Efficient Implementation of Schoof's Algorithm

Tetsuya Izu, Jun Kogure, Masayuki Noro, and Kazuhiro Yokoyama

FUJITSU LABORATORIES LTD., 4-1-1 Kamikodanaka

Nakahara-ku Kawasaki 211-8588, Japan

{izu, kogure}@flab.fujitsu.co.jp, {noro, yokoyama}@para.flab.fujitsu.co.jp

Abstract. Schoof's algorithm is used to find a secure elliptic curve for cryptosystems, as it can compute the number of rational points on a randomly selected elliptic curve defined over a finite field. By realizing efficient combination of several improvements, such as Atkin-Elkies's method, the isogeny cycles method, and trial search by match-and-sort techniques, we can count the number of rational points on an elliptic curve over $GF(p)$ in a reasonable time, where p is a prime whose size is around 240-bits.

1 Introduction

When we use the elliptic curve cryptosystem [9,17] (ECC for short), we first have to define an elliptic curve over a finite field. Then, all cryptographic operations will be performed on the group of rational points on the curve. Since all the curves are not necessarily secure, we should be very careful when we choose an elliptic curve for ECC. There are several methods to select a curve for ECC, such as Schoof's method [22], CM(Complex Multiplication) method [2,18,10,3], and so on. The security of an elliptic curve for ECC depends mainly on the "cardinality" of the curve (the number of rational points on the curve or the order of the group of rational points on the curve). For example, the cardinality should have a large prime factor to guard against Pohlig-Hellman's attack. Recent studies [16,27,21] on special curves, supersingular curves and anomalous curves, suggest that there might exist certain efficient attacks on those curves using their speciality. From this point of view, Schoof's method is believed best to obtain most secure elliptic curves for ECC, as it can compute the cardinality of a randomly selected curve. Moreover, there are heuristic but strong evidence that random search on curves can find certainly good ones with prime cardinality. (See [11].)

Schoof's algorithm was not efficient in its original form. Thanks to the contributions of many people, such as Atkin[1], Elkies[7], Morain[19], Couveignes[6], Lercier[14], and so on, the algorithm became remarkably faster. In the process of computing the cardinality of a given curve, we can combine several improvements on Schoof's algorithm, such as Atkin-Elkies' method, Lercier's method, and the isogeny cycles method. However, as far as the authors know, there were no explicit criteria that give a good combination of these improvements as sub-procedures.

The purpose of this paper is to develop an explicit criterion by introducing several new strategies. In more detail, we introduce an “intelligent choice system” on several subprocedures based on estimate of the costs of them, and investigate its effects by practical experiments. By our implementation, one can count the cardinality of one curve for ECC corresponding to the RSA with 1024bit moduli within about one minutes in average on a PC with Pentium II 300MHz CPU. And the average time for counting the cardinality suggests that its complexity is $O(\log(p)^{5\pm\epsilon})$ for some $\epsilon \ll 1$. Thus, incorporated with the “early abort” strategy [13], Schoof's method is recommended, as a practical one, to find good secure curves.

In the following sections, we consider curves over fields of odd characteristic. The proposed strategy/methods, however, can be applied *independent* of the characteristics of the base field. In section 2, we will briefly look over the improvements of Atkin-Elkies. In section 3, we will introduce new improvements and show an explicit criterion. In section 4, we will give our experimental results to show our improvements are actually efficient.

2 Overview of Previous Works

Let p be an odd prime. We will consider an elliptic curve E defined over the finite field $GF(p)$ of order p : $E : y^2 = x^3 + Ax + B$, where $A, B \in GF(p)$ with $4A^3 + 27B^2 \not\equiv 0 \pmod{p}$. For the mathematical background, see [15,25,26].

2.1 Schoof's Algorithm

First we will briefly recall the Schoof's algorithm [22]. We denote the subgroup of ℓ -torsion points of E by $E[\ell]$ for a prime ℓ . The *Frobenius endomorphism* $\phi : (x, y) \rightarrow (x^p, y^p)$ of E is also defined on Tate module $T_\ell(E)$ as a linear map and satisfies the equation: $\phi^2 - t\phi + p = 0$, where t is the *trace* of the Frobenius map which does not depend on ℓ . Then

$$\#E(GF(p)) = p + 1 - t. \quad (1)$$

If we find an integer t_ℓ such that

$$\phi^2(P) + pP = t_\ell \phi(P) \quad (2)$$

for any $P \in E[\ell]$, we get $t \equiv t_\ell \pmod{\ell}$. By Hasse's theorem, t must satisfy $-2\sqrt{p} \leq t \leq 2\sqrt{p}$. Therefore if we compute $t \bmod \ell$ for various small primes until their product exceeds $4\sqrt{p}$, we can uniquely determine the cardinality of the curve by means of the Chinese Remainder Theorem. By the theory of prime distribution, the largest prime ℓ necessary to find t is bounded by $O(\log(p))$.

