MATHEMATICS OF COMPUTATION
VOLUME 60, NUMBER 201
JANUARY 1993, PAGES 407–420

COUNTING POINTS ON ELLIPTIC CURVES OVER F_{2^m}

ALFRED J. MENEZES, SCOTT A. VANSTONE, AND ROBERT J. ZUCCHERATO

ABSTRACT. In this paper we present an implementation of Schoof's algorithm for computing the number of F_{2^m}-points of an elliptic curve that is defined over the finite field F_{2^m}. We have implemented some heuristic improvements, and give running times for various problem instances.

1. INTRODUCTION

The use of elliptic curves in public key cryptography was first proposed by N. Koblitz [5] and V. Miller [10]. Since then, a significant amount of research has been done on the implementation of practical and secure cryptosystems based on elliptic curves. For a secure system, one should select a curve E over a finite field F_q such that the order, $\#E(F_q)$, of the group of points has a large prime divisor. There are some families of curves whose orders are trivial to compute (see [7] for some examples). However, if a random curve is chosen, then it is necessary to have an efficient algorithm for computing its order.

In 1985, Schoof [13] gave a polynomial-time algorithm for computing $\#E(F_q)$. The algorithm has a running time of $O(\log^8 q)$ bit operations, and is rather cumbersome in practice. In [3] the authors combined Schoof's algorithm with Shanks' baby-step giant-step algorithm, and were able to compute orders of curves over F_p, where p is a 27-decimal-digit prime. The algorithm took 4.5 hours on a SUN-1 SPARC station.

The work mentioned above was all described for the case q odd. From a practical point of view, however, curves over fields of characteristic 2 are more attractive, since the arithmetic in F_{2^m} is easier to implement in hardware than the arithmetic in F_q, q odd. In [6] Koblitz adapted Schoof's algorithm to curves over F_{2^m} and studied the implementation and security of a random-curve cryptosystem. Special emphasis was placed on the underlying field $F_{2^{135}}$. Recently, Agnew, Mullin, and Vanstone [1] have developed a VLSI device to perform arithmetic in $F_{2^{155}}$ and to perform computations on a random elliptic curve over this field. Consequently, it is of interest to determine the order of random curves over $F_{2^{155}}$.

We have implemented Schoof's algorithm for counting the points on an arbitrary curve over F_{2^m}, and have employed some heuristics to improve the actual running time. We are able to compute $\#E(F_{2^m})$ for $m = 155$ (and so $\#E(F_{2^m}) \approx 10^{47}$) in about 61 hours on a SUN-2 SPARC station.

Received by the editor February 20, 1992.

1991 *Mathematics Subject Classification.* Primary 11Y40, 11T71; Secondary 94A60, 68P25, 11Y16.

©1993 American Mathematical Society
0025-5718/93 \$1.00 + \$.25 per page

The remainder of the paper is organized as follows. In §2, we mention the relevant properties of elliptic curves over finite fields of characteristic 2. In §3, we outline Schoof's algorithm, and in §4 we present our heuristics for improving Schoof's algorithm. Section 5 discusses details of our implementation, and gives some running times for various problem instances. Finally, in §6, we survey the latest research on the problem of counting points on an elliptic curve.

2. Elliptic curves in characteristic 2

Let $q = 2^m$, and let $K = F_q$ be the finite field of q elements. We denote the algebraic closure of K by $\overline{K}$. If S is a field or an additive group, then S^* will denote the nonzero elements of S. There are two types of elliptic curves over K. A *supersingular* curve E over K is the set of solutions $(x, y) \in \overline{K} \times \overline{K}$ to an equation of the form

$$y^2 + a_3y = x^3 + a_4x + a_6, \tag{1}$$

with $a_3, a_4, a_6 \in K$, $a_3 \neq 0$, together with the "point at infinity" denoted $\mathcal{O}$. A *nonsupersingular* curve E over K is the set of solutions $(x, y) \in \overline{K} \times \overline{K}$ to an equation of the form

$$y^2 + xy = x^3 + a_2x^2 + a_6, \tag{2}$$

with $a_2, a_6 \in K$, $a_6 \neq 0$, together with the point $\mathcal{O}$.

If L is any field with $K \subseteq L \subseteq \overline{K}$, then let $E(L)$ denote the set of points in E both of whose coordinates lie in L, together with the point $\mathcal{O}$.

There are precisely three isomorphism classes of supersingular elliptic curves over K if m is odd, and seven classes if m is even. The number of points on a curve in each class is known [9]. Given a supersingular curve (1), we can then compute $\#E(K)$ by determining the isomorphism class that E belongs to. For the remainder of the paper we will thus be interested in computing $\#E(K)$, where E is a nonsupersingular elliptic curve.

There are $2q - 2$ isomorphism classes of nonsupersingular curves over K. A set of representatives of these classes is

$$\{y^2 + xy = x^3 + a_2x^2 + a_6 | a_6 \in K^*, \ a_2 \in \{0, \gamma\}\},$$

where $\gamma \in K$ is a fixed element of trace 1. If E and $\widetilde{E}$ are the curves $y^2+xy = x^3 + a_6$ and $y^2 + xy = x^3 + \gamma x^2 + a_6$, respectively, then it is easily verified that $\#E(K) + \#\widetilde{E}(K) = 2q + 2$. Henceforth we will always assume that the equation for E is of the form

$$y^2 + xy = x^3 + a_6, \qquad a_6 \in K^*. \tag{3}$$

It is well known that E has the structure of an abelian group, with the point $\mathcal{O}$ serving as its identity element. The rules for adding points on the curve (3) are the following. Let $P = (x_1, y_1) \in E^*$; then $-P = (x_1, y_1+x_1)$. Notice that P and $-P$ have the same x-coordinates. If $Q = (x_2, y_2) \in E^*$ and $Q \neq -P$, then $P + Q = (x_3, y_3)$, where

$$x_3 = \begin{cases} \left(\dfrac{y_1 + y_2}{x_1 + x_2}\right)^2 + \dfrac{y_1 + y_2}{x_1 + x_2} + x_1 + x_2, & P \neq Q, \\ \dfrac{a_6}{x_1^2} + x_1^2, & P = Q, \end{cases}$$

and

$$y_3 = \begin{cases} \left(\dfrac{y_1+y_2}{x_1+x_2}\right)(x_1+x_3)+x_3+y_1, & P \neq Q, \\ x_1^2 + \left(x_1 + \dfrac{y_1}{x_1}\right)x_3 + x_3, & P = Q. \end{cases}$$

If $K \subseteq L \subseteq \overline{K}$, then $E(L)$ is a subgroup of E. If L is finite with $\#L = q^r$, then Hasse's theorem states that

$$\#E(L) = q^r + 1 - t,$$

where $|t| \leq 2\sqrt{q^r}$. Thus, to compute $\#E(L)$, it suffices to compute t.

