

Point Counting Algorithms - A Survey

Heekwan Lee and Iftikhar Burhanuddin
{heekwanl | burhanud}@usc.edu

May 4, 2002

Abstract

The purpose of this paper is to survey the algorithms that compute the size of the elliptic curve group over finite fields. The techniques we describe range from naive ones which have exponential time complexity to sophisticated ones which run in polynomial time. We also present some computational results on the naive algorithms.

1 Introduction

Let E be an elliptic curve defined over a finite field \mathbb{F}_q with q elements and given by the Weierstraß equation

$$Y^2 = X^3 + aX + b$$

where $a, b \in \mathbb{F}_q$. We'll restrict our discussion to $\text{char } \mathbb{F}_q > 3$, unless otherwise stated. Let $E(\mathbb{F}_q)$ be the set of \mathbb{F}_q -rational points and O be the point at infinity, then,

$$E(\mathbb{F}_q) = \{(x, y) \mid x, y \in \mathbb{F}_q\} \cup \{O\}$$

Determining the cardinality of the aforementioned set is called the *Point Counting Problem* and we'll describe various algorithms which compute $\#E(\mathbb{F}_q)$ in this paper. The operation of point counting is an important one in elliptic curve cryptography as it is a yard stick to test whether a given curve is suitable for cryptography. In sections 2 and 3, we discuss brute force algorithms which naturally run in exponential time. Section 4 describes Schoof's method, which is the first polynomial time algorithm for this problem, which is not efficient in practice. In section 5, we talk about Satoh's algorithm which runs in time *cubic* in the length of input and is the best known as far as time and space complexity are concerned.

2 Naive Algorithm

Say point $P = (x, y) \in E(\mathbb{F}_p)$, where p is a prime. As x ranges over the elements in \mathbb{F}_p , the number of solutions to the equation $Y^2 = x^3 + ax + b$ in \mathbb{F}_p are either

2, 1 or 0 depending on whether i) $x^3 + ax + b$ is a quadratic residue in \mathbb{F}_p , ii) $x^3 + ax + b$ is divisible by p , in which case P is a 2-torsion point or iii) $x^3 + ax + b$ is a quadratic non-residue in \mathbb{F}_p . Adding 1 to account for the point at infinity O , we have

$$\begin{aligned} \#E(\mathbb{F}_p) &= 1 + \sum_{x \in \mathbb{F}_p} \left(1 + \left(\frac{x^3 + ax + b}{p}\right)\right) \\ &= 1 + p + \sum_{x \in \mathbb{F}_p} \left(\frac{x^3 + ax + b}{p}\right) \end{aligned}$$

We implemented and ran this algorithm using the PARI-GP software package on a Pentium III, 700 MHz, 128 MB computer and experimented with prime finite fields with $k \cdot 2^n + 1$ elements. The results tabulated in *Appendix I* indicate about a 10-fold increase in running time for each digit increase in the size of the field. This is an $O(p^{1+\epsilon})$ method, which is clearly exponential in $\log p$ and can be used for $p < 10000$ according to [6] which seems to be in line with our results.

3 Baby Step Giant Step Algorithm

The Baby Step Giant Step Algorithm is a generic technique which can be applied to finite abelian groups to determine group order, compute discrete log, etc [6]. If the size of search space is N , this method answers the question in $O(\sqrt{N})$ time. According to Hasse's theorem $\#E(\mathbb{F}_q)$ can be off by $O(\sqrt{q})$ and hence we can compute $\#E(\mathbb{F}_q)$ using this algorithm in $O(q^{\frac{1}{4}+\epsilon})$ time and space.

