Counting the number of points on elliptic curves over finite fields: strategies and performances

(To appear in Proc. EUROCRYPT '95)

Reynald Lercier*
François Morain†‡§

March 9, 1995

Abstract

Cryptographic schemes using elliptic curves over finite fields require the computation of the cardinality of the curves. Dramatic progress have been achieved recently in that field by various authors. The aim of this article is to highlight part of these improvements and to describe an efficient implementation of them in the particular case of the fields $GF(2^n)$, for $n \leq 600$.

1 Introduction

Elliptic curves have been used successfully to factor integers [26, 36], and prove the primality of large integers [6, 15, 4]. Moreover they turned out to be an interesting alternative to the use of $\mathbf{Z}/N\mathbf{Z}$ in cryptographical schemes [33, 21]. Elliptic curve cryptosystems over finite fields have been built, see [5, 30]; some have been proposed in $\mathbf{Z}/N\mathbf{Z}$, N composite [23, 12, 42]. More applications were studied in [19, 22]. The interested reader should also consult [31].

In order to perform key exchange algorithms using an elliptic curve E over a finite field K, the cardinality of E must be known. The first suggestions in that direction were to use supersingular curves for which the cardinality is easy to compute [33, 21, 18, 5, 30]. But these curves turned out to be disastrous, since the discrete logarithm problem can be reduced to the discrete logarithm problem over an extension field of K of small degree [29]. For non supersingular curves, no reduction algorithm is known in general and the only known attack on such schemes is to use a variant of Pollard's algorithm [16] and this algorithm has exponential running time. Hence, it appears promising to use these curves since we can achieve the same level of confidence one has with $\mathbf{Z}/N\mathbf{Z}$ with much shorter keys.

Two types of finite fields $GF(q)$ have been suggested. The first one considers curves over $GF(p)$ where p is a large prime, the second one curves defined over $GF(2^n)$ where n is some integer. It is

*CELAR/SSIG, Route de Laillé, F-35170 Bruz, **lercier@polytechnique.fr**

†LIX, École Polytechnique, F-91128 Palaiseau CEDEX, FRANCE, **morain@polytechnique.fr**

‡On leave from the French Department of Defense, Délégation Générale pour l'Armement.

§Part of this study was done under contract n°0044193 with DGA/CELAR.

possible to use the properties of complex multiplication as stated in [4] to build an elliptic curve with cardinality satisfying some properties [38, 22, 34, 35, 24, 8]. On the other hand, one can use random curves and try to compute its cardinality. It was not until recently that Schoof's polynomial time algorithm for solving this problem could be efficiently implemented and give satisfactory results. The aim of this paper is to give some hints on how this was made possible and to give some precise timings on randomly selected curves.

Since there are industrial applications for elliptic curves over $GF(2^n)$ [16, 31], we will focus on this case. We will briefly compare the running time of our implementation with that of the case $GF(p)$, p a large prime.

The structure of this paper is as follows. Section 2 recalls basic facts on elliptic curves. Section 3 describes Schoof's algorithm in a synthetic way using the contributions of Atkin, Elkies, Couveignes–Morain and the decisive ideas of Couveignes for the computation of isogenies in characteristic 2. We will present some strategies combining these ideas. Some details of the implementation are given in Section 4; precise timings on random curves for various fields are also given.

Throughout the paper, we let $K = GF(q) = GF(p^n)$ be a finite field of characteristic p.

2 Elliptic curves over finite fields

We recall well known properties of elliptic curves. All these can be found in [46] (see also [31]).

The general equation of an elliptic curve E is given as:

$$\mathcal{F}(X,Y,Z) := Y^2Z + a_1XYZ + a_3YZ^2 - (X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3) = 0$$

where the a_i's are in K and the discriminant Δ defined by

$$d_2 = a_1^2 + 4a_2, d_4 = 2a_4 + a_1a_3, d_6 = a_3^2 + 4a_6, d_8 = a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2,$$

$$c_4 = d_2^2 - 24d_4, \Delta = -d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6$$

is invertible in K. The j-invariant of the curve is $j(E) = c_4^3/\Delta$.

It is possible to define on the set of points $E(K)$ of E

$$E(K) = \{(x,y) \in K^2, \mathcal{F}(x,y,1) = 0\} \cup \{O_E\}$$

an Abelian law using the so-called *tangent-and-chord* method, O_E being the neutral element $(0,1,0)$. We refer to the references given above for the precise equations of the law.

Let m denote the cardinality of the set $E(K)$ of points on E. Then, it is well known that $m = q + 1 - t$ where t is an integer satisfying $|t| \leq 2\sqrt{q}$.

3 Counting the number of points

3.1 Torsion points

Let E be an elliptic curve and let N be an integer. Define $E[N]$ as the set of points of $E(\overline{K})$ of order N. When N is prime to p, then $E[N]$ is isomorphic to $(\mathbf{Z}/N\mathbf{Z}) \times (\mathbf{Z}/N\mathbf{Z})$ and when $N = p^e$, it is either $\{O_E\}$ or $(\mathbf{Z}/p^e\mathbf{Z})$.

