PAPER Special Section on Discrete Mathematics and Its Applications

Scalar Multiplication on Pairing Friendly Elliptic Curves

Naoki KANAYAMA[†a)], ***Member***, **Tadanori TERUYA**[†], ***Nonmember***, ***and*** **Eiji OKAMOTO**[†], ***Member***

SUMMARY In the present paper, we propose elliptic curve scalar multiplication methods on pairing-friendly elliptic curves. The proposed method is efficient on elliptic curves on which Ate_i pairing or optimal pairing is efficiently computed.

key words: *scalar multiplication, Ate_i pairing, optimal pairing, endomorphism*

1. Introduction

Efficient computation of elliptic curve scalar multiplication has been a significant problem since Koblitz [13] and Miller [14] independently proposed elliptic curve cryptography, and several efficient methods of scalar multiplication have been proposed (e.g., [8], [9], [12]).

A standard approach for computing scalar multiplication is to use the Frobenius endomorphism. If we compute the s-multiplication on a point Q, denoted by $[s]Q$ (see Sect. 2.1), on an elliptic curve E over a finite field $\mathbb{F}_q$ of characteristic p, we expand s in base q and apply $[q]Q = -\pi_q^2(Q) + [t]\pi_q(Q)$ (where t is the trace of the Frobenius endomorphism π_q). This approach is very useful for small characteristic p (e.g., $p = 2, 3$). Kobayashi et al. [12] proposed an efficient method for the relatively large characteristic case.

Since Boneh et al. [2] and Sakai et al. [17] independently proposed ID-based cryptosystems using pairings, pairing-based cryptography has been a subject of great interest in cryptography. The fundamental components of several pairing-based cryptosystems are pairing computation and elliptic curve scalar multiplication.

The standard approach using the Frobenius endomorphism is useful for supersingular elliptic curves over small characteristic finite fields (e.g., $\mathbb{F}_{2^m}$ or $\mathbb{F}_{3^m}$). However, the Frobenius endomorphism is difficult to apply directly to elliptic curves over prime finite fields of large characteristic p because p is generally larger than the scalar s. Therefore, it is necessary to find a good "base" in which to expand s.

In the present paper, we propose elliptic curve scalar multiplication methods using the concepts of Ate_i pairing [21] and optimal pairing [22]. We can reduce the number of group operations over an elliptic curve by using these proposed methods.

The remainder of the present paper is organized as follows. Section 2 presents a brief description of elliptic curves and pairings and related research. Section 3 describes efficient scalar multiplication using the concepts of Ate_i pairing and optimal pairing. Finally, conclusions are presented in Sect. 4.

Manuscript received September 27, 2010.
Manuscript revised January 10, 2011.

[†]The authors are with the Department of Risk Engineering, Faculty of Systems and Information Engineering, University of Tsukuba, Tsukuba-shi, 305-8573 Japan.
a) E-mail: kanayama@risk.tsukuba.ac.jp

DOI: 10.1587/transfun.E94.A.1285

2. Preliminaries

2.1 Elliptic Curves

In this subsection, we present a brief description of elliptic curves and pairings. See [7] or [23] for details.

Let $E : Y^2 = X^3 + aX + b$ be an elliptic curve over a finite field $\mathbb{F}_q$ with characteristic $p \neq 2, 3$. We denote the point at infinity of E as O. For an extension field $\mathbb{F}_{q^i}$, the set of $\mathbb{F}_{q^i}$-rational points on E, denoted by $E(\mathbb{F}_{q^i})$, is defined by $E(\mathbb{F}_{q^i}) = \{(x, y) \in E \ : \ x, y \in \mathbb{F}_{q^i}\} \cup \{O\}$, and $E(\mathbb{F}_{q^i})$ is a group under addition on elliptic curves. We denote the inverse of $Q \in E$ as $-Q$.

For a point $Q \in E$ and an integer n, n-multiplication of Q, denoted by $[n]Q$, is defined by

$$[n]Q = \begin{cases} \underbrace{Q + Q + \cdots + Q}_{(n-1)\text{ additions}} & \text{if } n > 0, \\ O & \text{if } n = 0, \\ \underbrace{(-Q) + (-Q) + \cdots + (-Q)}_{(-n-1)\text{ additions}} & \text{if } n < 0. \end{cases}$$

We denote by π_q the q-th Frobenius endomorphism on E, that is, $\pi_q : (x, y) \mapsto (x^q, y^q)$. Then, $\#E(\mathbb{F}_q) = q + 1 - t$, where t is the trace of π_q.