Theorem 3.1 (Hasse, 1933) *Let E be an elliptic curve defined over a finite field \mathbf{F}_q with q elements and given by the Weierstraß equation*

$$Y^2 = X^3 + aX + b$$

where $a, b \in \mathbf{F}_q$ then

$$\begin{aligned} \#E(\mathbf{F}_q) &= q + 1 - t \\ |t| &\leq 2\sqrt{q} \end{aligned}$$

Say $P \in E(\mathbb{F}_q)$ and $\#E(\mathbb{F}_q) = m$. Using the division algorithm we can write m as follows,

$$m = ig + j$$

where $0 \leq i < g, 0 \leq j < b$, for some g and b . Now as m is the order of the group the point P gets killed by m , and we have

$$\begin{aligned} O = [m]P &= [ig]P + [j]P \\ -[ig]P &= [j]P \quad (*) \end{aligned}$$

The term *Baby Step* refers to the computing the RHS of (*) and *Giant Step* refers to the LHS. Minimizing the total number of baby and giant steps, that is $b + g$ subject to $b.g = m$, gives us $b = g = \lceil 2q^{\frac{1}{4}} \rceil$

Baby Step Giant Step Algorithm, 1971 [11]

INPUT: An elliptic curve E over a finite field \mathbb{F}_q

OUTPUT: The order of $E(\mathbb{F}_q)$

1. Select a random point $P \in E(\mathbb{F}_q)$ and compute $[j]P$,
for $j = 0$ to $b - 1$ and store them in some *convenient* manner
 2. $i \leftarrow 0$
 3. Compute $[q + 1 + 2\sqrt{q} - ig]P$ and check if it is in the table
 4. If the i th giant step is equal to the j th baby step,
then $t \leftarrow ig + j - 2\sqrt{q}$. Otherwise increment i and goto step 3
-

Schoof [1] suggests hashing as a *convenient* manner to store the points in the baby steps to aid efficient searching. The t obtained is the group order if the order of P was greater than $4\sqrt{q}$ otherwise there exist integers t_1, t_2 , such that $[t_1]P = [t_2]P = O$. By a theorem due to Mestre [1] a curve or it's quadratic twist allows a point of order greater than $4\sqrt{p}$, when $p > 461$. Hence appropriately using the curve or it's twist eliminates the *point of small order* problem.

We used the *ellap* function call of the PARI-GP software package which implements the BSGS algorithm with Mestre's enhancement to compute the size of a randomly generated elliptic curve group over prime finite fields with $k.2^n + 1$ elements. The computation was performed on a Pentium III, 700 MHz, 128 MB computer and the results are tabulated in *Appendix II*. The PARI-GP user's manual [13] suggests that this method can be used for $p < 10^{30}$. Another algorithm which runs in the same asymptotic time is Pollard's ρ method but has the advantage of having practically no memory requirements [5].

4 Schoof's Algorithm

Let E be an elliptic curve defined over a finite field \mathbb{F}_q of char $p > 3$ with q elements. The number of \mathbb{F}_q rational points on E is denoted by $\#E(\mathbb{F}_q)$

Consider the q -th power Frobenius isogeny acting on E . Let E^q denote the equation defining curve E with it's coefficients a, b raised to the q th power, but since $a, b \in \mathbb{F}_q$, we get $E^q = E$ and the Frobenius isogeny is an endomorphism from E to itself.

$$\begin{array}{ccc} \phi : E(\overline{\mathbb{F}}_q) & \rightarrow & E(\overline{\mathbb{F}}_q) \\ (x, y) & \mapsto & (x^q, y^q) \\ O & \mapsto & O \end{array}$$

Observe that the action of Frobenius on the \mathbb{F}_q rational points is trivial. Say

$\hat{\phi}$ denotes the dual isogeny of ϕ such that $\phi \circ \hat{\phi} = [q]$,

$$\begin{array}{ccc} \hat{\phi} : E(\overline{\mathbb{F}}_q) & \rightarrow & E(\overline{\mathbb{F}}_q) \\ O & \mapsto & O \end{array}$$