It can be shown that there exists a polynomial $f_N(X)$ in $\mathbf{Q}[a_1, a_2, a_3, a_4, a_6][X]$ of degree

$$d_N = \begin{cases} (N^2-1)/2 & \text{if } (N,p)=1, N \text{ odd}, \\ (N^2-4)/2 & \text{if } (N,p)=1, N \text{ even}, \\ (p^{2e}-p^e)/2 & \text{if } N = p^e, \end{cases}$$

such that $P = (X, Y, 1)$ is in $E[N]$ if and only if $f_N(X) = 0$ in $\overline{K}$. The polynomial f_N is called *division polynomial.*

3.2 Schoof's algorithm

Schoof's algorithm [43] uses the properties of the Frobenius π_E which maps $E(\overline{K})$ onto itself and which sends a point $(X, Y, 1)$ to $(X^q, Y^q, 1)$. It is known that this endomorphism has characteristic equation

$$\pi^2 - t\pi + q = 0 \tag{1}$$

where t is related to the cardinality m of $E(K)$ via $m = q + 1 - t$.

Let ℓ be a prime number. Equation (1) is still valid when π_E is restricted to the group $E[\ell]$, and equivalently

$$\pi^2 - t\pi + q \equiv 0 \bmod \ell. \tag{2}$$

We can find $t_\ell \equiv t \bmod \ell$ by finding which value of τ, $0 \leq \tau < \ell$, satisfies

$$(X^{q^2}, Y^{q^2}) + q(X, Y) = \tau(X^q, Y^q)$$

in $GF(q)[X,Y]/(\mathcal{F}(X, Y, 1), f_\ell(X))$. If we know $t \bmod \ell$ for enough ℓ such that

$$\prod \ell > 4\sqrt{q}$$

then we can determine t using the Chinese remaindering theorem.

3.3 An overview of the improvements of Atkin and Elkies

Though Schoof's algorithm has polynomial running time, its implementation was rather inefficient, due to the size of the polynomials involved. However, Atkin first and then Elkies devised theoretical and practical improvements. We suppose from now on that we want to compute $t_\ell \equiv t \bmod \ell$, ℓ a prime number different from p (see below for the particular case $\ell = p$).

Firstly, Atkin [2] explained how to use the properties of the modular polynomial $\Phi_\ell(X, Y)$ modulo p to get a list of possible values of t_ℓ. The polynomial $\Phi_\ell(X, Y)$ is symmetric in X and Y and has degree $\ell + 1$. The polynomial $\Phi(X) = \Phi_\ell(X, j(E))$ describes the cyclic subgroups of $E[\ell]$. It can have basically two splitting in K: $(11r \ldots r)$ with $\ell - 1 = rs$ or $(r \ldots r)$ with $\ell + 1 = rs$ (there are two particular cases described in the paper which are rare and we omit the relevant details for the sake of simplicity). In the first case, ℓ is said to be an *Elkies prime* and an *Atkin prime* in the second. In each case r is the order of α/β where α and β are the roots of

$$\pi^2 - t\pi + q \equiv 0 \bmod \ell$$

and lie in $GF(\ell)$ if ℓ is an Elkies prime (and thus $t^2 - 4q$ must be a square modulo ℓ) and in $GF(\ell^2)$ otherwise (implying that $t^2 - 4q$ is not a square modulo ℓ). Once r is known, there are $\varphi(r)$ possible values of t_ℓ and in many cases, this value is much less than ℓ; we denote by $c(\ell)$ the number of possible values of $t \bmod \ell$. It remains to combine these values in a clever way, using a *match and sort* technique described in [2]. This paper contains also many ideas concerning the alternative use of other modular equations, that turn out to be essential in practice, but that we do not want to describe here (for this see also [40]).

Elkies [14] remarked that when $t^2 - 4q$ is a square modulo ℓ, then $f_\ell(X)$ has a factor $g_\ell(X)$ of degree $(\ell-1)/2$. Moreover, π_E has an eigenspace associated with g_ℓ, which means that we now look for some k, $1 \leq k < \ell$ such that

$$(X^q, Y^q) = k(X, Y)$$

in $GF(q)[X,Y]/(\mathcal{F}(X,Y,1), g_\ell(X))$; then we recover $t_\ell = (k^2+q)/k \bmod \ell$. This change was crucial, because it was then possible to use polynomials of degree $(\ell-1)/2$ rather than of degree $(\ell^2-1)/2$. Elkies gave an algorithm to compute g_ℓ using further properties of modular equations. Another approach was given in [9].

Atkin [3] gave his own solution to the problem of computing $g_\ell(X)$ using more modular equations and modular forms. Though rather tricky to implement, his approach is very fast in practice.

Recently, Couveignes and Morain showed how to use powers of small Elkies primes [11].

All these ideas are also described in [44] and were implemented [3, 25, 40, 39]. The results are striking, the record being that of the computation of the cardinality of a curve modulo a prime p of 500 digits (see the end of this article).

The only remaining problem was that these ideas could not work when $p = 2$. As a matter of fact, the theory of Atkin and Elkies remains valid, but one could not use the ordinary parameterization of elliptic curves via Weierstrass' $\wp$-functions to get a suitable way of computing g_ℓ. Couveignes solved this problem in his thesis [10], using formal groups as a powerful tool. The first successful implementation of these ideas is due to Lercier and Morain [27].

3.4 Couveignes's algorithm

Couveignes's algorithm [10] works in any characteristic $p > 0$. We simplify the exposition in the case $p = 2$.