Let r be a large prime number with $r|\#E(\mathbb{F}_q)$ and $(r, q) = 1$. We denote by $E[r]$ the group of r-torsion points on E. In this case, $E[r] \cong \mathbb{Z}/(r) \times \mathbb{Z}/(r)$, because $(r, q) = 1$.

The embedding degree k with respect to q and r is the smallest positive integer with $r|(q^k - 1)$. We denote by $\mu_r (\subset \mathbb{F}_{q^k}^*)$ the group of r-th roots of unity.

2.2 Related Research

In this section, we briefly review the techniques used in scalar multiplication.

2.2.1 GLV Multiplication

Gallant et al. [9] proposed an efficient scalar multiplication

on elliptic curves having endomorphisms of a certain form.

Let $E_1 : Y^2 = X^3 + aX$ be an elliptic curve over a prime finite field $\mathbb{F}_p$ with $p \equiv 1 \pmod 4$. Choose $\alpha \in \mathbb{F}_p^*$ of order 4. Then, the map $\phi_1 : (x, y) \mapsto (-x, \alpha y)$ is an endomorphism on E_1 over $\mathbb{F}_p$. The action of ϕ_1 on $E[r]$ corresponds to λ-multiplication, where, λ is a scalar that satisfies $\lambda^2 + 1 \equiv 0 \pmod r$.

Similarly, we can consider an elliptic curve $E_2 : Y^2 = X^3 + b$ over $\mathbb{F}_p$ with $p \equiv 1 \pmod 3$ having an endomorphism $\phi_2 : (x, y) \mapsto (\beta x, y)$ for $\beta \in \mathbb{F}_p^*$ of order 3 that corresponds to λ such that $\lambda^2 + \lambda + 1 \equiv 0 \pmod r$[†].

2.2.2 χ-Based Multiplication

Nogami et al. [16] and Sakemi et al. [18] proposed efficient pairing computation and scalar multiplication methods for the BN curves [4]. As they reported, their techniques can be applied for other pairing-friendly elliptic curve families. In the present paper, we briefly review their methods, particularly those of Nogami et al. [16].

These methods treat the following parameter family of embedding degree $k = 12$, as reported by Barreto et al. [4]:

$$\begin{aligned} p(X) &= 36X^4 - 36X^3 + 24X^2 - 6X + 1, \\ t(X) &= 6X^2 + 1, \\ r(X) &= 36X^4 - 36X^3 + 18X^2 - 6X + 1. \end{aligned}$$

(Polynomials $p(X)$ and $r(X)$ are slightly different from those in [4]. See **Example 2** in Sect. 3.1.1.)

The monomial $6X$ is written as

$$\begin{aligned} 6X &\equiv 1 + p(X) + p(X)^3(1 - p(X)) \\ &= 1 + p(X) + p(X)^3 - p(X)^4 \pmod{r(X)}, \end{aligned}$$

Therefore, $6X$-multiplication is expressed as a polynomial in the Frobenius endomorphism.

We now fix X as an integer. To compute $[s]Q$, we expand s as

$$s = s_0 + s_1 \cdot 6X + s_2 \cdot 6X^2 + s_3 \cdot 36X^3$$

because $6X^2 (= t(X) - 1))$ and $36X^3$ are polynomials in the Frobenius endomorphism. Nogami et al. and Sakemi et al. showed that $0 \le s_0, s_2 \le 6X$ and $0 \le s_1, s_3 \le X$. Therefore, computing $[s]Q$ can be done efficiently.

3. Primary Result

In this section, we explain the primary result of the present paper. As mentioned in Sect. 1, we consider scalar multiplication on pairing-friendly elliptic curves.

For applications to pairing cryptosystems, we usually consider points P and Q, which are elements in the following groups:

$$\begin{aligned} \mathbb{G}_1 &= E(\mathbb{F}_q)[r] = E[r] \cap \mathrm{Ker}(\pi_q - 1), \\ \mathbb{G}_2 &= E[r] \cap \mathrm{Ker}(\pi_q - q) \end{aligned}$$

We hereinafter assume that $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.

3.1 Scalar Multiplication on $\mathbb{G}_2$

Since $Q \in \mathbb{G}_2 = E[r] \cap \mathrm{Ker}(\pi_q - q)$, the Frobenius endomorphism π_q acts on Q as q-multiplication. Hence $[q^i]Q = [T_i]Q = \pi_q^i(Q)$. Therefore, due to its simplicity, we first consider scalar multiplication on $\mathbb{G}_2$. We begin with the results.