The *trace* of the Frobenius isogeny is defined as the number t such that $\phi + \hat{\phi} = [t]$, that is, for $P \in E(\overline{\mathbb{F}}_q)$

$$\phi(P) - [t](P) + \hat{\phi}(P) = O$$

Also the trace satisfies the equation $\#E(\mathbb{F}_q) = q + 1 - t$. Applying ϕ to both sides and using the fact that $\phi \circ \hat{\phi} = [q]$, we obtain

$$\phi^2(P) - [t]\phi(P) + [q](P) = O$$

A naive approach to compute t may be to select a *suitable* $P = (x, y) \in E(\overline{\mathbb{F}}_q)$ and plug into the above equation and determine t . Say $\phi^2(P) + [q](P) = Q$ and $\phi(P) = R$ then the equation reduces to $Q = [t]R$ and since t is quite big an efficient point counting algorithm using this approach is possible only if the elliptic curve discrete logarithm problem has an efficient algorithm.

Say $P \in E[l]$, is an l -torsion point, then

$$\phi^2(P) - [t_l]\phi(P) + [q_l](P) = O,$$

where $t_l \equiv t \pmod{l}$, $q_l \equiv q \pmod{l}$. Hasse [7] tells us that $|t| \leq 2\sqrt{q}$. So if we find $t \pmod{l}$ for sufficiently many - $O(\log q / \log \log q)$ primes l then using CRT we can recover t . So working with torsion points seems to be a better idea.

$$\begin{aligned} \phi^2(P) + [q_l](P) &= [\tau]\phi(P) \\ \Rightarrow (x^q, y^q) + [q_l](x, y) &= [\tau](x^q, y^q) \\ \text{where } \tau &\in \{0, 1, \dots, l-1\} \end{aligned}$$

Schoof's algorithm, 1985 [11], [2]

INPUT: An elliptic curve E over a finite field \mathbb{F}_q

OUTPUT: The order of $E(\mathbb{F}_q)$

1. $n \leftarrow 2$ and $l_0 \leftarrow 2$
 2. Find the smallest prime l_n greater than l_{n-1} and different from p
 3. Compute $t_n \equiv t \pmod{l_n}$
 4. If $\prod_{i=1}^n l_i < 4\sqrt{q}$; increment n , goto step 2
 5. Solve the system of congruences $t \equiv t_i \pmod{l_i}$ for $1 \leq i \leq n$
 6. $\#E(\mathbb{F}_q) \leftarrow q + 1 - t$
-

So can we efficiently find τ by trial and error? Working explicitly with the l -torsion points we seem to run into the discrete logarithm problem. Schoof's

idea was instead to work with the group law. By repeated application of the group law, we can express the multiplication by m map by rational functions and work with l -division polynomials instead due to the following theorems.

Theorem 4.1 ([11]) *Let E be an elliptic curve defined over a field K , and let m be a positive integer. There exist polynomials $\psi_m, \theta_m, \omega_m \in K[X, Y]$ such that, for $P = (x, y) \in E(\overline{K})$ such that $[m]P \neq O$, we have*

$$[m]P = \left(\frac{\theta_m(x, y)}{\psi_m(x, y)^2}, \frac{\omega_m(x, y)}{\psi_m(x, y)^3} \right)$$

The polynomial $\psi_m(x, y)$ is called the m th division polynomial of the curve E . Also the sequences θ_m and ω_m can be expressed in terms of the sequence ψ_m

Theorem 4.2 ([1]) *Let $P \in E(\overline{K}) - \{O\}$, and let $m \geq 1$. $P \in E[m] \Leftrightarrow \psi_m(P) = 0$*