We denote the ℓ -th *division polynomial* by f_ℓ , whose degree is $(\ell^2 - 1)/2$ for $\ell \geq 3$, and which vanishes precisely on the x -coordinates of the ℓ -torsion points. As we compute (2) in the ring $GF(p)[x, y]/(y^2 - x^3 - Ax - B, f_\ell(x))$, the dominant steps is the computation of x^p and y^p in that ring. From this, the complexity of this algorithm will be $O(\log^8(p))$.

2.2 Atkin-Elkies' Improvement (SEA)

Elkies' idea [7,23] is to make use of a degree $(\ell - 1)/2$ factor g_ℓ of f_ℓ when it is possible to compute g_ℓ in $GF(p)[x]$. (In this case, ℓ is called an *Elkies prime*. Otherwise ℓ is called an *Atkin prime*). The factor g_ℓ represents an eigenspace of the Frobenius map ϕ , which can be computed as a kernel of an isogeny map. Thus, $t \bmod \ell$ is calculated by the eigenvalue of the eigenspace. As the ratio of Elkies primes is expected $1/2$, this method will reduce the complexity to $O(\log^6(p))$. Rather than determining the unique value of $t \bmod \ell$, Atkin [1,23] obtained certain restrictions on the value. Then the real value of t is found among a lot of candidates by, for example, the match-and-sort technique [12].

The Schoof-Elkies-Atkin(SEA) method is obtained by combining the above two and so it is consisting of two stages, namely, (I) *collecting information stage* and (II) *trial search stage*:

SEA

(I) Collecting informations on $t \bmod \ell$ for various ℓ 's until $\prod \ell > 4\sqrt{p}$:

- (i) Compute the modular polynomial $\Phi_\ell(x, j)$.
- (ii) Check if $\bar{\Phi}(x) = \Phi_\ell(x, j(E)) \bmod p$ has a root in $GF(p)$.
 - (ii-E) If $\bar{\Phi}(x)$ has a root in $GF(p)$ (we call ℓ an Elkies prime) calculate $t_\ell = t \bmod \ell$ using g_ℓ .
 - (ii-A) Otherwise (we call ℓ an Atkin prime) calculate possible values of $t \bmod \ell$. We denote the set of possible values of $t \bmod \ell$ by $\mathcal{T}_\ell$.

The set of Elkies primes is represented by $\mathcal{E}$, and the set of Atkin primes is represented by $\mathcal{A}$.

(II) Determining the value of t by trial search:

Now, there are candidates T for the value of t , where

$$T \bmod \ell = t_\ell \text{ for } \ell \in \mathcal{E}, \quad \text{and} \quad T \bmod \ell \in \mathcal{T}_\ell \text{ for } \ell \in \mathcal{A}.$$

The value of t is (uniquely) determined by trial search, that is, by testing if $(p + 1 - T)P = \mathcal{O}$ for each candidate T , where P is a sample rational point of E and $\mathcal{O}$ is the point of infinity, and this test is efficiently executed by the match-and-sort technique.

2.3 Isogeny Cycles Method

According to Morain *et al.* [5,6], $t \bmod \ell^2$, $t \bmod \ell^3, \dots$ can be computed efficiently when ℓ is an Elkies prime. In the method, a factor g_{ℓ^k} of the ℓ^k -th division polynomial f_{ℓ^k} is computed, where the degree of g_{ℓ^k} is at most $\ell^{k-1}(\ell - 1)/2$. The isogeny cycles method is designed as a practical improvement to SEA, and so the following shall be modified: In (I), we replace $\prod \ell$ with $\prod \ell^k$ and in (II) $T \bmod \ell^k = t_{\ell^k}$ for each $\ell \in \mathcal{E}$, where k depends on each Elkies prime ℓ . For practical implementations and improvements, see [5,6].

From now on, we will consider methods based on SEA with the isogeny cycles method. Thus, we will gather informations of $t \bmod \ell^k$ for some $k > 0$ in the stage (I) and we call the product of all primes or prime powers whose informations will be used in the trial search stage (II) the **counter**. So, when **counter** exceeds $4\sqrt{p}$ in the stage (I), we enter the next stage (II).

3 Intelligent Choice System

In implementation of SEA with the isogeny cycles method, the following choices are very important for the total efficiency;

1. decision whether to apply the isogeny cycles method for $t \bmod \ell^k$ or not, when we find an Elkies prime ℓ ,
2. decision whether to compute the candidates for the value of t or just abandon it, when we find an Atkin prime, and
3. the setting on **counter** and usage of informations with respect to Atkin primes.

To give an efficient choice in various situations, and to optimize the total computation, we have to examine how the total efficiency will be changed for each choice. From this point, the estimate (guess) of the computational cost shall be very helpful. Here, as an attempt for this optimization, we will propose new strategies, usage of the estimate of costs to make an “efficient choice” at each step and systematic treatment of Atkin primes. We call the total system with these strategies an *intelligent choice system*. Of course, we can incorporate strategies in previous works into our system, as those were proposed to improve the efficiency of *each* subprocedure (method). (See Remark 2 (1).) Here we note that to make our approach effective, estimates should correspond to the real costs precisely, which requires “efficient implementation” of each operation.