When ℓ is an Elkies prime, we known that the initial curve

$$E : y^2 + xy = x^3 + a_6, \tag{3}$$

is isogenous to a curve

$$E^* : y^2 + xy = x^3 + a_6^* \tag{4}$$

that can be easily computed from the modular equation Φ_ℓ. The difficulty lies in the computation of the isogeny I from E to E^* defined by

$$I(x,y) = (U(x), V(x,y)) = \left(\frac{g(x)}{h^2(x)}, \frac{k(x)}{yh^3(x)}\right). \tag{5}$$

Then $h(x)$ is the factor of the division polynomial we are looking for.

Setting $t = -x/y$ and $s = -1/y$, the formal groups defined by (3) is the set of pairs (t, s) satisfying

$$t^3 + ts + a_6 s^3 = s$$

where t and s are formal series in $K((\tau))$. A morphism M from E to E^* satisfies the equality

$$M((t_1(\tau), s_1(\tau)) + (t_2(\tau), s_2(\tau))) = M(t_1(\tau), s_1(\tau)) + M(t_2(\tau), s_2(\tau)) \tag{6}$$

in $K((\tau))$. This equation is not sufficient to get I since there are much more morphisms than isogenies.

Since I can be written as (5), letting $z(\tau) = s(\tau)/t(\tau)$, U is a formal series such that

$$U(\tau) = U(z(\tau)) = z(\tau)\frac{\hat{h}^2(z(\tau))}{\hat{g}(z(\tau))} \tag{7}$$

with $\hat{h}$, a polynomial of degree $(\ell - 1)/2$ and $\hat{g}$, a polynomial of degree ℓ. We write $U(\tau) = \tau + \sum_{i=2}^{\infty} u_i \tau^i$ and we find the u_i's coefficient by coefficient. If i is not a power of 2, we look for u_i such that the equality

$$U((\tau, s(\tau)) + (A\tau, s(A\tau))) = (U(\tau), s^*(U(\tau))) + (U(A\tau), s^*(U(A\tau))), \tag{8}$$

holds up to τ^{i+1}, A being a constant in the field chosen as described in [10]; when i is a power of 2, we do the same thing using

$$U((\tau, s(\tau)) + (\tau, s(\tau))) = (U(\tau), s^*(U(\tau))) + (U(\tau), s^*(U(\tau))). \tag{9}$$

We have to compute $4\ell + 1$ terms of $U(\tau)$ in order to get $4\ell + 2$ terms once substituted in $z^*(\tau) = s^*(t)/t$ and finally have $2\ell + 1$ terms as a series in $z(\tau) = s(\tau)/\tau$, to be able to recognize

$$U(z) = \frac{zg(z)^2}{h(z)}$$

with the Massey–Berlekamp algorithm [28].

3.5 A synthetic description of the algorithm

The general algorithm for computing the cardinality of $E(K)$ runs as follows: we use two variables M_u and M_l which contain respectively the product of primes ℓ for which t_ℓ is known and for which t_ℓ is in some subset of possible values. Typically, M_u contains Elkies primes and M_l Atkin primes. The variable M will contain the current number of combinations to be tried; a bound on M is given as a constant $\mathcal{M}$ (more details on its choice will be given later).

The general procedure is:

procedure `SEA`(K, E)

1. $\ell := 1$; $M_u := 1$; $M_l := 1$; $M := 1$;
2. **while** $(M_u \times M_l < 4\sqrt{q})$ **or** $(M > \mathcal{M})$ **do**
   (a) $\ell :=$ nextprime(ℓ);
   (b) **if** $\ell = p$ **then** `LEqualPCase`(ℓ);
   (c) compute $\Phi(X) = \Phi_\ell(X, j(E))$ and find the number ν of roots of Φ in K;

(d) **if** $\nu = 2$ (ℓ is an Elkies prime) **then** `ElkiesCase`(ℓ);

(e) **if** $\nu = 0$ (ℓ is an Atkin prime) **then** `AtkinCase`(ℓ);

3. use the match and sort technique to finish the computations.

The core of the computations consists of the two procedures:

procedure `ElkiesCase`(ℓ)

1. compute a factor $g_\ell(X)$ of $f_\ell(X)$ of degree $(\ell-1)/2$ using Atkin's algorithm if p is large and Couveignes's otherwise;
2. find an eigenvalue of π_E related to g_ℓ, i.e., $1 \le k < \ell$ such that $(X^q, Y^q) = k(X, Y)$ in $GF(q)[X,Y](\mathcal{F}(X,Y,1), g_\ell(X))$; deduce from this that $t \equiv (k^2+q)/k \bmod \ell$;
3. $M_u := M_u \times \ell$.

procedure `AtkinCase`(ℓ)

1. find the least r such that $X^{q^r} \equiv X \bmod \Phi$ and set $c(\ell) = \varphi(r)$; $M := M \times c(\ell)$;
2. $M_l := M_l \times \ell$.

Details concerning these two procedures are given in [3, 14, 9, 40, 25]. Recent improvements are due to Müller [41] (see also [45]) and Dewaghe [13] and have been incorporated in our programs.

In the case $\ell = p$, covered by procedure `LEqualPCase`, the polynomial $f_{p^e}(X)$ can be written as $P(X)^{p^{e-1}}$ where $P(X)$ is of degree $(p^{e+1} - p)/2$. Schoof's original algorithm or sometimes Elkies' algorithm can be used (see [31] for the case $p = \ell = 2$; the general case will be dealt with in [27]).