So for $P = (x, y) \in E[l]$

$$\begin{aligned} \phi^2(P) + [q_l](P) &= [\tau]\phi(P) \\ &\Downarrow \\ (X^{q^2}, Y^{q^2}) + [q_l](X, Y) &\equiv [\tau](X^q, Y^q) \end{aligned}$$

in the ring $\mathbb{F}_q[X, Y] / \langle \psi_l(X), Y^2 - X^3 - AX - B \rangle$

Running time of the algorithm is $O(\log^8 q)$ with naive arithmetic. The l -division polynomials have $O(l^2)$ degree and working with them turns out to be expensive. With $p \approx 10^{200}$, representing one element in the the ring requires 1.5 MB [1]. Elkies' and Atkins' suggestions resulted in SEA algorithm [1]. It uses l -division polynomials of degree $O(l)$ found using modular polynomials and runs in $O(\log^{4+\epsilon} q)$

Records using Schoof's algorithm and it's variants are Morain's point counting result in 1995 with $q = 10^{499} + 153$. Vercauteren in 1999 was able to compute the size of the elliptic curve group over a finite field with 2^{1999} elements [14].

5 Satoh's Algorithm

Definitions [14] Let π_n be the projection from $\mathbb{Z}/p^{n+1}\mathbb{Z}$ onto $\mathbb{Z}/p^n\mathbb{Z}$. A p -adic integer is a sequence $x = (x_1, x_2, \dots, x_n, \dots)$ with $x_n \in \mathbb{Z}/p^n\mathbb{Z}$ and such that $\pi_n(x_{n+1}) = x_n$ for $n \geq 1$. The ring of p -adic integers is denoted by \mathbb{Z}_p

Let $q = p^n$ with p prime. Let $f(t)$ be a monic polynomial in $\mathbb{Z}_p[t]$ of degree n such that the polynomial $\pi(f)$ obtained by projecting the coefficients is irreducible in $\mathbb{F}_p[t]$. The ring \mathbb{Z}_q is $\mathbb{Z}_p[t]$ modulo (the ideal generated by) $f(t)$.

The Frobenius endomorphism F (we defined in the previous section as ϕ) is difficult to lift because it is inseparable. Therefore one works with the dual of

the Frobenius endomorphism F , called the Verschiebung (*shift* in German) \hat{F} , which is separable if and only if E is non-supersingular.

Lubin-Serre-Tate theorem [16] tells us that there is a canonical way to lift the curve E to \mathbb{Z}_q by lifting its j -invariant. We'll shall see this lift "preserves" the trace of the Frobenius and also $\mathcal{E} \bmod p$ equals E .

Theorem 5.1 (Lubin-Serre-Tate, 1964) *Let E be a non-supersingular elliptic curve over \mathbb{F}_q with j -invariant $j(E) \in \mathbb{F}_q - \mathbb{F}_{p^2}$. Denote with Σ the Frobenius substitution on \mathbb{Z}_q and with $\phi_p(X, Y)$ the p -th modular polynomial. Then the system of equations*

$$\phi_p(X, \Sigma(X)) = 0 \text{ and } X \equiv j(E) \bmod p \quad (*),$$

has a unique solution $J \in \mathbb{Z}_q$, which is the j -invariant of the canonical lift \mathcal{E} of E .

Explicit computation of Σ is rather slow in practice [14] and hence solving the system of equations (*) directly would lead to a slow algorithm. Satoh's algorithm [8] choses another route and avoids computing Σ . Satoh's idea was to view the Frobenius endomorphism as a cycle of isogenies and instead of lifting j to J , was to lift j along with all its conjugates j_i simultaneously to appropriate p -adic precision.

Let $\sigma : E \rightarrow E^\sigma : (x, y) \mapsto (x^p, y^p)$ be the p -th power Frobenius map, where E^σ is the curve obtained by raising each coefficient to the p -th power and let $\hat{\sigma}$ be the dual of σ . Repeatedly applying $\hat{\sigma}$ gives rise to the following cycle

$$E_0 \xrightarrow{\hat{\sigma}_0} E_1 \xrightarrow{\hat{\sigma}_1} \dots \xrightarrow{\hat{\sigma}_{n-2}} E_{n-1} \xrightarrow{\hat{\sigma}_{n-1}} E_0$$