Theorem 1 *Let E be an elliptic curve over a finite field $\mathbb{F}_q$. Let r be a prime number of $\#E(\mathbb{F}_q)$ with $(r, q) = 1$, and let k be the embedding degree with respect to r and q. For $Q \in \mathbb{G}_2 = E[r] \cap \mathrm{Ker}(\pi_q - q)$,*
(1) Let n be an integer such that $T_n := q^n \bmod r$ is the minimum of $\{T_i := q^i \bmod r : i = 1, 2, \cdots, k-1, 0 \le T_i \le r-1\}$. Then, we can reduce the number of group operations over an elliptic curve for computing $[s]Q$ by expanding the scalar s in base T_n. The number of group operations over an elliptic curve is at most $\log_2(T_n)$.
(2) Furthermore, assume that optimal pairing can be defined on E, that is, we can find an expression $\sum_{i=0}^{\phi(k)} c_i q^i \equiv 0 \pmod r$, where $\phi(k)$ is the Euler function of k. Let c_w be the component with the largest absolute value, that is, $|c_w| = \max\{|c_j|\}$. Then, we can reduce the number of group operations over an elliptic curve for computing $[s]Q$ by expanding s in base c_w. The number of group operations over an elliptic curve is at most $\log_2(c_w)$ if $c_i = 0, \pm 1$ for all $i \ne w$.

See Appendix C for details on Ate_i and optimal pairings.

3.1.1 Method 1: Scalar Multiplication Using the Concept of Ate_i Pairing

Let $T_n := \min\{T_i : i = 1, 2, \cdots, k-1, 0 \le T_i \le r-1\}$, and let n be an integer such that $[T_n]Q = \pi_q^n(Q)$. If the size of T_n is sufficiently smaller than that of the scalar s, s can be written as

$$s = s_0 + s_1 T_n + \cdots s_d (T_n)^d,$$

where d is the largest integer less than $\log_{T_n}(s)$ and $0 \le s_j \le T_n - 1$ for each $0 \le j \le d$. Then, $[s]Q$ for $Q \in \mathbb{G}_2$ is expressed as

$$\begin{aligned} [s]Q &= [s_0]Q + [s_1 T_n]Q + \cdots [s_d (T_n)^d]Q \\ &= [s_0]Q + [T_n]([s_1]Q) + \cdots [(T_n)^d]([s_d]Q) \\ &= [s_0]Q + \pi_q^n([s_1]Q) + \cdots \pi_q^{dn}([s_d]Q). \end{aligned}$$

Let $s_{max} := \max\{s_j\}_{1 \le j \le d}$. In the proposed method, the number of group operations over an elliptic curve is $\log_2(s_{max})$, as compared with that of the usual scalar multiplication, $\log_2(s) \approx \log_2(r)$. Therefore, the proposed method can be performed faster than the usual binary method.

Example 1 (A parameter family with $k = 24$) We apply

[†]The curve E_2 has another endomorphism: $\phi_2' : (x, y) \mapsto (\beta x, -y)$ for $\beta \in \mathbb{F}_p^*$ of order 3 that corresponds to λ such that $\lambda^2 - \lambda + 1 \equiv 0 \pmod r$.

the proposed method to a parameter family with $k = 24$ reported in [6].

$$\begin{aligned} p(X) &= \frac{1}{3}(X^{10} - 2X^9 + X^8 - X^6 + 2X^5 \\ &\quad - X^4 + X^2 + X + 1), \\ t(X) &= X + 1, \\ r(X) &= X^8 - X^4 + 1. \end{aligned}$$

In this case, $T_n = T_1 = X$. Therefore, the number of group operations over the elliptic curve is approximately $\frac{1}{8}\log_2(r)$.

Example 2 (BN elliptic curves)

$$\begin{aligned} p(X) &= 36X^4 + 36X^3 + 24X^2 + 6X + 1, \\ t(X) &= 6X^2 + 1, \\ r(X) &= 36X^4 + 36X^3 + 18X^2 + 6X + 1. \end{aligned} \tag{1}$$

In this case, $T_n = T_1 = 6X^2$. Therefore, the χ-based method of Nogami et al. is more efficient.