3.6 A more elaborate strategy

Let us give a variant of the algorithm we described above using four more constants $\mathcal{A}$, $\mathcal{E}$, $\mathcal{S}$ and $\mathcal{C}$ that will reflect the choice of possible strategies:

(c) **if** $\nu = 2$ **and** $\ell \le \mathcal{E}$ **then**
  1. `ElkiesCase`(ℓ); compute the semi-order d of the eigenvalue $k \bmod \ell$,
   i.e., the smallest d such that $k^d \equiv \pm 1 \bmod \ell$;
  2. **for** $n := 2$ **while** $\ell^{n-1} d \le \mathcal{C}$ **do** compute $t \bmod \ell^n$; $M_u := M_u \times \ell$;
 else if $(\ell^2-1)/2 \le \mathcal{S}$ **then for** $n := 1$ **while** $(\ell^{2n} - \ell^{2n-2})/2 \le \mathcal{S}$ **do**
   compute $t \bmod \ell^n$ using Schoof's original algorithm;
  else if $\ell \le \mathcal{A}$ **then** `AtkinCase`(ℓ);

In the above description, the quantity $\ell^{n-1}d$ represents the degree of a factor of $f_{\ell^n}(X)$, see [11]; $(\ell^{2n} - \ell^{2n-2})/2$ is the degree of a factor of $f_{\ell^n}(X)$.

This presentation captures many possible strategies. First of all, setting $\mathcal{E} = \mathcal{A} = 0$ yields Schoof's original algorithm. Setting $\mathcal{E} = 0$ gives Atkin's first algorithm [2]. Introducing $\mathcal{C}$ makes it possible to use the ideas of [11]. We will detail the constants of our implementation in the next section.

Table 1: Benchmarks for field arithmetic in $GF(2^n)$

n	Squaring	Multiplication	Inversion
65	5.2	27.4	567
89	5.3	30.8	834
105	5.8	33.6	994
155	7.3	63.8	1963
196	8.5	101.6	2835

4 Implementation and results

4.1 General remarks

We note that almost all the ideas (and tricks) of Atkin are still valid when the characteristic is 2. The first implementation of part of the above ideas is described in [32], which contains many interesting details.

4.2 Basic arithmetic

Our implementation is based on the library `GFM` written by F. Chabaud [7] (on top of `BigNum` – cf. [17]), and improved by the authors. It represents $GF(2^n)$ as the residue class ring $GF(2)[T]/(T^n + f(T))$ where $f(T)$ is a polynomial of degree smaller than n such that $T^n + f(T)$ is irreducible over $GF(2)$. In practice – in the range $1 \leq n \leq 600$ – we were always able to find a suitable f of degree less than 15.

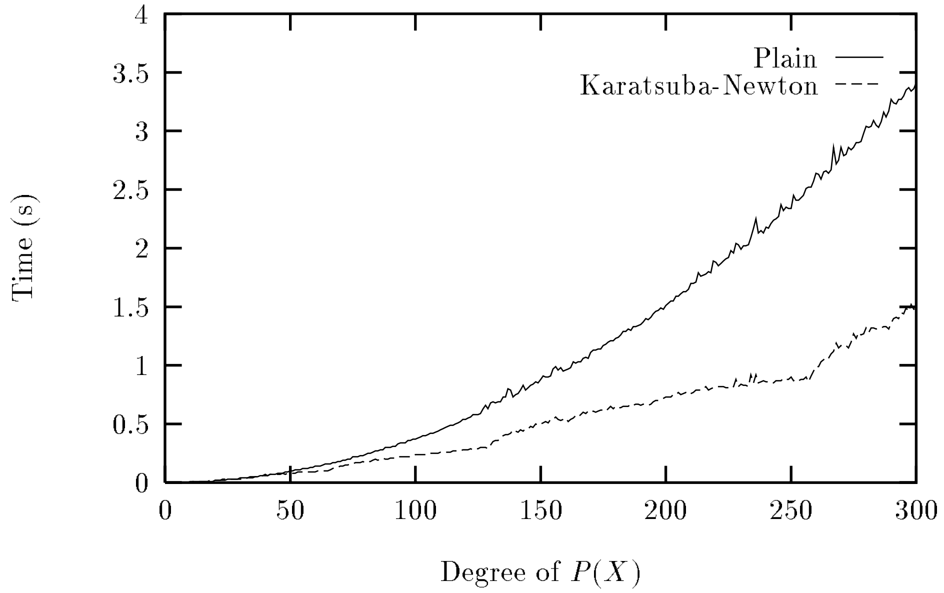
The algorithm spends most of the time doing multiplications of elements in the field. To speed up this operation, we first perform the multiplication of two polynomials with coefficients in $GF(2)$ using a table storing all the products PQ of polynomials P and Q of degree at most 7 (at the expense of a storage of 128 kilo-bytes). Then we reduce this polynomial of degree at most $2n - 2$ modulo $T^n + f(T)$ using a second table storing the coefficients of $q(T)f(T)$ for all $q(T)$ of degree smaller than 15 (at the expense of a storage of 256 kilo-bytes too). Inversion in the field is done as in [31, Chap. 6, pp. 85].

We give in Table 1 containing precise timings (in seconds) for performing 10^6 operations. All benchmarks have been done on a DecAlpha 3000/500.