Let $\hat{\sigma}_i$ the dual of $\sigma_i : E_{i+1} \rightarrow E_i : (x, y) \mapsto (x^p, y^p)$ and $E_n = E_0$. Composing these, we see that $\hat{F} = \hat{\sigma}_{n-1} \circ \hat{\sigma}_{n-2} \circ \dots \circ \hat{\sigma}_1 \circ \hat{\sigma}_0$. Satoh's idea instead of lifting E and \hat{F} directly was to lift the whole cycle $(E_0, E_0, \dots, E_{n-1})$ simultaneously leading to the diagram

$$\begin{array}{cccccccc} \mathcal{E}_0 & \xrightarrow{\hat{\Sigma}_0} & \mathcal{E}_1 & \xrightarrow{\hat{\Sigma}_1} & \dots & \xrightarrow{\hat{\Sigma}_{n-2}} & \mathcal{E}_{n-1} & \xrightarrow{\hat{\Sigma}_{n-1}} & \mathcal{E}_0 \\ \uparrow & & \uparrow & & & & \uparrow & & \uparrow \\ E_0 & \xrightarrow{\hat{\sigma}_0} & E_1 & \xrightarrow{\hat{\sigma}_1} & \dots & \xrightarrow{\hat{\sigma}_{n-2}} & E_{n-1} & \xrightarrow{\hat{\sigma}_{n-1}} & E_0 \end{array}$$

with \mathcal{E}_i the canonical lift of E_i and $\hat{\Sigma}_i$ the corresponding lift of $\hat{\sigma}_i$. The theorem of Lubin, Serre and Tate implies that the j -invariants of \mathcal{E}_i satisfy

$$\phi_p(j(\mathcal{E}_i), j(\mathcal{E}_{i+1})) = 0 \text{ and } j(\mathcal{E}_i) \equiv j(E_i) \bmod p,$$

for $i = 0, 1, \dots, n-1$ and $\mathcal{E}_n = \mathcal{E}_0$. Using Hensel's lifting on the modular polynomial relations, we can compute $J_i \equiv j(\mathcal{E}_i) \bmod p^N$ in $\log N$ iterations.

The canonical lift of \mathcal{E} of a non-supersingular elliptic curve E over \mathbb{F}_q has the property that $\text{End}(E) \cong \text{End}(\mathcal{E})$ as a ring. Therefore we have $\text{Tr}(F) =$

$\text{Tr}(\mathcal{F})$. Furthermore, the trace of an endomorphism equals the trace of its dual, so $\text{Tr}(F) = \text{Tr}(\hat{F}) = \text{Tr}(\mathcal{F}) = \text{Tr}(\hat{\mathcal{F}})$.

Next we calculate the trace assuming that we have computed \mathcal{E} to appropriate precision. By a theorem of Satoh's which analyzes the action of \mathcal{F} on the formal group of \mathcal{E} , we have $\text{Tr}(\hat{\mathcal{F}}) = c + \frac{q}{c}$ with $\hat{\mathcal{F}}(\tau) = c\tau + O(\tau^2)$, where τ is the local parameter of \mathcal{E} at O .

We know that

$$\text{Tr}(\hat{\mathcal{F}}) = \text{Tr}(\hat{\Sigma}_{n-1} \circ \hat{\Sigma}_{n-2} \circ \dots \circ \hat{\Sigma}_1 \circ \hat{\Sigma}_0)$$

Therefore we can compute c as the product of the leading coefficients of the factors in the formal groups.