3.1.2 Method 2: Scalar Multiplication Using the Concept of Optimal Pairing

If we can compute the optimal pairing on E, then $\sum_{i=0}^{l} c_i q^i \equiv 0 \pmod r$. Let c_w be the coefficient such that $|c_w| = \max\{|c_j|\}$. Since $(q, r) = 1$,

$$c_w = -q^{w(k-1)} \sum_{i=0, i\neq w}^{l} c_i q^i.$$

Thus, we can express $[c_w]Q$ for $Q \in \mathbb{G}_2$ as

$$\begin{aligned} [c_w]Q &= \left[-q^{w(k-1)} \sum_{i=0,i\neq w}^{l} c_i q^i\right] Q \\ &= -\left[\sum_{i=0,i\neq w}^{l} c_i q^{w(k-1)+i}\right] Q \\ &= -\sum_{i=0,i\neq w}^{l} \pi_q^{w(k-1)+i}([c_i]Q), \end{aligned}$$

and hence we can compute $[s]Q$ efficiently by expanding s in base c_w.

Example 3 Consider the BN curves given in (1) again. In this case, we find a coefficient vector $[c_0, c_1, c_2, c_3] = [6X + 2, 1, -1, 1]$. Therefore, $c_w = c_0$ and

$$[6X + 2]Q = -\pi_q(Q) + \pi_q^2(Q) - \pi_q^3(Q).$$

Example 4 (A parameter family with $k = 10$ in [5]) Freeman [5] found the following parameter family, such that $k = 10$:

$$\begin{aligned} p(X) &= 25X^4 + 25X^3 + 25X^2 + 10X + 3, \\ t(X) &= 10X^2 + 5X + 3, \\ r(X) &= 25X^4 + 25X^3 + 15X^2 + 5X + 1. \end{aligned}$$

In this case, $[c_0, c_1, c_2, c_3] = [1, 1, -1, -5X - 1]$. Therefore, $c_w = c_3$ and

$$[5X + 1]Q = \pi_q^7(Q) + \pi_q^8(Q) - \pi_q^9(Q).$$

Remark 1 *Method 2 is efficient if $c_i = 0, \pm 1$ for all $i \neq w$. We refer to such a vector as a "good vector." The above examples show that BN curves and Freeman curves have good vectors.*

3.2 Scalar Multiplication on $\mathbb{G}_1$

In the case of $\mathbb{G}_1$, some powers of q give eigenvalues of some endomorphisms. According to a previous study [11], if $\#\mathrm{Aut}(E) = 4, 6$, $d = 3, 4, 6$, and $[\xi_d] \in \mathrm{Aut}(E)$ is an order d automorphism, then there exists $i \in \mathbb{Z}$ such that $[\xi_d]P = [q^{ei}]P = [T_{ei}]P$ for all $P \in \mathbb{G}_1$, where $e = k/\gcd(k, d)$. Maps $[\xi_d]$ for $d = 3, 4, 6$ are defined as follows:

$$\begin{aligned} &[\xi_4] : (x, y) \mapsto (-x, \alpha y), \text{ where } \alpha \in \mathbb{F}_q \text{ of order 4.} \\ &[\xi_3] : (x, y) \mapsto (\beta x, y), \text{ where } \beta \in \mathbb{F}_q \text{ of order 3.} \\ &[\xi_6] : (x, y) \mapsto (\beta x, -y), \text{ where } \beta \in \mathbb{F}_q \text{ of order 3.} \end{aligned}$$

By the definitions of q, r, and e, the above implies that $r \mid \Phi_d(q^e)$, where Φ_d is the d-th cyclotomic polynomial, and $\varphi(3) = \varphi(4) = \varphi(6) = 2$. Thus, if there exists $i \in \mathbb{Z}$ such that $\log_2(q^{ei} \bmod r) = \log_2 T_{ei} \approx \frac{1}{2}\log_2 r$, then we can decompose an integer $x \in \mathbb{Z}/(r)$ into two approximately equal parts as $x = x_0 + x_1 T_{ei}$, where $0 \leq x_0, x_1 < T_{ei}$, and we can then perform scalar multiplication by the GLV method, $[x]P = [x_0]P + [x_1][\xi_d]P$.

Example 5 We obtain the following parameter (p, r, t) with $k = 6$ and discriminant $\Delta = 3$ using Construction 6.6. in [6].

$$\begin{aligned} q(X) &= 27X^4 + 9X^3 + 3X^2 + 3X + 1, \\ r(X) &= 9X^2 + 3X + 1, \\ t(X) &= 3X + 2. \end{aligned}$$

In this case, we can set $d = 6$ because $\Delta = 3$. Therefore, we have $e = 1$. Hence, $T_{2e} = q(X)^{2e} \bmod r(X) = q(X)^2 \bmod r(X) = 3X$.