Fig. 1. The diagram of choice of methods

So far we have three methods (categories) to get the information on $t \bmod \ell^k$:

- (a) *Schoof's original algorithm*, i.e. finding $t \bmod \ell$ in (2) for $P \in E[\ell]$,
- (b) *Atkin-Elkies' method*, i.e. the part (I-ii) of SEA for ℓ , and
- (c) *the isogeny cycles method* for ℓ^k .

Now we add *virtual methods* to take a shortcut to the trial search stage (II), which we call *virtual Atkin and virtual isogeny cycles method*.

3.1 Shortcut to the Trial Search Stage

In the stage (I), to make good use of Atkin primes ℓ , we allow candidates for $t \bmod \ell$ as information on t . Thus, in the stage (II), we search a correct value of t among all candidates constructed from informations in the stage (I). And it takes $O(\sqrt{N_C})$ additions of points on the curve, where N_C denotes the total number of candidates, by most efficient algorithm (match-and-sort/baby-step-giant-step algorithm). If N_C gets too large, the search will take much time. Therefore we set an upper limit to the total number of candidates, which we denote by **CanMAX**. We define **CanMAX** from experiments according to the size of p based on the complexity for doing match-and-sort algorithm. In “lucky” cases, we enter the stage (II) with much smaller N_C than **CanMAX**. In this case, by introducing *virtual methods*, we can take a shortcut to the stage (II).

From now on, $\mathcal{T}_{\ell^k}$ denotes the set of candidates for $t \bmod \ell^k$, which is obtained by Schoof's original method, Atkin-Elkies' method or the isogeny cycles method.

Virtual (Atkin/Isogeny Cycles) Method: Suppose that $\mathcal{T}_{\ell^k}$ is already computed. (For $k = 0$, we have no information on $\bmod \ell$.) If a prime ℓ satisfies the following three conditions, we regard the candidates for $t \bmod \ell^{k+j}$ for some $j > 0$ as $\mathcal{T}_{\ell^{k+j}} = \{a + b\ell^k \mid a \in \mathcal{T}_{\ell^k}, 0 \leq b \leq \ell^j - 1\}$.

- (i) For primes $\ell_1, \dots, \ell_s$, we have already computed $\mathcal{T}_{\ell_i^{k_i}}$ and $4\sqrt{p} > \mathbf{counter}$,
i.e. $4\sqrt{p} > (\prod_{1 \leq i \leq s; \ell_i \neq \ell} \ell_i^{k_i}) \times \ell^k$.
- (ii) $4\sqrt{p} < (\prod_{1 \leq i \leq s; \ell_i \neq \ell} \ell_i^{k_i}) \times \ell^{k+j}$. (The updated **counter** exceeds $4\sqrt{p}$.)
- (iii) **CanMAX** $> (\prod_{1 \leq i \leq s; \ell_i \neq \ell} \#\mathcal{T}_{\ell_i^{k_i}}) \times \#\mathcal{T}_{\ell^{k+j}}$. (The updated N_C does not exceed **CanMAX**.)

We can consider several primes ℓ_m 's at the same time. In this case, we can replace the inequalities (ii), (iii) with

- (iia) $4\sqrt{p} < (\prod_i \ell_i^{k_i}) \times (\prod_m \ell_m^{k_m+j_m})$, where $\ell_i \neq \ell_m$.
- (iiia) **CanMAX** $> (\prod_i \#\mathcal{T}_{\ell_i^{k_i}}) \times (\prod_m \#\mathcal{T}_{\ell_m^{k_m+j_m}})$, where $\ell_i \neq \ell_m$.

By this, we can reduce the candidates as well as the operations in the search.

3.2 Choice of Methods Based on Estimate of the Complexity

Now we have four methods to get the information on $t \bmod \ell^k$: (a) Schoof's original algorithm, (b) Atkin-Elkies' method, (c) the isogeny cycles method, and

(d) the virtual method. In each step we should be careful in choosing one of these method, as the choice will effect the total time of computations. For example, when **counter** is very close to $4\sqrt{p}$, it might be more efficient to compute $t \bmod \ell^k$ for a small Elkies prime ℓ , rather than to compute $t \bmod \ell$ for a large prime ℓ . Thus, as for the isogeny cycles method, it is very important to decide when we apply it and which prime we use.

Estimate of the Complexity: Here, we propose a simple strategy where we will estimate the complexity of each method and choose the most efficient one and a prime ℓ as a most efficient systematic choice. As a simple but practical example, we construct the following “complexity estimate function” by estimating the dominating computations. (We count the number of arithmetic operations over $GF(p)$.) For simplicity, we only deal with $t \bmod \ell^2$ in the isogeny cycles case and assume that $\ell \ll \log(p)$. The function is very simple, but it works very well for actual computation. (See Section 4.2.) In the implementation we fix each *weight* $w_s, w_a, w_{i,1}, w_{i,2}, w_{i,3}$ to fit the actual computation. Here we denote by $M(n)$ the time needed to compute the product of two polynomials of degree n .