More precisely, let c_i be defined by $\tau_{i+1} \circ \hat{\Sigma}_i = c_i \tau_i + O(\tau_i^2)$, with τ_i the local parameter of \mathcal{E}_i at O , then $c = \prod_{0 \leq i < n} c_i$. Since \mathcal{F} is separable, c will be non-zero modulo p and we conclude

$$\text{Tr}(F) \equiv \prod_{0 \leq i < n} c_i \pmod{q}$$

Vélu's formulae [14, 10] allow us to compute the coefficients c_i , based on the equations for \mathcal{E}_i and \mathcal{E}_{i+1} and the kernel of $\hat{\Sigma}_i$. The equations \mathcal{E}_i and \mathcal{E}_{i+1} can be easily computed via a univariate Newton iteration, since we already know their j -invariants. The isogenies $\hat{\sigma}_i$ and $\hat{\Sigma}_i$ are separable and of degree p , so $\hat{\sigma}_i$ can be explicitly lifted to $\hat{\Sigma}_i$ by lifting its kernel, which is a subgroup of the p -torsion group of E .

Satoh's algorithm, 2000 [14]

INPUT: An elliptic curve E over a finite field \mathbb{F}_q , with $j(E) \notin \mathbb{F}_{p^2}$

OUTPUT: The order of $E(\mathbb{F}_q)$

Read *Lift* as lift to appropriate precision

1. Compute the cycle of d curves E_i and their j -invariants j_i
 2. Lift all the j_i simultaneously, yielding J_i
 3. Lift each curve by lifting its coefficients
 4. Lift each curve's p -torsion subgroup
 5. Compute the trace t from the lifted data using Vélu's formula
 6. $\#E(\mathbb{F}_q) \leftarrow q + 1 - t$
-

For the field \mathbb{F}_q , where $q = p^n$ we work with \mathbb{Z}_q with $O(p^{O(n)})$ precision. So each "coefficient" takes $O(n \log p)$ and each element of \mathbb{Z}_q takes $O(n^2 \log p)$. Working with *bounded* number of elements for each of the n conjugate curve, the total memory is $O(n^3 \log p)$ [14]. Therefore Satoh's algorithm computes the order of E for fixed p in $O(n^{3+\varepsilon})$ time and requires $O(n^3)$ of memory for $p \geq 5$. Skjærnaa [9] and Fouquet, Gaudry and Harley [14] independently extended Satoh's algorithm to characteristic 2 and 2, 3 respectively. Vercauteren et al [15] refined the algorithm to consume $O(n^2)$ of memory and run in the same asymptotic time.

A current record using Satoh's algorithm for odd characteristic is point counting on curve with $q = 5^{569}$. Also with FGH's extension to even characteristic computing the size of the elliptic curve group over a finite field with 2^{8009} elements was accomplished.

6 Conclusions and Future Directions

Point counting is an important operation in Elliptic Curve Cryptography and as always there is a need for more efficient algorithms with respect to computation and memory. Lower bounds for the point counting problem will come in handy to indicate where will stand and hence proving lower bounds is one possible direction of research. With the recent advances point counting in char 2 is almost as fast as an RSA key generation [4] and ECC seems to be a viable and infact superior alternative when compared to RSA in resource constrained environments like embedded applications and wireless devices provided ECDLP has no sub exponential time algorithm.

References

- [1] Rene Schoof. *Counting Points On Elliptic Curves Over Finite Fields*. Journal de Theorie des Nombres de Bordeaux 7, 1995
- [2] Kristian Gjøsteen. *Schoof's algorithm*. Preprint, 2000
- [3] Antonia W. Bluer. *A Leisurely Introduction to Formal Groups and Elliptic Curves*. Preprint
- [4] Pierrick Gaudry. *Algorithms for counting points on curves*. Presentation, ECC 2001
- [5] Andreas Enge. *Elliptic Curves and Their Applications to Cryptography - An Introduction*. Kluwer Academic Publishers, 1999
- [6] Henri Cohen. *A Course in Computational Algebraic Number Theory*. Graduate Texts in Mathematics, Springer-Verlag, 1993
- [7] Helmut Hasse. *Beweis des Analogons der Riemannschen Vermutung für die Artinschen und F. K. Smidtschen Kongruenzetafunktionen in gewissen elliptischen Fällen*. Ges. d. Wiss. Narichten. Math.-Phys. Klasse, 1933
- [8] Takakazu Satoh. *The Canonical Lift of an Ordinary Elliptic Curve over a Finite Field and its Point Counting*. Journal of the Ramanujan Mathematical Society, vol. 15, 2000
- [9] Berit Skjernaas. *Satoh's algorithm in Characteristic 2*. Preprint, 2000
- [10] J.Vélu. *Isogénies entre courbes elliptiques*. Comptes rendus l'Académie des sciences de Paris, 273, Série A, 1971