4.3 Polynomial arithmetic

One of the main costs of the algorithm is the computation of $X^{2^n} \bmod P(X)$ where $P(X) \in GF(2^n)[X]$. As squaring of polynomials of degree d can be performed in $O(d)$ squarings in $GF(2^n)$, we have to improve the reduction of a polynomial $g(X)$ (of degree at most $2d$) modulo a polynomial $P(X)$ (of degree d). This usually costs $O(d^2)$ multiplications in K (cf. [20]), but can be improved using Newton's method and Karatsuba's algorithm as described for instance in [1] (see also [37]). In the graph given in Figure 1, we plotted the time needed to square a polynomial modulo another polynomial in $GF(2^{105})$ for all degree less than 300.

Figure 1: Time for squaring a polynomial over $GF(2^{105})$



4.4 Timings

In [16], the authors give running times for curves defined over $GF(2^{65})$, $GF(2^{89})$ and $GF(2^{105})$. We used these fields as benchmarks for our implementation. We took the 50 curves defined as $y^2 + xy = x^3 + a_6$ where $a_6 \in GF(2)[T]$ and $2 \leq a_6(2) \leq 51$ (none of such coefficient a_6 belongs to a smaller extension of $GF(2^{65})$, $GF(2^{89})$ and $GF(2^{105})$).

We give: ℓ_{max}, the maximal prime used; the number of U (resp. L) primes; M, the number of combinations; the cumulated time for X^q, X^{q^r}, Schoof's algorithm; computing g_ℓ and k when ℓ is Elkies; the time for the match and sort program; the total time. For each category, we give the minimal, maximal and average values.

Consider first $K = GF(2^{65})$. In this case, $\mathcal{M} = \infty$. Results are put in Table 2. These tables show immediately that throwing away Atkin primes is really a bad idea. Playing with the different parameters finally yields the best results for our three fields in Table 3. In each case, one has $\mathcal{A} = \infty$.

4.4.1 A dynamic strategy.

When ℓ is an Elkies prime, the cost of `ElkiesCase` turns out to be greater than that of `AtkinCase`. In order to have a program as fast as possible, it is sometimes better to treat an Elkies prime as an Atkin prime. This motivates our *dynamic* strategy. Let L denote the least prime such that $\prod_{\ell \leq L} \ell > 4\sqrt{q}$ and denote by $\tilde{c}(\ell)$ an upper bound on $c(\ell)$. An upper bound for the number of combinations is then $\prod_{\ell \leq L} \tilde{c}(\ell)$. The program runs as above and as soon as for the current prime

Table 2: Different parameters for $GF(2^{65})$

$\mathcal{E}=\infty,\mathcal{C}=64,\mathcal{S}=84,\mathcal{A}=0$				$\mathcal{E}=0,\mathcal{C}=0,\mathcal{S}=0,\mathcal{A}=\infty$			
	min	max	avg		min	max	avg
$\ell_{\max}$	19	53	35	$\ell_{\max}$	31	31	31
$\#U$	6	9	7	$\#U$	1	3	2
$\#L$	0	10	4	$\#L$	8	10	9
M	1	1	1	M	$1.02\,10^3$	$1.47\,10^6$	$2.79\,10^5$
X^q	1.5	18.9	7.2	X^q	4.7	4.9	4.8
X^{q^r}	0.0	0.0	0.0	X^{q^r}	1.1	3.9	3.0
Schoof	0.0	71.5	17.4	Schoof	0.0	0.0	0.0
g	17.4	337.0	106.0	g	0.0	0.0	0.0
k	5.1	27.3	14.7	k	0.0	0.0	0.0
M-S	0.0	0.1	0.1	M-S	0.1	2.1	1.3
Total	26.9	410.0	146.0	Total	7.2	10.5	9.1

ℓ, one has

$$\left(\prod_{l<\ell} c(l)\right)\left(\prod_{\ell\le l\le L} \tilde{c}(l)\right) < \mathcal{M}$$

one decides to treat the remaining primes as Atkin primes.

We can compute an upper bound $\tilde{c}(\ell)$ for $c(\ell)$ as $\tilde{c}(\ell) = \max\{\varphi(r)\}$ where $r \mid \ell - \varepsilon$ and $(q/\ell) = (-1)^{(\ell-\varepsilon)/r}$ for all choices of ε in $\{\pm 1\}$. (Note that $\varepsilon = +1$ if ℓ is an Elkies prime and -1 otherwise.)

The results of this strategy are as follows, with $\mathcal{A} = \infty$ in all cases: This shows that this strategy is useful in the last case only.

4.5 Comparison with the case $GF(p)$

For the sake of comparisons, we give some timings when using the field $GF(p)$ where p is the least prime greater than 2^{65} (resp. 2^{89}, etc.). We considered the 50 random curves of equation $y^2 = x^3 + x + b$ for $1 \le b \le 50$ for each of these primes. In all cases, one has $\mathcal{A} = \infty$ and $\mathcal{C} = 0$. Results are given in Table 5.