Let n be a positive integer, and let $\mathbb{Z}_n$ denote the cyclic group of n elements. The group $E(K)$ has rank either 1 or 2; we can write $E(K) \cong \mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2}$, where $n_2|n_1$ and $n_2|q-1$. A point $P \in E$ is called an n-torsion point if $nP = \mathcal{O}$. Let $E[n]$ denote the group of n-torsion points in E. If $\gcd(n, q) = 1$, then $E[n] \cong \mathbb{Z}_n \oplus \mathbb{Z}_n$. If $n = 2^e$, then either $E[n] \cong \{\mathcal{O}\}$ if E is supersingular, or else $E[n] \cong \mathbb{Z}_n$ if E is nonsupersingular.

We introduce the division polynomials $f_n \in K[x]$ associated with the nonsupersingular curve E given by the equation (2) (see [6]):

$$f_0 = 0, \quad f_1 = 1, \quad f_2 = x, \quad f_3 = x^4 + x^3 + a_6, \quad f_4 = x^6 + a_6 x^2,$$

$$f_{2n+1} = f_n^3 f_{n+2} + f_{n-1} f_{n+1}^3, \qquad n \geq 2, \tag{4}$$

$$x f_{2n} = f_{n-1}^2 f_n f_{n+2} + f_{n-2} f_n f_{n+1}^2, \qquad n \geq 3. \tag{5}$$

The polynomials f_n are monic in x, and if n is odd, then the degree of f_n is $(n^2-1)/2$. The division polynomials have the following useful properties which will enable us to perform computations in $E[n]$. Theorem 1 is from [8], while Theorem 2 is from [6].

Theorem 1. *Let $P = (x, y) \in E^*$ and let $n \geq 0$. Then $P \in E[n]$ if and only if $f_n(x) = 0$.*

Theorem 2. *Let $n \geq 2$, and let $P = (x, y) \in E^*$ with $nP \neq \mathcal{O}$. Then*

$$nP = \left(x + \frac{f_{n-1}f_{n+1}}{f_n^2}, \; x + y + \frac{f_{n-1}f_{n+1}}{f_n^2} + \frac{f_{n-2}f_{n+1}^2}{x f_n^3} + (x^2+y)\frac{f_{n-1}f_{n+1}}{x f_n^2}\right).$$

The ring of endomorphisms of E that are defined over K is denoted by $\mathrm{End}_K E$. The map $\phi \in \mathrm{End}_K E$ sending (x, y) to (x^q, y^q) and fixing $\mathcal{O}$ is called the Frobenius endomorphism of E. In $\mathrm{End}_K E$, ϕ satisfies the relation

$$\phi^2 - t\phi + q = 0$$

for a unique $t \in \mathbb{Z}$. In fact, $t = q + 1 - \#E(K)$. If l is an odd prime, then $E[l]$ can be viewed as a vector space over F_l; the vector space has dimension 2. The map ϕ restricted to $E[l]$ is a linear transformation on $E[l]$ with characteristic equation $\phi^2 - t\phi + q = 0$.

3. Outline of Schoof's algorithm

We give an outline of Schoof's algorithm for computing $\#E(K)$, where $K = F_q$, $q = 2^m$, and E is given by equation (3). The method in [13] is described for fields of odd characteristic. More details for the case q even will be given in §4.

Let $\#E(F_q) = q + 1 - t$. Choose a prime L' such that $\prod l > 4\sqrt{q}$, where the product ranges over all primes l from 3 to L'. We proceed to compute $t \pmod{l}$ for each odd prime $l \le L'$; since $|t| \le 2\sqrt{q}$, we can then recover t by the Chinese Remainder Theorem.

Let $P = (\overline{x}, \overline{y}) \in E[l]^*$, and let $k \equiv q \pmod{l}$, $0 \le k \le l - 1$. We search for an integer τ, $0 \le \tau \le l - 1$, such that

$$\phi^2(P) + kP = \tau\phi(P). \tag{6}$$

Since $\phi^2(P) + kP = t\phi(P)$, we deduce that $(t - \tau)\phi(P) = \mathcal{O}$, and hence $t \equiv \tau \pmod{l}$. The problem with implementing this idea is that the coordinates of P, which are in $\overline{K}$, may not lie in any small extension of K, and thus cannot be efficiently found in general. We overcome this problem by observing that $\overline{x}$ is a root of the division polynomial $f_l(x) \in K[x]$. Moreover, we can use Theorem 2 to obtain an expression for kP and $\tau\phi(P)$, where the coordinates of the expressions are rational functions in x and y. We may then use the addition rules to sum $\phi^2(P)$ and kP.

To test whether there exists some $P = (x, y) \in E[l]^*$ satisfying (6), we equate the x-coordinates of the expressions for $\phi^2(P) + kP$ and $\tau\phi(P)$, and eliminate denominators and the variable y to obtain an equation $h_1(x) = 0$. We then compute $H_1(x) = \gcd(h_1(x), f_l(x))$. If $H_1(x) = 1$, then there is no $P \in E[l]^*$ satisfying (6). If $H_1(x) \ne 1$, then there exists $P \in E[l]^*$ with $\phi^2(P) + kP = \pm\tau\phi(P)$. To determine the sign, we equate the y-coordinates of the expressions for $\phi^2(P) + kP$ and $\tau\phi(P)$, eliminate denominators and the variable y to obtain an equation $h_2(x) = 0$, and then compute $H_2(x) = \gcd(h_2(x), f_l(x))$. If $H_2(x) \ne 1$, then P satisfies (6), otherwise P satisfies $\phi^2(P) + kP = -\tau\phi(P)$. Note that all computations now take place in the ring $K[x]$.