Complexity Estimate Function (Weights $w_s, w_a, w_{i,1}, w_{i,2}, w_{i,3}$ are positive numbers.)

- (a) In Schoof's (original) algorithm case, we estimate its complexity at $w_s M((\ell^2 - 1)/2) \log(p)$ for a prime ℓ .
- (b) In Atkin-Elkies' method case, we estimate its complexity at $w_a M(\ell + 1) \log(p)$ for a prime ℓ .
- (c) In the isogeny cycles method case, we estimate its complexity at $w_{i,1} M(U) \log(p) + w_{i,2} M(\ell + 1) \log(p) + w_{i,3} M((\ell - 1)/2) \log((\ell - 1)/2) \log(p)$ for an Elkies prime ℓ , where U is the degree of a factor g_{ℓ^2} of f_{ℓ^2} that is used in computing $t \bmod \ell^2$.
- (d) If we can apply the virtual method, i.e. three conditions for the virtual method is satisfied, we estimate its complexity at 0. Otherwise, we estimate its complexity at ∞ .

We explain the above estimate briefly. In each method, its dominant step is the computation of $h_0(x)^p \bmod h(x)$ or that of $h_0(x)^{(p-1)/2} \bmod h(x)$ for polynomials h_0, h . In more detail, $h_0 = x, y(= x^3 + ax + b)$ and $h = f_\ell$ for (a), and $h_0 = x$ and $h = \Phi_\ell(x, j(E))$ for (b). For (c), $h_0 = x$ and $h = \Phi_\ell(x, j(E/C))$, $h_0 = x$ and $h = g_{\ell^2}$, and randomly chosen h_0 and $h = \hat{g}_\ell$. (See Lemma 1 for E/C and $\hat{g}_\ell$.) By the experimental analysis in [14] and our experiment, these steps amount to more than $2/3$ of the total time, which shall support the validity of the function.

In (a) we assume that we apply Schoof's original method to smaller ℓ and so the computation of $\phi(P)$ is dominant. (See Remark 2 (3).)

In (b) we count only the cost to check if $\Phi_\ell(x, j(E))$ has a root in $GF(p)$. If ℓ is an Atkin prime and we execute Atkin's method, i.e. (ii-A) in SEA, additional cost does not effect the total cost so much. But, if ℓ is an Elkies prime and we execute Elkies' method, i.e. (ii-E) in SEA, additional computations, the computation of

$\phi(P)$ and finding the eigenvalues, have certain effect on the total cost. See the paragraph Further Discussion on Practical Choice for more precise estimate.

In (c) the degree U is guessed by the following lemma used in [6], which can be shown easily by seeing the action of the Galois group.

Lemma 1 *Suppose that ℓ is an odd Elkies prime and g_ℓ corresponds to an eigenspace C of ϕ in $E[\ell]$ with eigenvalue s_0 . Let d_0 be the order of s_0 in the multiplicative group $GF(\ell)^*$ and set $d = d_0 / \gcd(2, d_0)$. Then, g_ℓ has a factor of degree d over $GF(p)$ and so its corresponding polynomial $\hat{g}_\ell$ with respect to the isogeny curve E/C , has also a factor of degree d over $GF(p)$. Moreover, f_{ℓ^2} has a factor g_{ℓ^2} of degree ld over $GF(p)$, which can be used for computing $t \bmod \ell^2$.*

In the computation of $\phi(P)$, the step $x^p \bmod g_{\ell^2}$ is dominant. (See Section 4.1.) In the computation of $\hat{g}_\ell$, the step $x^p \bmod \Phi_\ell(x, j(E/C))$ is dominant, and in the computation of a factor of $\hat{g}_\ell$, the steps $h_1^{(p-1)/2} \bmod \hat{g}_\ell$ are dominant, where h_1 are randomly chosen polynomials. See [24] for details of factoring polynomials.

Remark 2 (1) By the isogeny cycles method, we can compute $t \bmod \ell^k$ exactly, however, its cost tends large compared with Atkin-Elkies' method for the same ℓ . So its arrangement is very important. In existing works, the isogeny cycles method was considered as an option to Elkies primes, and it is applied for ℓ just after Elkies' method was executed for the same ℓ ([14]). (The power k for ℓ^k is also decided at this step.) But, for primes $\ell_1 < \ell_2$, there are many cases where the cost for $t \bmod \ell_1^2$ by the isogeny cycles method exceeds that for $t \bmod \ell_2$ by Atkin-Elkies' method. Moreover, for Elkies primes $\ell_1 < \ell_2$, the cost for $t \bmod \ell_1^2$ is not always smaller than that for $t \bmod \ell_2^2$. (See Lemma 1.)

(2) The condition for the virtual method shall depend on the cost of search in the stage (II). To guarantee the efficiency of the intelligent choice system, the current setting is derived from an assumption that the cost of search among **CanMAX** candidates corresponds to that of Atkin-Elkies' method just before closing the stage (I).