- [11] Ian F. Blake, Gadiel Seroussi, and Nigel P. Smart. *Elliptic Curves in Cryptography*. Cambridge University Press, Cambridge, 1999
- [12] Joeseoph H. Silverman. *The Arithmetic of Elliptic Curves*. Graduate Texts in Mathematics, Springer-Verlag, 1986
- [13] Henri Cohen et al. *User's Guide to PARI/GP*. 2000
- [14] Mireille Fouquet, Pierrick Gaudry and Robert Harley. *An extension of Satoh's algorithm and its implementation*. J. Ramanujan Math. Soc. 15, 2000
- [15] Frederik Vercauteren, Bart Preneel, Joos Vandewalle. *A Memory Efficient Version of Satoh's Algorithm*. Advances in Cryptology - Eurocrypt 2001, Lecture Notes in Computer Science 2045, Springer, 2001
- [16] J. Lubin, Jean-Pierre Serre and John T. Tate. *Elliptic curves and formal groups*. Lecture notes prepared in connection with the seminars held at the Summer Institute on Algebraic Geometry, Whitney Estate, Woods Hole, Massachusetts, July 6-July 31, 1964

7 Appendix I

$y^2 = x^3 + ax + b$	$char \mathbb{F}_p$ [# Digits]	$\#E(\mathbb{F}_p)$	Time
$y^2 = x^3 + 3x + 1$	$3 * 2^1 + 1 = 7$ [1]	12	10 ms
$y^2 = x^3 + 33x + 60$	$9 * 2^3 + 1 = 73$ [2]	69	10 ms
$y^2 = x^3 + 261x + 332$	$7 * 2^6 + 1 = 449$ [3]	420	20 ms
$y^2 = x^3 + 6186x + 3381$	$31 * 2^8 + 1 = 7937$ [4]	8040	250 ms
$y^2 = x^3 + 51914x + 20880$	$1 * 2^{16} + 1 = 65537$ [5]	65280	2, 103 ms
$y^2 = x^3 + 288149x + 134680$	$3 * 2^{18} + 1 = 786433$ [6]	786205	27, 479 ms
$y^2 = x^3 + 4078904x + 2662894$	$11 * 2^{19} + 1 = 5767169$ [7]	5765985	3 mn, 20, 878 ms
$y^2 = x^3 + 2903151x + 19802935$	$11 * 2^{21} + 1 = 23068673$ [8]	23060676	13 mn, 35, 462 ms
$y^2 = x^3 + 13523297x + 33527882$	$5 * 2^{25} + 1 = 167772161$ [9]	167776175	1 h, 37 mn, 12, 817 ms
$y^2 = x^3 + 439167662x + 2995643896$	$13 * 2^{28} + 1 = 3489660929$ [10]	3489664260	> 11 h, 15 mn

