certificates on a DEC 3000

the primality of large numbers [10, 25, 26, 27]. An intermediate version was donated to the MAGMA group and incorporated in version 2.2.

The new public version (5.6.1) has been available for a year, without any publicity made anywhere (at least until november 1997). It has been copied by several hundred people. One of these users (E. Mayer) sent me enough mails to wake me up and force me to improve my program. This resulted in a small step for theory, but a giant step for users: a restart option was added to the program, making it very useful for long runs. With this option, E. Mayer and I were able to prove the primality of the number $(2^{7331}-1)/458072843161$ in one month of DecAlpha 400 MHz, thus setting the new ECPP record at 2196 decimal digits, which is 500 digits more than the preceding one¶ which had 1505 decimal digits (see [25]).

¶Actually, E. Mayer broke that record some time before without even realizing it!

7. Conclusions

We have described the recent developments in effective complex multiplication, yielding fast and direct ways of building CM curves. This together with many algorithmic improvements made it possible to release a new and faster version of ECPP.

Note that the only concurrent of ECPP, the Jacobi sums test, has been recently improved by Mihailescu [19, 20] who announced a new primality record [18], with $N = 2^{10000} + 177$, whose primality was established in 138 hours on an Alpha 500. This might tend to prove that this cyclotomic algorithm is much faster than ECPP. However, the fact that ECPP gives a certificate of primality that can be checked independently, with a small program, continues to be an advantage. As a final remark, Mihailescu's thesis raises interesting connections between cyclotomic ideas and elliptic curves. These connections will be investigated in a near future.

Final remarks: The ECPP program is written in C and uses the `BigNum` package [7]. A small package, containing some binaries and data files, is available on `http://www.lix.polytechnique.fr/~morain/`.

Acknowledgments. The author wants to express his gratitude to E. Mayer for his interest in ECPP and for developing much enthusiasm for intermediate versions. No doubt that without him, the record would still be at 1500 digits. Also, thanks to P. Mihailescu for his stimulating competition and many interesting questions related to his work. Finally, G. Hanrot's reading of this manuscript was heartily welcomed.

References

[1] A. O. L. Atkin and F. Morain. Elliptic curves and primality proving. *Math. Comp.*, 61(203):29–68, July 1993.

[2] R. P. Brent and J. M. Pollard. Factorization of the eighth Fermat number. *Math. Comp.*, 36(154):627–630, April 1981.

[3] J. Chao, K. Harada, K. Tanada, and S. Tsujii. Design of cryptographically secure elliptic curves over extension fields with CM field method. Preprint, 1995.

[4] J.-M. Couveignes, A. Joux, and F. Morain. Sur quelques sommes de caractères. In preparation, February 1994.

[5] S. Goldwasser and J. Kilian. Almost all primes can be quickly certified. In *Proc. 18th STOC*, pages 316–329. ACM, 1986. May 28–30, Berkeley.

[6] T. Granlund and P. L. Montgomery. Division by invariant integers using multiplication. *SIGPLAN Notices*, 29(6):61–72, 1994.

[7] J.-C. Hervé, F. Morain, D. Salesin, B. Serpette, J. Vuillemin, and P. Zimmermann. Bignum: A portable and efficient package for arbitrary precision arithmetic. Rapport de Recherche 1016, INRIA, April 1989.

[8] A. Joux and F. Morain. Sur les sommes de caractères liées aux courbes elliptiques à multiplication complexe. *J. Number Theory*, 55(1):108–128, November 1995.

[9] S. A. Katre. Jacobsthal sums in terms of quadratic partitions of a prime. In K. Alladi, editor, *Number Theory*, volume 1122 of *Lecture Notes in Math.*, pages 153–162. Springer-Verlag, 1985. Proceedings of the 4th Matscience Conference held at Ootacamund, India, January 5-10, 1984.

[10] W. Keller and F. Morain. The complete factorization of some large Mersenne composites. *Abstracts of the AMS*, 13(5):506, October 1992. 92T-11-163.

[11] G.-J. Lay and H. G. Zimmer. Constructing elliptic curves with given group order over large finite fields. In L. Adleman and M.-D. Huang, editors, *ANTS-I*, volume 877 of *Lecture Notes in Comput. Sci.*, pages 250–263. Springer-Verlag, 1994. 1st Algorithmic Number Theory Symposium - Cornell University, May 6-9, 1994.