4.6 Records

In [32, 31], the authors gave timings for larger fields $GF(2^{155})$ and $GF(2^{195})$. For these fields and for larger fields (the last one being the current record, as of February 1995), our implementation gave the following timings, for the curve:

$$E_X : y^2 + xy = x^3 + T^{16} + T^{14} + T^{13} + T^9 + T^8 + T^7 + T^6 + T^5 + T^4 + T^3$$

(the coefficient was chosen as the binary expression of 91128 – our zip code – converted to a polynomial if $GF(2^n)$). Table 6 corresponds to the first version of our implementation, using an

Table 3: Best parameters for $GF(2^n)$

	$GF(2^{65})$ $\mathcal{E}=2,\mathcal{C}=2,\mathcal{S}=0$ min	max	avg	$GF(2^{89})$ $\mathcal{E}=3,\mathcal{C}=4,\mathcal{S}=0$ min	max	avg	$GF(2^{105})$ $\mathcal{E}=3,\mathcal{C}=4,\mathcal{S}=24$ min	max	avg
ℓmax	29	29	29	37	41	39	43	43	43
$\#U$	1	4	2	1	6	3	4	6	5
$\#L$	6	9	7	6	12	9	8	10	9
M	10^3	$3.7\,10^5$	$5.8\,10^4$	$7.7\,10^2$	$2.8\,10^7$	$2.4\,10^6$	$1.5\,10^5$	$7.1\,10^8$	$6.6\,10^7$
X^q	3.8	4.0	3.9	10.2	14.9	12.5	22.4	24.8	23.3
X^{q^r}	1.3	3.2	2.2	3.8	12.0	8.1	11.5	18.3	14.7
Schoof	0.0	0.0	0.0	0.0	0.0	0.0	0.0	30.0	12.9
g	0.0	1.1	0.4	0.0	2.5	1.0	0.0	2.4	1.0
k	0.1	0.2	0.1	0.3	0.6	0.5	0.3	0.8	0.6
M-S	0.1	1.7	1.1	0.2	5.9	2.5	0.5	18.8	5.7
Total	6.1	8.8	7.7	17.9	32.2	24.6	43.0	73.9	58.1

incremental search for the isogeny relying on a lot of composition of series, but in such a way that we could not use fast algorithms for series computations. The gap between $GF(2^{400})$ and $GF(2^{500})$ is due to a simplification of the formulas used by Couveignes in his work.

Table 7 refers to the new implementation that uses another approach, enabling one to use fast algorithms for doing series computations. The comparison for the case $GF(2^{300})$ is striking. More details will be given in [27]. We also add Table 8 which gives timings for fields $GF(p)$ with p a large prime. It seems that the program for $GF(2^n)$ is slower than the program for large prime characteristic around $n = 150$.

The record for the large prime case (as of February 1995) is our computation of the cardinality of $E : Y^2 = X^3 + 4589X + 91228$ modulo $p = 10^{499} + 153$ (4589 is the extension number of one of us). It took roughly 4200 hours of DEC 3000 - M300X including 2900 hours for the computation of various X^p. We had to consider 163 primes less than 1000, out of which 71 were of type L.

5 Conclusion

It should be apparent from the preceding tables that the implementation of Schoof's algorithm in characteristic 2 is somewhat faster than in large characteristic, at least for small fields. As a matter of fact, asymptotically, the large prime case is faster. One of the reasons for this is that polynomial arithmetic is faster for $GF(2^n)$, since X^q consists of squarings only, and squaring is an easy operation in characteristic 2. As n increases, the cost of computing the isogeny takes much more time than in the large prime case. There is still room for many improvements in that direction. We think that the situation might evolve very rapidly soon.

Acknowledgments. First of all, the authors want to express their gratitude to J.-M. Couveignes, without whom this work could not have been possible. We also thank one of the referees for suggesting a title corresponding more closely to the content of this article.

Table 4: Best parameters for the dynamic strategy

	$GF(2^{65})$ $\mathcal{C}=16,\mathcal{S}=4,\mathcal{M}=10^6$			$GF(2^{89})$ $\mathcal{C}=16,\mathcal{S}=4,\mathcal{M}=10^8$			$GF(2^{105})$ $\mathcal{C}=32,\mathcal{S}=12,\mathcal{M}=10^{10}$		
	min	max	avg	min	max	avg	min	max	avg
ℓmax	29	29	29	37	41	37	41	43	42
$\#U$	2	4	2	2	5	3	2	5	3
$\#L$	6	8	80	7	10	9	9	12	10
M	10^3	$3.7\,10^5$	$5.3\,10^4$	$3\,10^3$	$2.7\,10^7$	$2.9\,10^6$	$7.4\,10^4$	$9.4\,10^8$	$8.3\,10^7$
X^q	3.3	3.5	3.4	9.0	12.3	9.4	15.6	22.2	20.2
X^{q^r}	1.1	2.7	1.9	3.2	8.6	5.3	8.7	15.9	12.3
Schoof	0.0	3.3	1.8	0.0	3.4	1.7	0.0	11.0	2.5
g	0.0	0.2	0.1	0.0	0.9	0.2	0.0	2.1	0.7
k	0.1	0.2	0.2	0.3	0.9	0.6	0.4	2.2	0.9
M-S	0.1	1.5	1.0	0.2	5.7	2.5	0.6	27.3	6.7
Total	6.0	10.8	8.3	15.4	25.0	19.6	32.0	65.2	43.3

References

[1] Aho, A. V., Hopcroft, J. E. , and Ullman, J. D. *The design and analysis of computer algorithms*. Reading. Addison–Wesley, 1974.

[2] Atkin, A. O. L. The number of points on an elliptic curve modulo a prime. Draft, 1988.

[3] Atkin, A. O. L. The number of points on an elliptic curve modulo a prime (ii). Draft, 1992.

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

[5] Bender, A., and Castagnoli, G. On the implementation of elliptic curve cryptosystems. In *Advances in Cryptology* (1989), G. Brassard, Ed., vol. 435 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 186–192. Proc. Crypto '89, Santa Barbara, August 20–24.