The running time of $O(\log^8 q)$ bit operations is obtained as follows. We have that $L' = O(\log q)$. For each l, the search for τ satisfying (6) is dominated by the computations of the residues of x^{q^2} and y^{q^2} modulo $f_l(x)$ (note that $\phi^2(P) = (x^{q^2}, y^{q^2})$). Since the degree of $f_l(x)$ is $O(\log^2 q)$, these residues can be computed in $O(\log^5 q)$ field operations, or $O(\log^7 q)$ bit operations. If fast multiplication techniques are used for multiplication in $K[x]$ and in F_q, then the total running time reduces to $O(\log^{5+\varepsilon} q)$, for any $\varepsilon > 0$. However, since the fast multiplication techniques are only practical for very large q, we will henceforth only use classical multiplication algorithms.

4. Some heuristics

Again, we assume that $K = F_q$, where $q = 2^m$, and that the curve E has equation (3). Let $\#E(F_q) = q + 1 - t$, where $|t| \le 2\sqrt{q}$. From the expression for the division polynomial f_4 we can deduce that $\#E(F_q) \equiv 0 \pmod{4}$, so we can easily determine $t \pmod{4}$.

In §§4.1 and 4.2 we describe how to find $t \pmod{l}$, where l is an odd prime.

4.1. **Finding an eigenvalue of ϕ, if one exists.** Recall from §2 that when viewing ϕ as a linear transformation on $E[l]$, the characteristic equation of ϕ

is $\phi^2 - t\phi + q = 0$. Thus, ϕ has eigenvalues in F_l if and only if either $t^2 - 4q$ is a quadratic residue mod l, or $t^2 - 4q$ is $0 \bmod l$. Assume that s, r are eigenvalues of ϕ in F_l. The following two observations are useful.

- Since $s^2 - ts + q = 0$, we have $t \equiv s + q/s \pmod{l}$.
- If $s \neq r$, then let S denote the set of x-coordinates of nonzero points in the one-dimensional eigenspace corresponding to s. Observe that if $\alpha \in S$, then $\alpha^q \in S$; it follows that $f(x) = \prod_{\alpha \in S}(x - \alpha)$ is a degree $(l-1)/2$ factor of $f_l(x)$ in $K[x]$.

Let w be an integer, $1 \leq w \leq (l-1)/2$. To test whether $\pm w$ is an eigenvalue of ϕ, we have to check if there exists $P = (x, y) \in E[l]^*$ with $\phi(P) = \pm wP$. Explicitly, we equate the x-coordinates of $\phi(P)$ and $\pm wP$ to obtain

$$x^q = x + \frac{f_{w-1}f_{w+1}}{f_w^2}.$$

Thus, the search is successful if and only if

$$g_1(x) = \gcd((x^q + x)f_w^2 + f_{w-1}f_{w+1}, f_l) \neq 1. \tag{7}$$

The dominant step in these calculations is the computation of x^q modulo $f_l(x)$.

If $g_1(x) \neq 1$, then we need to test if $\phi(P) = wP$ or $\phi(P) = -wP$. The roots of $g_1(x)$ are the x-coordinates of points $P \in E[l]^*$ satisfying $\phi(P) = \pm wP$. If the eigenvalues of ϕ are w and $-w$, then $t \equiv 0 \pmod{l}$, and this will be detected since the degree of $g_1(x)$ will be $l-1$. If the eigenvalues of ϕ are the same, then $g_1(x) = f_l(x)$ or the degree of $f_l(x)$ is $(l-1)/2$. Otherwise, if either w or $-w$ (but not both) is one of the two eigenvalues of ϕ in F_l, then the degree of $g_1(x)$ is $(l-1)/2$. In the following computations, all polynomials in x are reduced modulo $g_1(x)$. Equating y-coordinates of $\phi(P)$ and $-wP$, and clearing denominators, we obtain the equation

$$h(x, y) = xf_w^3(y + y^q) + f_{w-2}f_{w+1}^2 + (x^2 + y)f_{w-1}f_wf_{w+1} = 0. \tag{8}$$

Since $y^2 = x^3 + a_6 + xy$, we can compute y^q by repeatedly squaring y^2. After $m-1$ squarings, we obtain

$$y^q = a(x) + b(x)y,$$

with $a(x)$ and $b(x)$ both reduced modulo $g_1(x)$. Equation (8) then reduces to

$$\overline{a}(x) + \overline{b}(x)y = 0.$$

Substituting $y = \overline{a}(x)/\overline{b}(x)$ into the equation of the curve (3) yields the following equation of the curve:

$$\overline{h}(x) = \overline{a}(x)^2 + \overline{a}(x)\overline{b}(x)x + (x^3 + a_6)\overline{b}(x)^2 = 0.$$

Finally, if $\gcd(\overline{h}(x), g_1(x)) = 1$, then $t \equiv w + q/w \pmod{l}$, otherwise $t \equiv -w - q/w \pmod{l}$.

We comment that this method of searching for eigenvalues of ϕ easily extends to the case q an odd prime power.

4.2. **Schoof's algorithm.** If there is no eigenvalue of ϕ in F_l, i.e., if $t^2 - 4q$ is a quadratic nonresidue mod l, then we apply Schoof's test to determine the τ satisfying (6).

We first check if there is a $P = (x, y) \in E[l]^*$ with $\phi^2(P) = \pm kP$, where k is q modulo l. This is the case if and only if

$$\gcd((x^{q^2} + x)f_k^2 + f_{k-1}f_{k+1}, f_l) \neq 1.$$

Observe that if $t \equiv 0 \pmod{l}$, then $\phi^2(P) = -kP$. Now, if $\phi^2(P) = kP$, then $\phi(P) = (2k/t)P$, whence ϕ has an eigenvalue in F_l. But $t^2 - 4q$ is a quadratic nonresidue mod l, so we conclude that $\phi^2(P) = -kP$. Thus, $t\phi(P) = \mathcal{O}$ and $t \equiv 0 \pmod{l}$.