[12] H. W. Lenstra, Jr. Factoring integers with elliptic curves. *Ann. of Math. (2)*, 126:649–673, 1987.

[13] F. Leprévost and F. Morain. Revêtements de courbes elliptiques à multiplication complexe par des courbes hyperelliptiques et sommes de caractères. Technical Report MPI/95–124, Max-Planck-Institut für Mathematik Bonn, 1995.

[14] R. Lercier. *Algorithmique des courbes elliptiques dans les corps finis.* Thèse, École polytechnique, June 1997.

[15] R. Lercier. Finding good random elliptic curves for cryptosystems defined over F_{2^n}. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Comput. Sci.*, pages 379–392. Springer-Verlag, 1997. International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 1997, Proceedings.

[16] R. Lercier and F. Morain. Counting the number of points on elliptic curves over finite fields: strategies and performances. In L. C. Guillou and J.-J. Quisquater, editors, *Advances in Cryptology – EUROCRYPT '95*, volume 921 of *Lecture Notes in Comput. Sci.*, pages 79–94, 1995. International Conference on the Theory and Application of Cryptographic Techniques, Saint-Malo, France, May 1995, Proceedings.

[17] Alfred J. Menezes. *Elliptic curve public key cryptosystems.* Kluwer Academic Publishers, 1993.

[18] P. Mihailescu. Advances in cyclotomy primality proving. Email to the `NMBRTHRY` mailing list; available on `http://listserv.nodak.edu/archives/nmbrthry.html`, November 1997.

[19] P. Mihailescu. *Cyclotomy of rings and primality testing.* Diss. eth no. 12278, Swiss Federal Institute of Technology Zürich, 1997.

[20] P. Mihailescu. Cyclotomy news. Email to the `NMBRTHRY` mailing list; available on `http://listserv.nodak.edu/archives/nmbrthry.html`, January 1998.

[21] P. L. Montgomery. Modular multiplication without trial division. *Math. Comp.*, 44(170):519–521, April 1985.

[22] P. L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Math. Comp.*, 48(177):243–264, January 1987.

[23] F. Morain. *Courbes elliptiques et tests de primalité.* Thèse, Université Claude Bernard–Lyon I, September 1990.

[24] F. Morain. Building cyclic elliptic curves modulo large primes. In D. Davies, editor, *Advances in Cryptology – EUROCRYPT '91*, volume 547 of *Lecture Notes in Comput. Sci.*, pages 328–336. Springer–Verlag, 1991. Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques, Brighton, United Kingdom, April 8–11, 1991.

[25] F. Morain. Prime values of partition numbers and the primality of $p(1840926)$. Rapport de Recherche LIX/92/RR/11, Laboratoire d'Informatique de l'Ecole Polytechnique (LIX), 1992.

[26] F. Morain. `(2^10501+1)/3` is prime. Email to the `NMBRTHRY` mailing list; available on `http://listserv.nodak.edu/archives/nmbrthry.html`, April 1996.

[27] F. Morain. `(2^12391+1)/3` is prime. Email to the `NMBRTHRY` mailing list; available on `http://listserv.nodak.edu/archives/nmbrthry.html`, April 1996.

[28] A. Nitaj. L'algorithme de Cornacchia. *Exposition. Math.*, 13:358–365, 1995.

[29] R. Padma and S. Venkataraman. Elliptic curves with complex multiplication and a character sum. *J. Number Theory*, 61:274–282, 1996.

[30] R. Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comp.*, 44:483–494, 1985.

[31] J. H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics.* Springer, 1986.

[32] H. M. Stark. Counting points on CM elliptic curves. *Rocky Mountain J. Math.*, 26(3):1115–1138, 1996.

[33] H. von Lienen. Reelle kubische und biquadratische Legendre-Symbole. *J. Reine Angew. Math.*, 305:140–154, 1979.

Laboratoire d'Informatique de l'École polytechnique (LIX), F-91128 Palaiseau Cedex, France

E-mail address, F. Morain: `morain@lix.polytechnique.fr`