[6] Bosma, W. Primality testing using elliptic curves. Tech. Rep. 85-12, Math. Instituut, Universiteit van Amsterdam, 1985.

[7] Chabaud, F. Sécurité des crypto-systèmes de McEliece. Mémoire de DEA, École polytechnique, 1993.

[8] Chao, J., Tanada, K., and Tsujii, S. Design of elliptic curves with controllable lower boundary of extension degree for reduction attacks. In *Advances in Cryptology – CRYPTO '94* (1994), Y. Desmedt, Ed., vol. 839 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 50–55. Proc. 14th Annual International Cryptology Conference, Santa Barbara, Ca, USA, August 21–25.

[9] Charlap, L. S., Coley, R., and Robbins, D. P. Enumeration of rational points on elliptic curves over finite fields. Draft, 1991.

Table 5: Best running times for $GF(p)$

	$GF(2^{65}+131)$ $\mathcal{E}=7, \mathcal{S}=0$			$GF(2^{89}+29)$ $\mathcal{E}=\infty, \mathcal{S}=3$			$GF(2^{105}+39)$ $\mathcal{E}=\infty, \mathcal{S}=3$		
	min	max	avg	min	max	avg	min	max	avg
$\ell_{\max}$	29	31	30	41	43	41	43	47	46
$\#U$	2	6	3	5	13	7	3	13	8
$\#L$	5	9	7	1	8	5	2	11	6
M	$1.7\,10^3$	$6.6\,10^5$	$1.\,10^5$	8	$2.0\,10^6$	$6.1\,10^4$	32	$1.1\,10^7$	$6.0\,10^05$
X^q	1.1	2.2	1.8	0.3	4.2	2.3	0.3	10.9	4.7
X^{q^r}	1.1	2.2	1.8	0.3	4.2	2.3	0.3	10.9	4.7
Schoof	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
g	0.0	0.0	0.0	0.1	0.9	0.4	0.2	1.8	0.9
k	0.0	0.1	0.0	0.7	8.1	3.7	1.1	13.5	7.7
$M-S$	0.3	1.3	0.6	0.4	2.4	0.6	0.5	6.9	1.1
Total	6.3	10.0	8.8	23.5	32.9	25.3	40.1	60.4	51.6

[10] Couveignes, J.-M. *Quelques calculs en théorie des nombres.* Thèse, Université de Bordeaux I, July 1994.

[11] Couveignes, J.-M., and Morain, F. Schoof's algorithm and isogeny cycles. In preparation, February 1995. Preliminary version appeared in *ANTS-I* (1994), L. Adleman and M.-D. Huang, Eds., vol. 877 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 43–58. 1st Algorithmic Number Theory Symposium - Cornell University, May 6-9, 1994.

[12] Demytko, N. A new elliptic curve based analogue of RSA. In *Advances in Cryptology – EUROCRYPT '93* (1994), T. Helleseth, Ed., vol. 765 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 40–49. Workshop on the Theory and Application of Cryptographic Techniques, Lofthus, Norway, May 23–27, 1993.

[13] Dewaghe, L. Remarques sur l'algorithme SEA. In preparation, Dec. 1994.

[14] Elkies, N. D. Explicit isogenies. Draft, 1991.

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

[16] Harper, G., Menezes, A., and Vanstone, S. Public-key cryptosystems with very small key length. In *Advances in Cryptoloy – EUROCRYPT '92* (1993), R. A. Rueppel, Ed., vol. 658 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 163–173. Workshop on the Theory and Application of Cryptographic Techniques, Balatonfüred, Hungary, May 24–28, 1992, Proceedings.

[17] Hervé, J.-C., Serpette, B., and Vuillemin, J. BigNum: A portable and efficient package for arbitrary-precision arithmetic. Tech. Rep. 2, Digital Paris Research Laboratory, May 1989.

[18] Kaliski, Jr., B. S. A pseudo-random bit generator based on elliptic logarithms. In *Proc. Crypto 86* (1986), vol. 263 of *Lecture Notes in Comput. Sci.* Proceedings Crypto '86, Santa Barbara (USA), August 11–15, 1986.

Table 6: Records for the first implementation

	$GF(2^{155})$	$GF(2^{196})$	$GF(2^{300})$	$GF(2^{400})$	$GF(2^{500})$	$GF(2^{601})$
ℓ_{max}	59	73	109	173	179	241
$\#U$	8	12	20	26	27	29
$\#L$	9	9	9	13	11	23
M	$1.21\,10^6$	$1.06\,10^8$	$1.2\,10^{10}$	$2.5\,10^9$	$1.3\,10^7$	$2.1\,10^{10}$
X^q	128.7	453.5	15886	92643	29137	109708
X^{q^r}	40.2	195.9	47562	94965	6106	52885
Schoof	0	3.7	12194	186607	65799	240091
g	334.5	1714.7	672994	1119077	518697	3139250
k	46.8	116.6	40655	774895	492213	1392113
M-S	59	698	7183	27088	3609	1728
Total	609	3183	796474	2511000	1112093	4935776

Table 7: Records for the second implementation

	$GF(2^{155})$	$GF(2^{196})$	$GF(2^{300})$
$\ell_{\max}$	59	73	109
$\#U$	6	11	18
$\#L$	11	10	11
M	$2\,10^7$	10^8	$3\,10^8$
X^q	121	440	3221
X^{q^r}	42	127	356
Schoof	0	69	0
g	24	580	22974
k	19	141	3613
M-S	10	23	56
Total	217	1381	30221