(3) As a common strategy for Schoof's original algorithm, we apply it only for small primes and so we set a bound on the largest prime for which Schoof's original algorithm can be applied. Then, the estimated cost for Schoof's original algorithm for large primes become ∞ in the intelligent choice system.

Examples 3 Here we demonstrate the detail of actual computation by examples. We select typical examples for which virtual methods work well. In the below, series of triples represent the trace of computation, where each triple consists of the selected method, the selected prime, and the actual cost (in seconds). For simplicity, we write **a,e,i** for Atkin's method, Elkies' method and the isogeny cycles method, respectively. (In the implementation, we computed $t \bmod 2$ by checking if E has a rational point of order 2. Thus, each series begins with $\ell = 3$.) Let $p = 2^{155} + 15$ and E defined by $y^2 = x^3 + ax + b$.

(1)

$(a, b) = (1, 5)$:
 [e,3,0.1], [a,5,0.1], [i,3,0.11], [e,7,0.26], [e,11,0.51], [e,13,0.56], [i,7,0.4],
 [e,17,1.05], [e,19,1.05], [i,11,0.83], [a,23,1.31], [i,13,1.66], [a,29,1.84],
 [e,31,2.75], [i,17,2.4], [e,37,3.5], [e,41,4.99], [e,43,5.23]

Then, the virtual method was applied with the following setting: $2 \rightarrow 2^4$, $3 \rightarrow 3^2$, $7^2 \rightarrow 7^3$, $11^2 \rightarrow 11^3$. The trial search took 1.67 seconds.

(2) $(a, b) = (1, 64)$:

[e,3,0.18], [e,5,0.18], [i,3,0.12], [e,7,0.25], [i,5,0.25], [a,11,0.32], [e,13,0.55],
[e,17,0.92], [a,19,0.73], [a,23,1.32], [i,13,1.25], [e,29,2.55], [i,7,0.97], [a,31,2.09],
[i,17,2], [e,37,3.49], [a,41,3.59], [e,43,5.32], [a,47,4.33]

Then, the virtual method was applied with the following setting: $2 \rightarrow 2^3$, $3^2 \rightarrow 3^3$, $5^2 \rightarrow 5^3$, The trial search took 1.04 seconds.

Further Discussion on Practical Choice: Here we proposed a strategy based on complexity estimate as a practical optimization. But, from precise analysis, it might be better to use the following “contribution index” instead of the complexity estimate function.

1. **Contribution Index:** On the choice of methods, the “gain” in `counter` must be taken into account. (Informations for large moduli shall contribute much more than those for small moduli.) As a realization of such “contribution”, we may set the *contribution index* for each method by

$$\frac{\text{the estimate of the complexity}}{\text{the gain in counter}}.$$

We can give a precise argument on the gain. As for Atkin-Elkies' method with a prime ℓ , if ℓ is an Elkies prime, then the gain is exactly ℓ , however, if ℓ is an Atkin prime, the gain varies according to the number of candidates of $t \bmod \ell$ and the amount of already computed candidates. So, we might compute the “expected gain” by taking the probabilities into account. (See Section 3.3 for usage of Atkin primes.)

2. **Cost of Virtual Method:** We can give a “more reasonable” estimate to the cost of the virtual method. As the virtual method has much influence on searching the real value of t , it is better to add the estimate of the cost of search to the original estimate, and to compare it with the expected total cost for the case where we do not apply the virtual method.

By our experiment, there is little difference between the original complexity estimate function and the contribution index for 300 curves over $GF(2^{160}+7)$. So, “simple estimate of the cost” seems work well for curves over finite fields of 160-240 bits order. But, to deal with curves over larger finite fields, we have to consider a precise contribution index function to make the intelligent choice system efficient. As for the cost of the virtual method, we could not have any practical experiment due to the hardness to estimate actual costs of search by the match-and-sort technique. This shall be done in the future work.

3.3 Re-ordering Atkin Primes

When Atkin-Elkies' method is chosen, we check whether $\Phi_\ell(x, j(E))$ has a root in $GF(p)$ from $\gcd(x^p - x, \Phi_\ell(x, j(E)))$. Thus, we computed $x^p \pmod{\Phi_\ell(x, j(E))}$

for this check. If $\Phi_\ell(x, j(E))$ has not such a root, i.e. ℓ is an Atkin prime, then the candidates of $t \bmod \ell$ are computed from the distinct degree decomposition (DDD). Once we have $x^p \pmod{\Phi_\ell(x, j(E))}$, we can compute the DDD very efficiently and so the computation $x^p \pmod{\Phi_\ell(x, j(E))}$ is the dominant step in this case. But, to improve the total efficiency, there proposed a strategy that we do not execute any additional computation for Atkin primes when the total number N_C of candidates of t exceeds **CanMAX**.