Assume now that there is no $P \in E[l]^*$ with $\phi^2(P) = \pm kP$. In order to determine $t \pmod{l}$, we check for each τ, $1 \leq \tau \leq l-1$, if there exists $P \in E[l]^*$ satisfying (6). Since $\phi^2(P) \neq \pm kP$, we can use the rule for adding distinct points to compute an expression for $\phi^2(P) + kP$. Explicitly, let $(P)_x$ denote the x-coordinate of point P. Then for $k \geq 2$

$$(\pm\tau\phi(P))_x = x^q + \frac{f_{\tau-1}^q f_{\tau+1}^q}{f_\tau^{2q}} \tag{9}$$

and

$$(\phi^2(P) + kP)_x = x^{q^2} + x + \frac{f_{k-1}f_{k+1}}{f_k^2} + \lambda^2 + \lambda,$$

where

$$\lambda = \frac{(y^{q^2} + y + x)xf_k^3 + f_{k-2}f_{k+1}^2 + (x^2 + x + y)(f_{k-1}f_kf_{k+1})}{xf_k^3(x + x^{q^2}) + xf_{k-1}f_kf_{k+1}}. \tag{10}$$

Similar equations can be obtained for the case $k = 1$. Equate the x-coordinates of $\phi^2(P) + kP$ and $\pm\tau\phi(P)$, and eliminate denominators and the variable y, to get an identity $h_3(x) = 0$. Then there exists a $P \in E[l]^*$ with $\phi^2(P) + kP = \pm\tau\phi(P)$ if and only if $h_4(x) = \gcd(h_3(x), f_l(x)) \neq 1$. This is repeated for each τ, $1 \leq \tau \leq (l-1)/2$, for which $\tau^2 - 4q$ is a nonresidue $(\bmod\, l)$. If the gcd is nontrivial, then we can determine the correct sign by first equating the y-coordinates of $\phi^2(P) + kP$ and $\tau\phi(P)$. Explicitly, for $\tau \geq 2$,

$$(\tau\phi(P))_y = x^q + y^q + \frac{f_{\tau-1}^q f_{\tau+1}^q}{f_\tau^{2q}} + \frac{f_{\tau-2}^q f_{\tau+1}^{2q}}{x^q f_\tau^{3q}} + (x^{2q} + y^q)\frac{f_{\tau-1}^q f_{\tau+1}^q}{x^q f_\tau^{2q}} \tag{11}$$

and

$$(\phi^2(P) + kP)_y = \lambda(x^{q^2} + x_3) + x_3 + y^{q^2},$$

where $x_3 = (\phi^2(P) + kP)_x$ and λ is as in (10) (similar equations can be obtained for the case $\tau = 1$). As was done above, we then proceed to eliminate the denominator and the variable y to get an identity $h_5(x) = 0$. Then, if $\gcd(f_l(x), h_5(x)) \neq 1$, we have $t = \tau$; otherwise $t = -\tau$. The dominant step in these calculations is the computation of x^{q^2} and y^{q^2} modulo $f_l(x)$.

To determine $t \pmod{l}$ in practice, one would first search for an eigenvalue of ϕ in F_l, and if this fails, then Schoof's algorithm is applied. The first method is faster since it only requires the residue of x^q modulo $f_l(x)$, while the second method requires the residues x^q, x^{q^2}, y^q, and y^{q^2} modulo $f_l(x)$. Heuristically, for a random curve, we would expect ϕ to have an eigenvalue in F_l (i.e., $t^2 - 4q$ is a quadratic residue in F_l) for half of all l's. Moreover, if ϕ does have eigenvlaues in F_l, then in most cases the eigenvalues will be distinct, and so the test if $\phi(P) = wP$ or $\phi(P) = -wP$ in (4.1) takes negligible time (since $\deg g_1(x) = (l-1)/2$ or $l-1$).

4.3. **Determining** t **modulo** $l = 2^c$. If $l = 2^c$, then the following lemma proves that $f_l(x)$ has a factor of small degree.

Lemma 3. *If $l = 2^c$, then $f_l(x)$ has a factor $f(x)$ of degree $l/4$ in $K[x]$.*

Proof. Since $E[l] \cong \mathbb{Z}_l$, $f_l(x)$ has only $l/2$ distinct roots. Of these, only $l/4$ are x-coordinates of points of order l. Thus, $f_l(x)$ has a factor $f(x)$ of degree $l/4$ in $F_q[x]$, whose roots are precisely the x-coordinates of points of order l. □

The next lemma shows how the factor $f(x)$ may be easily constructed.

Lemma 4. *Let $l = 2^c$. Define the sequence of polynomials $\{g_i(x)\}$ in $K[x]$ as follows:*

$$g_0 = x,$$
$$g_1 = b_1 + x, \quad \text{where } a_6 = b_1^4,$$
$$g_i = g_{i-1}^2 + b_i x \prod_{j=1}^{i-2} g_j^2, \quad \text{where } a_6 = b_i^{2^{i+1}}, \text{ for } i \geq 2.$$

Then $f(x) = g_{c-1}(x)$ is a degree $l/4$ factor of $f_l(x)$ in $K[x]$. Moreover, the roots of $f(x)$ are precisely the x-coordinates of points of order l.

Proof. Define the sequence of polynomials $\{h_i(x)\}$ in $K[x]$ by

$$h_0 = 1, \qquad h_1 = x, \qquad h_i = x \prod_{j=1}^{i} g_j^2 \quad \text{for } i \geq 2.$$

Let $P = (x, y) \in E^*$, and let $(2^n P)_x = G_n/H_n$ for $n \geq 0$. From the formula for doubling a point, we see that G_n and H_n are polynomials in $K[x]$. We prove by induction that $G_n = (g_n)^{2^{n+1}}$ and $H_n = (h_n)^{2^n}$ for $n \geq 1$.