8 Appendix II

$y^2 = x^3 + ax + b$	$char \mathbb{F}_p$ [# Digits]	$\#E(\mathbb{F}_p)$	Time
$y^2 = x^3 + 3x + 1$	$3 * 2^1 + 1 = 7$ [1]	12	0 – 30 ms
$y^2 = x^3 + 33x + 60$	$9 * 2^3 + 1 = 73$ [2]	69	0 – 30 ms
$y^2 = x^3 + 261x + 332$	$7 * 2^6 + 1 = 449$ [3]	420	0 – 30 ms
$y^2 = x^3 + 6186x + 3381$	$31 * 2^8 + 1 = 7937$ [4]	8040	0 – 30 ms
$y^2 = x^3 + 51914x + 20880$	$1 * 2^{16} + 1 = 65537$ [5]	65280	0 – 30 ms
$y^2 = x^3 + 288149x + 134680$	$3 * 2^{18} + 1 = 786433$ [6]	786205	0 – 30 ms
$y^2 = x^3 + 4078904x + 2662894$	$11 * 2^{19} + 1 = 5767169$ [7]	5765985	0 – 30 ms
$y^2 = x^3 + 2903151x + 19802935$	$11 * 2^{21} + 1 = 23068673$ [8]	23060676	0 – 30 ms
$y^2 = x^3 + 13523297x + 33527882$	$5 * 2^{25} + 1 = 167772161$ [9]	167776175	0 – 30 ms

$y^2 = x^3 + 46257640654x + 380391811$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 49 * 2^{30} + 1 = 52613349377$ [11] $\#E(\mathbb{F}_p) = 52613138470$ $\text{Time} = \mathbf{10 \text{ ms}}$
$y^2 = x^3 + 173075890854290x + 15340676211085$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 35 * 2^{45} + 1 = 1231453023109121$ [16] $\#E(\mathbb{F}_p) = 1231453038889714$ $\text{Time} = \mathbf{130 \text{ ms}}$
$y^2 = x^3 + 19997399169037516x + 25109674361472332$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 17 * 2^{51} + 1 = 38280596832649217$ [17] $\#E(\mathbb{F}_p) = 38280596799790406$ $\text{Time} = \mathbf{251 \text{ ms}}$
$y^2 = x^3 + 3310642384618942998x + 2216029861822881760$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 29 * 2^{57} + 1 = 4179340454199820289$ [19] $\#E(\mathbb{F}_p) = 4179340454698527751$ $\text{Time} = \mathbf{1,041 \text{ ms}}$
$y^2 = x^3 + 105468761489682535187x + 40246170019083146395$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 9 * 2^{65} + 1 = 332041393326771929089$ [21] $\#E(\mathbb{F}_p) = 332041393291195225233$ $\text{Time} = \mathbf{8,041 \text{ ms}}$

$y^2 = x^3 + 36475090356678594228271x + 3888898517015898167970$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 39 * 2^{70} + 1 = 46043073207979040833537$ [23] $\#E(\mathbb{F}_p) = 46043073207990880985350$ $\text{Time} = \mathbf{10,976 \text{ ms}}$
$y^2 = x^3 + 112471453303800447633864x + 90963216517616084604445$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 5 * 2^{75} + 1 = 188894659314785808547841$ [24] $\#E(\mathbb{F}_p) = 188894659314576663574008$ $\text{Time} = \mathbf{21,511 \text{ ms}}$
$y^2 = x^3 + 1120655742151729212438458x + 2083002591035693767455878$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 15 * 2^{78} + 1 = 4533471823554859405148161$ [25] $\#E(\mathbb{F}_p) = 4533471823556601964187121$ $\text{Time} = \mathbf{1 \text{ mn}, 3,861 \text{ ms}}$
$y^2 = x^3 + 153540579541521345473887025x + 171108961758054635503737369$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 5 * 2^{85} + 1 = 193428131138340667952988161 =$ [27] $\#E(\mathbb{F}_p) = 193428131138339230512213786$ $\text{Time} = \mathbf{1 \text{ mn}, 50,830 \text{ ms}}$
$y^2 = x^3 + 94010262021911566563859251729x + 83424307937653472480453887283$ $\text{char } \mathbb{F}_p [\# \text{ Digits}] = 27 * 2^{92} + 1 = 133697524242821069689105416193 =$ [30] $\#E(\mathbb{F}_p) = \mathbf{\text{not enough memory}}$