Here, we propose a new strategy “re-ordering” that even if N_C exceeds **CanMAX**, we do not give up using new Atkin primes. A “good” Atkin prime is the one, which itself is fairly large and the number of whose candidates for t is small. We define “Atkin index” of an Atkin prime ℓ as:

$$\frac{\text{the number of candidates for the value } t}{\ell}.$$

In this context, Atkin primes of smaller index can be used more efficiently for the computation. When we find a new Atkin prime and N_C exceeds **CanMAX**, we look for “worse” Atkin primes and replace them with the new “better” one so that N_C does not exceed **CanMAX**. In good cases we can proceed to the trial search stage without further computations.

Examples 4 We consider the case where $p = 2^{160} + 7$, $E : y^2 = x^3 + x + 8$ and **CanMAX** = 10^8 . In this case, successive 7 primes from $\ell = 59$ are Atkin primes and N_C exceeds **CanMAX** at $\ell = 71$. However, by “re-ordering”, **counter** exceeds $4\sqrt{p}$ at $\ell = 79$ with $N_C = 47185920$. By the notation in Example 3, the following presents the record of computation:

[a,3,0.05], [a,5,0.09], [e,7,0.21], [a,11,0.29], [a,13,0.39], [e,17,0.99],
[a,19,0.78], [a,23,1.25], [e,29,2.86], [e,31,2.79], [e,37,3.74], [e,41,5.77],
[a,43,4.61], [a,47,5.6], [e,53,7.82], [a,59,7.14], [a,61,8.2], [a,67,12.05],
[a,71,14.49], [a,73,14.71], [a,79,13.91]

On the other hand, without “re-ordering”, we have to search the next Elkies prime, which will be found at $\ell = 89$. Thus, the computation becomes 1.3 times slower.

Remark 5 The “re-ordering” strategy for Atkin primes obliges us to execute additional computation even for “bad” Atkin primes. But, this fact supports the validity of the complexity estimate function, because the difference between the cost for Elkies’ method and that for Atkin’s method for the same size prime becomes smaller.

4 Implementation and Experiment

We have implemented the intelligent choice system using Risa/Asir computer algebra system [20] developed by FUJITSU Labs. We examined its ability and efficiency by experiment on number of examples. We set $w_s = \infty$, $w_a = 1$, $w_{i,1} = w_{i,2} = 2$, $w_{i,3} = 0$, and $M(n) = n^2$. Although we used efficient multiplication techniques, where $M(n) = O(n^{1.6})$, the function gives reasonable choices

for our examples. For the experiment, we pre-computed “canonical” modular polynomials [19] up to $\ell = 229$.

4.1 Details in Implementation

Here, we explain the following important operations.

Multiplication and Powering: The SEA algorithm spends most of the time for multiplications of polynomials over $GF(p)$ and so those in the field $GF(p)$. Thus, as recommended in [14], we used Karatsuba's algorithm in the multiplication of polynomials so that we can get the complexity $M(n) = O(n^{1.6})$ instead of $O(n^2)$. This will be effective when p is around 240-bit long. Moreover, almost all are *modular multiplications*, i.e. polynomial multiplications accompanied with polynomial division. Among them, as remarked in [14], powering polynomials dominates among other operations and it appears in the following steps;

- (a) DDD computation of $\Phi_\ell(x, j(E))$,
- (b) eigenvalue computation, i.e. finding t_ℓ by $\phi(P) = t_\ell P$.

Because both (a) and (b) require

- (c) $x^{p^k} \bmod h(x)$ for some polynomial $h(x)$, or $h_0(x)^{(p-1)/2} \bmod h(x)$ for some polynomials $h_0(x), h(x)$, where $\deg(h) = O(\log(p))$.

For ordinary powering, we can convert one polynomial division into two polynomial multiplications with truncation, see [24]. For this truncated multiplication, we can extend Karatsuba's algorithm [8]. For p^k -th powering with $k \geq 2$, we can use multiplication tables, see also [24]. These techniques improve the total efficiency very much. We shall need, however, the FFT technique to attain drastical improvement for computation over much larger finite fields.

Eigenvalue Computation: In the isogeny cycles case, we made use of a match-and-sort algorithm in calculating an eigenvalue. As it was remarked in [23] that such a technique might work well for large ℓ , it seems to work well for isogeny cycles cases. We outline the implemented procedure briefly.

Suppose that ℓ is an odd Elkies prime and we have already computed g_ℓ and a factor g_{ℓ^2} of f_{ℓ^2} . Let s_0 be the computed eigenvalue of the eigenspace corresponding to g_ℓ . By using g_{ℓ^2} , we will compute the eigenvalue $s_0 + \ell s_1$, where $s_0, s_1 \in \{0, 1, \dots, \ell-1\}$ and $s_0 \neq 0$.