For $n = 1$, we have

$$\frac{G_1}{H_1} = \frac{g_1^4}{h_1^2} = \frac{(b_1 + x)^4}{x^2} = \frac{a_6}{x^2} + x^2,$$

which indeed is $(2P)_x$. Assuming that the statement is true for $n = i$, we have

$$(2^{i+1}P)_x = \frac{G_{i+1}}{H_{i+1}} = (2^i P + 2^i P)_x = \frac{a_6 H_i^2}{G_i^2} + \frac{G_i^2}{H_i^2}$$
$$= \frac{(b_1 H_i + G_i)^4}{(G_i H_i)^2} = \frac{(b_{i+1} h_i + g_i^2)^{2^{i+2}}}{(g_i^2 h_i)^{2^{i+1}}} = \frac{(g_{i+1})^{2^{i+2}}}{(h_{i+1})^{2^{i+1}}}.$$

It is also easily proved by induction that $\deg g_n = 2^{n-1}$ for $n \geq 1$, and $\gcd(g_n, h_n) = 1$ for $n \geq 0$.

Now, let $P = (\overline{x}, \overline{y}) \in E^*$. Since $(2^{c-1}P)_x = (g_{c-1})^{2^c}/(h_{c-1})^{2^{c-1}}$, we have $\operatorname{ord}(P) = 2^c$ if and only if $g_{c-1}(\overline{x}) = 0$ and $g_i(\overline{x}) \neq 0$ for $0 \leq i \leq c-2$. But, since $h_{c-1} = g_0 \prod_{j=1}^{c-2} g_j^2$ and $\gcd(g_{c-1}, h_{c-1}) = 1$, we have $\operatorname{ord}(P) = 2^c$ if and only if $g_{c-1}(\overline{x}) = 0$. Finally, since $\deg g_{c-1} = l/4$, the desired factor $f(x)$ must in fact be $g_{c-1}(x)$. □

For $l = 2^c$ that divides q, we have $q \equiv 0 \pmod{l}$. Hence, for $P \in E[l]^*$, we know that $\phi^2(P) - \tau\phi(P) = \mathcal{O}$. Since ϕ is the Frobenius endomorphism,

$\phi(P) \neq \mathcal{O}$ for $P \neq \mathcal{O}$. Therefore, $\phi(P) - \tau P = \mathcal{O}$ and τ is an eigenvalue of ϕ in $\mathbb{Z}_l$.

Since we know that $\#E(F_q) \equiv 0 \pmod 4$, we have that $t \equiv 1 \pmod 4$ and $\tau \equiv 1 \pmod 4$. This gives us only two choices for τ modulo 8. We can easily obtain this eigenvalue using a factor of $f_8(x)$ obtained as above, and using our heuristic for finding eigenvalues. This procedure can then similarly be applied to finding eigenvalues for $l = 16, 32, 64, \ldots$. The method is efficient for l being a small power of 2, since the polynomial arithmetic is performed modulo a degree $l/4$ factor of $f_l(x)$.

4.4. **Baby-step giant-step algorithm.** The calculation of t modulo l using Schoof's algorithm for small primes l is very simple. However, since $\deg(f_l(x)) = (l^2 - 1)/2$, the calculation quickly becomes infeasible as the value of l increases. In [3], the authors combined Schoof's algorithm with Shanks' baby-step giant-step method. In this method, one first computes $\#E(F_q)$ modulo $L = l_0 \cdot l_1 \cdots l_r$, where $l_1, \ldots, l_r$ are small primes and l_0 is a small power of 2. We then use the baby-step giant-step algorithm to determine $\#E(F_q)$.

We describe Shanks' algorithm with suitable modifications for use with Schoof's algorithm.

Step 1. Choose a random point P in $E(F_q)$ and set

$$k = \min\{k' | k' \geq \lceil \sqrt{L \cdot 4 \cdot \sqrt{q}} \rceil,\ k' \equiv 0 \pmod{L}\}.$$

Step 2. Compute iP for $i \equiv (\lfloor q + 1 - 2\sqrt{q} \rfloor - \#E(F_q)) \pmod{L}$ for $0 \leq i \leq k - 1$. If for some i we have $iP = \mathcal{O}$, then return to Step 1. Otherwise, store i and the first 32 bits of the x-coordinate of iP in a table sorted by the entry iP.

Step 3. Set $Q = kP$.

Step 4. Compute

$$H_j = \lfloor q + 1 - 2\sqrt{q} \rfloor P + jQ$$

for $j = 1, 2, \ldots, k/L$ and check (by a binary search) whether the first 32 bits of the x-coordinate of H_j correspond to the first 32 bits of the x-coordinate of iP for some i. If it does, we then check if $H_j = iP$ (by recalculating iP). If we have only one pair (i, j) with $H_j = iP$, then

$$\#E(F_q) = \lfloor q + 1 - 2\sqrt{q} \rfloor + kj - i,$$

and the algorithm terminates. If not, then return to Step 1.

We sketch the correctness and running time of the algorithm.

Since $P \in E(F_q)$, then $\mathrm{ord}(P)$ divides $\#E(F_q)$. Thus, if there exists a unique integer $r \in [q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}]$ such that $rP = \mathcal{O}$, then $r = \#E(F_q)$; if not, then $\mathrm{ord}(P) \leq 4\sqrt{q}$. Either case is detected in Step 4. Thus in Step 1 we hope that $\mathrm{ord}(P) > 4\sqrt{q}$.

Recall that $E(F_q) \cong \mathbb{Z}_{n_1} \oplus \mathbb{Z}_{n_2}$, where $n_1 \mid n_2$ and $n_2 \mid (q - 1)$. For a random elliptic curve, we would expect $n_1 \gg n_2$ and so $n_1 \gg 4\sqrt{q}$. Thus, with very high probability, $\mathrm{ord}(P) > 4\sqrt{q}$. Since $\#E(F_q) \geq (\sqrt{q} - 1)^2$, we have $n_1 \geq \sqrt{q} - 1$. Moreover, since $4 \mid \#E(F_q)$ and n_2 is odd, we have $n_1 \geq 2(\sqrt{q} - 1)$. If in fact $n_1 \leq 4\sqrt{q}$, then there is no point in $E(F_q)$ of order greater than $4\sqrt{q}$. This will be detected since the algorithm will fail in Step 4 each time. If this happens, then we determine $\operatorname{ord}(P)$ and repeat the algorithm until $\operatorname{ord}(P) \geq 2(\sqrt{q}-1)$. We then search for a point P' which has order ≥ 3 in the quotient group $E(F_q)/\langle P\rangle$. For more details, consult [3].