Table 8: Comparisons with $GF(p)$

	$GF(2^{155}+15)$	$GF(2^{196}+21)$	$GF(2^{300}+157)$
$\ell_{\max}$	67	79	113
$\#U$	11	16	20
$\#L$	8	6	10
M	$1.5\,10^6$	$6.1\,10^5$	$6.6\,10^8$
X^q	126.2	307.5	1575.3
X^{q^r}	16.6	6.8	97.3
Schoof	0.1	0.1	0.2
g	1.8	7.1	19.4
k	25.1	113.1	491.9
M-S	3.3	2.8	104.4
Total	179.1	443.7	2350.1

[19] Kaliski, Jr., B. S. One-way permutations on elliptic curves. *Journal of Cryptology 3*, 3 (1990), 187–199.

[20] Knuth, D. E. *The Art of Computer Programming: Seminumerical Algorithms.* Addison-Wesley, 1981.

[21] Koblitz, N. Elliptic curve cryptosystems. *Math. Comp. 48*, 177 (Jan. 1987), 203–209.

[22] Koblitz, N. Elliptic curve implementation of zero-knowledge blobs. *Journal of Cryptology 4*, 3 (1991), 207–213.

[23] Koyama, K., Maurer, U. M., Okamoto, T., and Vanstone, S. A. New public-key schemes based on elliptic curves over the ring Z_n. In *Advances in Cryptology* (1991), vol. 576 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 252–266. Proc. Crypto '91, Santa Barbara, August 12–15.

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

[25] Lehmann, F., Maurer, M., Müller, V., and Shoup, V. Counting the number of points on elliptic curves over finite fields of characteristic greater than three. In *ANTS-I* (1994), L. Adleman and M.-D. Huang, Eds., vol. 877 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 60–70. 1st Algorithmic Number Theory Symposium - Cornell University, May 6-9, 1994.

[26] Lenstra, Jr., H. W. Factoring integers with elliptic curves. *Annals of Math. 126* (1987), 649–673.

[27] Lercier, R., and Morain, F. Counting the number of points on elliptic curves over finite fields of characteristic 2. In preparation, Oct. 1994.

[28] Massey, J. L. Shift-register and BCH decoding. *IEEE Trans. on Information Theory IT-15*, 1 (Jan. 1969), 122–127.

[29] Menezes, A., Okamoto, T., and Vanstone, S. A. Reducing elliptic curves logarithms to logarithms in a finite field. In *Proceedings 23rd Annual ACM Symposium on Theory of Computing (STOC)* (1991), ACM Press, pp. 80–89. May 6–8, New Orleans, Louisiana.

[30] Menezes, A., and Vanstone, S. A. The implementation of elliptic curve cryptosystems. In *Advances in Cryptology* (1990), J. Seberry and J. Pieprzyk, Eds., no. 453 in Lecture Notes in Comput. Sci., Springer–Verlag, pp. 2–13. Proceedings Auscrypt '90, Sysdney (Australia), January 1990.

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

[32] Menezes, A. J., Vanstone, S. A., and Zuccherato, R. J. Counting points on elliptic curves over F_{2^m}. *Math. Comp. 60*, 201 (Jan. 1993), 407–420.

[33] Miller, V. Use of elliptic curves in cryptography. In *Advances in Cryptology* (1987), A. M. Odlyzko, Ed., vol. 263 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 417–426. Proceedings Crypto '86, Santa Barbara (USA), August11–15, 1986.

[34] Miyaji, A. On ordinary elliptic curve cryptosystems. In *Advances in Cryptology – ASIACRYPT '91* (1991), vol. 739 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 50–55.

[35] Miyaji, A. Elliptic curves over F_p suitable for cryptosystems. In *Advances in cryptology - AUSCRYPT '92* (1993), J. Seberry and Y. Zheng, Eds., vol. 718 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 479–491. Workshop on the theory and application of cryptographic techniques, Gold Coast, Queensland, Australia, December 13-16, 1992.

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

[37] Montgomery, P. L. *An FFT extension of the Elliptic Curve Method of factorization.* PhD thesis, University of California – Los Angeles, 1992.

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

[39] Morain, F. Implantation de l'algorithme de Schoof-Elkies-Atkin. Preprint, January, 1994.

[40] Morain, F. Calcul du nombre de points sur une courbe elliptique dans un corps fini : aspects algorithmiques. To appear in the Actes des Journées Arithmétiques 1993, Feb. 1995.

[41] Müller, V. Looking for the eigenvalue in Schoof's algorithm. In preparation, Oct. 1994.

[42] Okamoto, T., Fujikoda, A., and Fujisaki, E. An efficient digital signature scheme based on an elliptic curve over the ring Z_n. In *Advances in Cryptology – CRYPTO '92* (1992), vol. 740 of *Lecture Notes in Comput. Sci.*, Springer-Verlag, pp. 54–65.

[43] SCHOOF, R. Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comp. 44* (1985), 483–494.

[44] SCHOOF, R. Counting points on elliptic curves over finite fields. To appear in Proc. Journées Arithmétiques 93, Jan. 1995.

[45] SHOUP, V. A new polynomial factorization algorithm and its implementation. Preprint, 1994.

[46] SILVERMAN, J. H. *The arithmetic of elliptic curves*, vol. 106 of *Graduate Texts in Mathematics*. Springer, 1986.