- (i) We compute $P, \dots, (\ell-1)P$, where the x -coordinate of P is a root of g_{ℓ^2} .
- (ii) For $k = 1, \dots, \ell-1$, we compute $k\phi(P)$ and check whether (ii-1) $vP = u\phi(P)$ or (ii-2) $-vP = u\phi(P)$, where $v = s_0 u \pmod{\ell}$.
If (ii-1) holds, then $s_0 + \ell s_1 = vu^{-1} \pmod{\ell^2}$. If (ii-2) holds, then $s_0 + \ell s_1 = -vu^{-1} \pmod{\ell^2}$.

This algorithm concerns only the x -coordinates of points and the number of additions of points is bounded by $2(\ell - 2)$. The following lemma gives the mathematical base for the correctness of the method.

Lemma 6 *For $s = s_0 + \ell s_1$, $s_0 \neq 0$, there exists an element $u_0 \in \{1, \dots, \ell - 1\}$ such that $u_0 s \bmod \ell^2$ belongs to $\{1, \dots, \ell - 1\} \cup \{\ell^2 - \ell + 1, \dots, \ell^2 - 1\}$.*

Match-and-Sort Computation in Trial Search: In the trial search stage, we applied a certain kind of match-and-sort algorithm described in [12] for improving the efficiency. (See [12] for details.) In the implementation, we made use of projective coordinates and pre-computed multiples of a fixed point for computation of baby step and giant step. Moreover, we take much care of partitioning Atkin primes into the baby part and the giant part. To give a good partition, we apply “re-ordering” again to all Atkin primes. Also in the virtual method, we choose primes to optimize the algorithm.

4.2 Experimental Results

To examine the ability of the intelligent choice system, we choose 300 curves over $GF(p)$, where $p = 2^{160} + 7$, $A = 1$, and $1 \leq B \leq 300$, and measured the average time needed to compute the cardinality of one curve on a PC with Pentium II of 300MHz. We set the value of **CanMAX**= 10^8 . We also put the best and the worst time in the following table. In order to see the effect of our methods we tried several combinations of our strategies. We did not implement the Schoof’s original algorithm.

Table 1. Using Intelligent Choice System (seconds):

No.	isogeny	virtual	re-ordering	best	average	worst
(1)	YES	YES	YES	34.7	66.5	334.7
(2)	NO	YES	YES	56.2	82.8	330.9
(3)	YES	NO	YES	43.7	76.1	339.4
(4)	YES	YES	NO	34.4	68.0	348.2

Table 2. Not Using Intelligent Choice System (seconds):

No.	isogeny	virtual	re-ordering	best	average	worst
(5)	YES	NO	NO	43.6	83.4	365.3
(6)	YES	NO	NO	43.6	86.9	374.1

(5) uses isogeny cycles if f_{ℓ^2} has a factor of degree ≤ 32 .

(6) uses isogeny cycles if f_{ℓ^2} has a factor of degree ≤ 64 .

From the above data, our strategies (in the intelligent choice system) will be characterized as follows:

1. The “estimate of the complexity” strategy has the main effect of speeding up the computation process overall.
2. The strategy on the isogeny cycles method and that on the virtual method have the main effect of speeding up the computation process in good cases. (Case we can proceed to the trial search stage early on.)

3. The re-ordering strategy should have the main effect of speeding up the computation process in bad cases. (Case the number of candidates for t exceeds **CanMAX**). Currently our implementation of calculation of x^{p^i} has not yet been tuned up. The authors believe that we can see a better effect after tuning up the process.

In Table 3 we show the timings (seconds) for 300 curves over other fields, which might assert that one can count the cardinality of curves used for elliptic curve cryptosystems in a reasonable time. And the average times suggest that the complexity of our implementation is $O(n^{5\pm\epsilon})$ for some $\epsilon \ll 1$. We also counted the cardinality of sample curves listed in X9.62 Working Draft and found the similar behavior on their timings as in Table 3. (It took 559 seconds for Example 1 with a 256-bit prime in H.5.3.)

Table 3. Statistics on Timings

prime	average time	best time	worst time	CanMAX
$2^{240} + 115$	454.1	242.8	1143.2	10^9
$2^{160} + 7$	66.5	34.7	334.7	10^8
$2^{155} + 15$	50.4	30.6	142.3	10^8

The authors are implementing the intelligent choice system for curves over finite fields of characteristic 2. As the basic arithmetics over finite fields of characteristic 2 can be done quite efficiently, the total computation over such fields seems faster than that over finite fields of odd characteristics.

4.3 Finding Elliptic Curves of Prime Cardinality

For secure ECC, it is strongly recommended to use a curve whose cardinality is a prime. For this purpose, we can use “early abort” strategy [13]. In this strategy, we check if the cardinality has a factor in each step of the computation of $t \bmod \ell$. If we find that the cardinality is not a prime, we can abandon the curve and try the next one. The effect of the strategy is supported by mathematical analysis [11,13].

We incorporated this strategy to our implementation and searched curve with prime cardinality. For $p = 2^{240} + 115$, we could try 3569 curves in 52.5 hours, and found 16 curves whose cardinalities are prime. It means that we can handle each curve within 1 minute in average due to the effect of early abort strategy (almost 8 times faster than the average time in Table 3).