The table in Step 2 has about $S = 2q^{1/4}/\sqrt{L}$ entries, which are computed with $O(S)$ field operations. The table is then sorted using $O(S\log S)$ comparisons. Computing H_j for $j = 1, 2, \ldots, k/L$ takes $O(S)$ field operations, while each binary search takes $O(\log S)$ comparisons. Thus the whole algorithm takes $O(q^{1/4}(\log q)^2/\sqrt{L})$ bit operations, and requires $O(q^{1/4}(\log q)/\sqrt{L})$ bits of storage.

4.5. **Checking results.** Let $\#E(F_q) = q + 1 - t$, where t is unknown, and suppose that our algorithm outputs $\#E(F_q) = q + 1 - t'$. We may verify that $t = t'$ as follows.

Let P be the point in the baby-step giant-step algorithm. Since the algorithm terminated, we believe that $\operatorname{ord}(P) > 4\sqrt{q}$. We first verify that $(q+1-t')P = \mathcal{O}$; if this does not hold, then $t \neq t'$. We then proceed to factor $q+1-t'$, which is an easy task since $q+1-t' \leq 10^{50}$ for the q's we are concerned with. Given the prime factorization of $q+1-t'$, we can easily determine $\operatorname{ord}(P)$, and we then check that $\operatorname{ord}(P) > 4\sqrt{q}$. Now, since $(q+1-t)P = \mathcal{O}$ and $(q+1-t')P = \mathcal{O}$, we deduce that $(t-t')P = \mathcal{O}$. Finally, since $\operatorname{ord}(P) > 4\sqrt{q}$ and $|t-t'| \leq 4\sqrt{q}$, we conclude that $t = t'$.

Of course, this check is only successful if $n_1 > 4\sqrt{q}$, which, as pointed out in §4.4, is true for most curves.

5. Implementation and results

The algorithm described in §4 was implemented in the C programming language on a SUN-2 SPARC station with 64 Mbytes of main memory. We make some comments on our implementation.

(i) The elements of $F_q = F_{2^m}$ were represented with respect to a normal basis. This has the advantage that squaring a field element involves only a cyclic shift of the vector representation. Explicitly, if β is a normal basis generator and $\alpha = \sum_{i=0}^{m-1}\lambda_i\beta^{2^i}$, where $\lambda_i \in F_2$, then $\alpha^2 = \sum_{i=0}^{m-1}\lambda_{i-1}\beta^{2^i}$ (with subscripts reduced modulo m). For computational efficiency in multiplying field elements, we use the special class of normal bases known as optimal normal bases [11]; these bases only exist for certain values of m but are perhaps the most important for practical purposes.

(ii) Let $n = \deg f_l(x)$. To compute $\gcd(A(x), f_l(x))$ for some $A(x) \in K[x]$, we first reduce $A(x)$ modulo $f_l(x)$, and then compute the gcd of the resulting polynomial with $f_l(x)$. In order to compute $x^q \pmod{f_l(x)}$, which is needed, for example, in (7), we precompute the residues x^{2j} modulo $f_l(x)$, for $0 \leq j \leq n-1$. Then $x^q \pmod{f_l(x)}$ is obtained by repeatedly squaring x. Explicitly,

$$\begin{aligned} x^{2^i}(\operatorname{mod} f_l(x)) &= (x^{2^{i-1}}(\operatorname{mod} f_l(x)))^2(\operatorname{mod} f_l(x)) \\ &= \left(\sum_{j=0}^{n-1} a_j x^j\right)^2 (\operatorname{mod} f_l(x)) = \sum_{j=0}^{n-1} a_j^2(x^{2j}(\operatorname{mod} f_l(x))). \end{aligned}$$

The residues of x^{q^2}, y^q, and y^{q^2} modulo $f_l(x)$ are obtained in a similar manner.

(iii) In calculating (9) and (11), we need to compute $f_\tau^q \pmod{f_l(x)}$, for $0 \le \tau \le (l-1)/2+1$. Since we already know $x^q \pmod{f_l(x)}$, we can easily compute $f_\tau^q \pmod{f_l(x)}$ recursively:

$$\begin{aligned}
f_0^q &= 0 \pmod{f_l(x)}, \\
f_1^q &= 1 \pmod{f_l(x)}, \\
f_2^q &= x^q \pmod{f_l(x)}, \\
f_3^q &= x^{4q} + x^{3q} + a_6 \pmod{f_l(x)}, \\
f_4^q &= x^{6q} + a_6 x^{2q} \pmod{f_l(x)}, \\
f_{2i+1}^q &= f_i^{3q} f_{i+2}^q + f_{i-1}^q f_{i+1}^{3q} \pmod{f_l(x)}, \qquad i \ge 2, \\
f_{2i}^q &= s(x)(f_{i-1}^{2q} f_i^q f_{i+2}^q + f_{i-2}^q f_i^q f_{i+2}^{2q}) \pmod{f_l(x)}, \qquad i \ge 3,
\end{aligned}$$

where $s(x) \in K[x]$ satisfies $s(x)x^q \equiv 1 \pmod{f_l(x)}$. Note that indeed

$$\gcd(x^q, f_l(x)) = 1$$

when l is odd, since the only points with x-coordinates equal to 0 have order 2.

(iv) We chose l's up to 31 in order to keep manageable the size of the space searched in the baby-step giant-step part of the method. If more memory is available, then the cases $l = 29$ and $l = 31$ may be excluded, at the expense of an increase in the time for the baby-step giant-step part.

Using the method of §4.3, we also computed t modulo 64. If (t modulo 64) ≤ 31, then we compute t modulo 128 (for this we only need the division polynomials $f_i(x)$, $1 \le i \le 31$, modulo the degree-32 factor of $f_{128}(x)$). Similarly, if (t modulo 128) ≤ 31, we compute t modulo 256. In this way we may compute t modulo 1024.