5 Conclusion

We have introduced an explicit criterion for efficient computation of the cardinality of an elliptic curve over a finite field. The experiment shows that we could speed up the process almost 20%. In the experiment we can find elliptic curves whose cardinalities are prime numbers in a reasonable time when the characteristics p of the base field is around 240-bit long.

Although our current implementation can not be the best, its experimental result is quite satisfactory in practice. We are going to tune up the *complexity estimate function* and **CanMAX** to get a better result. We will also implement FFT for the case when p is a larger prime and modify the intelligent choice system for parallel computation. Moreover, theoretical analysis will be our further studies.

References

1. Atkin, A.O., The number of points on an elliptic curve modulo a prime, preprint, 1988.
2. Atkin, A.O., Morain, F., Elliptic curves and primality proving, *Math. Comp.* **61** (1993) 29–68.
3. Chao, J., Tanada, K., Tsujii, S., Design of elliptic curves with controllable lower boundary of extension degree for reduction attacks, in *CRYPTO '94*, Y.Desmedt, Ed., Lecture Notes in Computer Science, **839**, pp.50–55, 1994.
4. Charlap, L.S., Coley, R., Robbins, D.P., Enumeration of rational points on elliptic curves over finite fields, preprint, 1991.
5. Couveignes, J.-M., Dewaghe, L., Morain, F., Isogeny cycles and the Schoof-Elkies-Atkin algorithm, LIX/RR/96/03, 1996.
6. Couveignes, J.-M., Morain, F., Schoof's algorithm and isogeny cycles, in *ANT-I*, L.Adleman and M.-D.Huang, Eds., Lecture Notes in Computer Science, **877**, pp.43–58, 1994.
7. Elkies, N.D., Explicit isogenies, preprint, 1991.
8. Izu, T., Noro, M., Fast remainder calculation in polynomial multiplication, in preparation.
9. Koblitz N., Elliptic curve cryptosystems, *Math. Comp.* **48** (1987) 203–209.
10. Lay, G.-J., Zimmer, H.G., Constructing elliptic curves with given group order over large finite fields, in *ANT-I*, L.Adleman and M.-D.Huang, Eds., Lecture Notes in Computer Science, **877**, pp.250–263, 1994.
11. Lenstra Jr., H.W., Factoring integers with elliptic curves, *Annals of Mathematics* **126** (1987) 649–673.
12. Lercier, R., Algorithmique des courbes elliptiques dans les corps finis, Doctoral Thesis, L'École Polytechnique, 1997.
13. Lercier, R., Finding good random elliptic curves for cryptosystems defined over F_{2^n} , in *EURO-CRYPT '97*, W.Fumy, Ed., Lecture Notes in Computer Science, **1233**, pp.379–392, 1997.
14. Lercier, R., Morain, F., Counting the number of points on elliptic curves over finite fields: strategy and performances, in *EURO-CRYPT '95*, L.C.Guillou and J.-J.Quisquater, Eds., Lecture Notes in Computer Science, **921**, pp.79–94, 1995.
15. Menezes, A., Elliptic curve public key cryptosystems, Kluwer Academic Publishers, Boston, 1993.
16. Menezes, A., Okamoto, T., Vanstone, S.E., Reducing elliptic curves logarithms to logarithms in a finite field, in *STOC '91*, ACM Press, New York, pp.80–89, 1991.
17. Miller, V.S., Uses of elliptic curves in cryptography, in *CRYPTO '85*, Lecture Notes in Computer Science, **218**, pp.417–426, 1986.
18. Miyaji, A., Elliptic curves over F_p suitable for cryptosystems, in *AUSCRYPT '92*, J.Seberry and Y.Zheng, Eds., Lecture Notes in Computer Science, **718**, pp.479–491, 1992.

19. Morain, F., Calcul du nombre de points sur une courbe elliptique dans un corps fini: aspects algorithmiques, *J. Théor. Nombres Bordeaux* **7** (1995) 255–282.
20. Risa/Asir, (<ftp://endeavor.fujitsu.co.jp/pub/isis/asir>).
21. Satoh, T., Araki, K., Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves, *Commentarii Mathematici Universitatis Sancti Pauli* **47** (1998) 81–92.
22. Schoof, R., Elliptic curves over finite fields and the computation of square roots mod p , *Math. Comp.* **44** (1985) 483–494.
23. Schoof, R., Counting points on elliptic curves over finite fields, *J. Théor. Nombres Bordeaux* **7** (1995) 219–254.
24. Shoup, V., A new polynomial factorization algorithm and its implementation, *J. Symbolic Computation*. **20** (1995) 364–397.
25. Silverman, J.H., The arithmetic of elliptic curves, Graduate Texts in Mathematics **106**, Springer-Verlag, 1986.
26. Silverman, J.H., Advanced topics in the arithmetic of elliptic curves, Graduate Texts in Mathematics **151**, Springer-Verlag, 1994.
27. Smart, N.P., The discrete logarithm problem on elliptic curves of trace one, preprint, 1997.