(v) In the baby-step giant-step algorithm we need to select points uniformly at random from $E(F_q)$. This is accomplished as follows. First pick a random element $\overline{x} \in F_q$. The probablity that $\overline{x}$ is the x-coordinate of some $P \in E(F_q)$ is roughly $\frac{1}{2}$; this follows from Hasse's theorem. We then attempt to solve the equation

$$y^2 + \overline{x}y = \overline{x}^3 + a_6$$

for y. There is a solution if and only if there is a solution to $y^2 + y = b$, where $b = \overline{x}^{-2}(\overline{x}^3 + a_6)$. Compute b, and let

$$b = \sum_{i=0}^{m-1} b_i \beta^{2^i} \quad \text{and} \quad \overline{y} = \sum_{i=0}^{m-1} y_i \beta^{2^i}.$$

Then

$$\overline{y}^2 + \overline{y} = \sum_{i=0}^{m-1} (y_{i-1} + y_i)\beta^{2^i} = \sum_{i=0}^{m-1} b_i \beta^{2^i}.$$

Select $y_0 = 0$ or $y_0 = 1$ at random. Since $y_0 + y_1 = b_1$, this determines y_1. Similarly, $y_2, y_3, \dots, y_{m-1}$ are determined. Finally, if $y_{m-1} + y_0 = b_0$, then $(\overline{x}, \overline{x}\,\overline{y})$ is a random point in $E(F_q)$. Otherwise, $\overline{x}$ is not the x-coordinate of a point in $E(F_q)$.

In Table 1, we list the time taken for the major steps in (4.1), (4.2), and (4.3) of our algorithm for counting points on a single randomly chosen curve over

TABLE 1. Times (in seconds) for the major steps in (4.1), (4.2), and (4.3) of the algorithm for counting points on a single randomly chosen curve over F_q, $q = 2^{155}$

Time to compute $f_i(x)$, $0 \le i \le 31$						245.3				
Time to compute t modulo 128						162.7				
l	3	5	7	11	13	17	19	23	29	31
(4.1) (a)	1.7	9.4	35.6	278.1	469.8	1231.3	2149.8	4612.9	11939.1	14170.2
(b)	0.1	0.7	1.1	31.5	69.8	89.9	458.3	1243.2	778.2	5252.0
(c)	–	–	13.1	–	–	88.3	–	–	72.3	–
(4.2) (d)	1.7	9.7	–	247.7	488.9	–	2268.1	4890.6	–	15188.2
(e)	11.5	–	–	552.6	1026.8	–	4539.4	9525.4	–	28869.2
(f)	3.4	–	–	495.4	977.7	–	4536.3	9805.2	–	30141.0
(g)	0.1	–	–	87.2	299.7	–	2036.9	6072.8	–	22463.9
(h)	0.7	–	–	173.2	177.3	–	2018.3	786.3	–	6298.5
(i)	0.9	–	–	213.0	348.8	–	1831.9	3444.4	–	9971.7

Legend

(a) Compute $x^q \pmod{f_l(x)}$.
(b) Search for an eigenvalue.
(c) Determine the sign of the eigenvalue.
(d) Compute $x^{q^2} \pmod{f_l(x)}$.
(e) Compute $y^q \pmod{f_l(x)}$.
(f) Compute $y^{q^2} \pmod{f_l(x)}$.
(g) Compute $f_i^q \pmod{f_l(x)}$, $0 \le i \le (l-1)/2+1$.
(h) Search for τ, $1 \le \tau \le (l-1)/2$.
(i) Determine the sign of τ.

$F_{2^{155}}$. As was expected, the computation of $x^q \pmod{f_l}$ dominated the time to search for an eigenvalue, while the computation of x^{q^2}, y^q, and y^{q^2} modulo f_l is the dominant step in the Schoof part of the algorithm. If an eigenvalue exists, then determining its sign takes negligible time. Observe that searching for an eigenvalue is a useful heuristic, and results in a big time savings should one exist. Lastly, note that the time taken to compute the division polynomials, and to compute t modulo 128, is also negligible.

In Table 2, we list the time for the baby-step giant-step method (step (4.4)) for various problem instances. The size of the space searched is $4\sqrt{q}/L$, where L is the product of those l's for which t modulo l is known.

TABLE 2. Times for the baby-step giant-step part (step (4.4)) for a curve over F_{2^m}

m	l's used in steps 4.1, 4.2, and 4.3	Size of space searched	Time
33	3, 5, 64	$3.9 \cdot 10^2$	0.2 sec
52	3, 5, 7, 11, 128	$1.8 \cdot 10^3$	0.5 sec
65	3, 5, 7, 11, 13, 64	$2.5 \cdot 10^4$	1 sec
82	3, 5, 7, 11, 13, 17, 64	$5.4 \cdot 10^5$	4 sec
100	3, 5, 7, 11, 13, 17, 64	$2.8 \cdot 10^8$	1 min 43 sec
113	3, 5, 7, 11, 13, 17, 64	$2.5 \cdot 10^{10}$	18 min 31 sec
135	3, 5, 7, 11, 13, 17, 19, 23, 64	$1.2 \cdot 10^{11}$	51 min 22 sec
148	3, 5, 7, 11, 13, 17, 19, 23, 29, 64	$3.6 \cdot 10^{11}$	100 min 42 sec
155	3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 128	$6.7 \cdot 10^{10}$	44 min 11 sec

Table 3. Total time for counting points on randomly chosen curves over F_{2^m}

m	l's for which an eigenvalue of ϕ was found in F_l	Total running time (steps (4.1), (4.2), (4.3) and (4.4))
33	3	1 min 6 sec
52	3, 5, 7	4 min 51 sec
65	5	22 min 29 sec
82	3, 7, 11, 13	57 min 46 sec
100	5, 7, 11, 17	46 min 21 sec
113	3, 7, 17	1 hr 8 min 7 sec
135	3, 7, 13, 19, 23	5 hr 43 min 47 sec
148	5, 7, 11, 13, 17, 19, 29	16 hr 7 min 26 sec
155	7, 17, 29	60 hr 29 min 33 sec

Finally, Table 3 presents the total running time of our method for evaluating $\#E(F_{2^m})$ for single randomly chosen curves and several values of m. For a fixed m, the running time for counting $\#E(F_{2^m})$ has a large variance; the longest running times happen when no eigenvalue of ϕ exists in F_l for the largest prime l's used.

6. A survey of recent work

Let $K = F_q$. As observed in §4, there is a degree $(l-1)/2$ factor $f(x)$ of $f_l(x)$ in $K[x]$ for those primes l for which ϕ has distinct eigenvalues in F_l. If this factor exists and is known, then it may be used instead of $f_l(x)$ in Schoof's algorithm for a considerable savings in time. In unpublished work, Elkies and Miller independently showed how to construct the factor $f(x)$ without having to first construct $f_l(x)$. In [4], Elkies' work is modified, whereby $f(x)$ can be easily computed after some one-time work. These modifications reduce the work for determining $\#E(K)$ from $O(\log^8 q)$ to $O(\log^6 q)$ bit operations. The running of $O(\log^6 q)$ is not rigorously proved since, for example, it is assumed that $t^2 - 4q$ is a quadratic residue modulo l for roughly half of all odd primes l. The method is described only for the case q an odd prime, and the generalization to the case $q = 2^m$ does not appear to be straightforward. We are unaware of any implementations of this method.

Recently Atkin [2] described a new algorithm for computing $\#E(K)$ which uses modular equations. For each odd prime l, the algorithm performs operations in $K[x]$ modulo a polynomial of degree $l+1$ instead of the polynomial $f_l(x)$ of degree $(l^2-1)/2$. Each iteration determines that $t \pmod{l} \in S_l$, where S_l is a subset of $\{0, 1, 2, \ldots, l\}$, and where $|S_l| < l/2$ but usually $|S_l| \ll l/2$. This partial information for various l's is then combined to reveal t. Again, the algorithm has been described only for the case q an odd prime. The algorithm has not been rigorously analyzed but performs remarkably well in practice. It is almost certain to work when $q \approx 10^{50}$, and Atkin has recently computed $\#E(K)$, where q is an odd prime and $q \approx 10^{68}$.

7. Concluding remarks

We have implemented Schoof's algorithm along with some heuristics, and we are able to compute $\#E(F_{2^m})$, where E is any elliptic curve over F_{2^m} and $m \leq 155$. For the Schoof part, we were able to compute t modulo

l for $l = 3, 5, 7, 11, 13, 17, 19, 23, 29, 31$, and 64 (and sometimes $l = 128, 256, 512, 1024$).

Computing $\#E(F_{2^{155}})$ takes roughly 61 hours on a SUN-2 SPARC station. (The algorithm takes 61 hours or less provided that ϕ has an eigenvalue in either F_{29} or F_{31}. Heuristically, one would expect this to occur about 75% of the time for random curves.) On the SPARC station, we can multiply field elements in $F_{2^{155}}$ at the rate of 900 multiplications per second. There exists a special purpose chip which does the field arithmetic in $F_{2^{155}}$ and can perform 250,000 multiplications per second [1]. Since roughly 90% of all time of the algorithm is spent in multiplying field elements in F_{2^m}, the use of this chip should reduce the time for computing $\#E(F_{2^{155}})$ to about 6 hours.

Possible improvements which we did not implement are the computation of t modulo 27, and using Pollard's Lambda method for catching kangaroos [12] instead of the baby-step giant-step algorithm. Pollard's method has the same expected running time as the latter method, but requires very little storage.

Finally, as pointed out by Atkin [2], we mention that the information obtained from Schoof's algorithm and the heuristics presented here can be combined with the information from Atkin's method to compute $\#E(F_{2^m})$ for even larger values of m.

Acknowledgment

We would like to thank an anonymous referee and A. Atkin for their careful reading of this paper, and for suggesting some useful improvements.

Bibliography

1. G. Agnew, R. Mullin, and S. Vanstone, *An implementation of elliptic curve cryptosystems over* $F_{2^{155}}$, preprint.
2. A. Atkin, *The number of points on an elliptic curve modulo a prime*, unpublished manuscript, 1991.
3. J. Buchmann and V. Muller, *Computing the number of points of elliptic curves over finite fields*, presented at International Symposium on Symbolic and Algebraic Computation, Bonn, July 1991.
4. L. Charlap, R. Coley, and D. Robbins, *Enumeration of rational points on elliptic curves over finite fields*, preprint.
5. N. Koblitz, *Elliptic curve cryptosystems*, Math. Comp. **48** (1987), 203–209.
6. ——, *Constructing elliptic curve cryptosystems in characteristic* 2, Advances in Cryptology—Proc. Crypto '90, Lecture Notes in Comput. Sci., vol. 537, Springer-Verlag, Berlin, 1991, pp. 156–167.
7. ——, *CM-curves with good cryptographic properties*, Advances in Cryptology—Proc. Crypto '91, Lecture Notes in Comput. Sci., vol. 576, Springer-Verlag, Berlin, 1992, pp. 279–287.
8. S. Lang, *Elliptic curves: Diophantine analysis*, Springer-Verlag, Berlin, 1978.
9. A. Menezes and S. Vanstone, *Isomorphism classes of elliptic curves over finite fields of characteristic* 2, Utilitas Math. **38** (1990), 135–153.
10. V. Miller, *Uses of elliptic curves in cryptography*, Advances in Cryptology—Proc. Crypto '85, Lecture Notes in Comput. Sci., vol. 218, Springer-Verlag, Berlin, 1986, pp. 417–426.
11. R. Mullin, I. Onyszchuk, S. Vanstone, and R. Wilson, *Optimal normal bases in* $GF(p^n)$, Discrete Appl. Math. **22** (1988/89), 149–161.

12. J. Pollard, *Monte Carlo methods for index computation* (mod p), Math. Comp. **32** (1978), 918–924.

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

Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

E-mail address: ajmeneze@math.waterloo.edu

E-mail address: savansto@math.waterloo.edu

E-mail address: rjzucche@math.waterloo.edu