The condition that there exists $i \in \mathbb{Z}$ such that $\log_2 T_{ei} \approx \frac{1}{2}\log_2 r$ is not always satisfied in general, for example, in the case of BN curves. Sakemi et al. [18] proposed a method by which to achieve almost half decomposition for BN curves using the skew Frobenius endomorphism.

3.3 Application for Elliptic Curves Generated by the Cocks–Pinch Method

Generally, the ratio of elliptic curves generated by the Cocks–Pinch method suitable for Ate_i pairing or optimal pairing is very small. The proposed method can be applied

for elliptic curves generated by the Cocks–Pinch method if the curves have sufficiently small T_n or "good" vectors $[c_0, c_1, c_2, \cdots, c_{\phi(k)-1}]$.

Example 6 We use the following parameters (r, p, t) with embedding degree $k = 16$ generated by the Cocks–Pinch method:

$r =$ `0x00000001 08182010 00000000 00000000\`
`00000000 00000001`

$p =$ `0x20f6894a e7a636cf b82f1104 ee299407\`
`9181cafa d1d4b59a f929bbb5 631dfc13\`
`3824a88d da16e98d`

$t =$ `0x70b70de7 d3a02e1b 4c6300e7 fb06d82c\`
`69285d86`

The sizes of primes p and r are 318 bits and 161 bits, respectively. We first consider multiplication on $\mathbb{G}_2$. In this case, $T_n = T_{14} = 1108101562368$ is a 41-bit integer, and the number of group operations over the elliptic curve is approximately $\frac{1}{4}\log_2(r)$. For multiplication on $\mathbb{G}_1$, $\#\mathrm{Aut}(E) = 4$, and we can find $T_{12} = 122788907252240259376742 4$ such that $\log_2 T_{12} \approx \frac{1}{2}\log_2 r$, and

$\alpha =$ `0x161147dd ccbc109b a623f327 404a0434\`
`ff517c39 b49f4ea7 33afd33c 67d33501\`
`dcafc291 f4e5a6da` $\in \mathbb{F}_p$.

If we define $[\xi_4] : (x, y) \mapsto (-x, \alpha y)$, then $[T_{12}]P = [\xi_4]P$ for all $P \in \mathbb{G}_1$.

Remark 2 *(1) The Cocks–Pinch method enables group order to be input in advance. If we find an integer λ such that $r = \lambda^2 + 1$ is a prime number, then we can obtain an elliptic curve with prime subgroup order r and nice decomposition by λ. In Example 6, $r = T_{12}^2 + 1$, and scalar multiplication over $\mathbb{G}_1$ can be half decomposed by T_{12}.*
(2) We obtain a sufficiently small vector $[c_0, c_1, c_2, \cdots, c_7]$ $= [211062, 330890, -749144, 195974, -70328, 696136, 331743, -127932]$. *However, proposed method 2 is not efficient because components $c_i (i \neq w)$ are not sufficiently small compared to $c_w = 749144$. Therefore, Method 1 is more efficient in this case.*

3.4 Implementation

In order to demonstrate the advantage of the proposed method, we implemented experimental programs on a software platform based on the method described in Sect. 3.1.1 and selected Example 6 for scalar multiplication over $\mathbb{G}_2$ and Example 6 for scalar multiplication over $\mathbb{G}_1$. The software platform is as follows. The processor is an Intel Core i7 970 (3.2 GHz). The OS is 64-bit Linux 2.6.35, and the compiler is gcc 4.5.1. The experimental program is written in the C++ programming language using the NTL library version 5.5.2 [20]. No optimization techniques were used.

Algorithm 1 Scalar multiplication using the binary method

Input: $R \in \mathbb{G}_1$ or $\mathbb{G}_2$, and $s \in \mathbb{Z}$. An integer s is a positive integer with representation $s = (s_{i-1}, \ldots, s_0)_2$.
Output: $[s]R$.
1: $T \leftarrow R$;
2: Set i equal to the bitlength of s;
3: **for** j **from** $i - 2$ **down to** 0 **do**
4: $\quad T \leftarrow [2]T$;
5: $\quad$ **if** $s_j = 1$ **then**
6: $\quad\quad T \leftarrow T + R$;
7: $\quad$ **end if**
8: **end for**
9: **return** T;

Algorithm 2 Simultaneous multiple point multiplication over $\mathbb{G}_2$

Input: $Q \in \mathbb{G}_2$ and $s \in \mathbb{Z}$. An integer s is a positive integer such that $s = s_0 + s_1 T_{14} + s_2 T_{14}^2 + s_3 T_{14}^3$ and $0 \leq s_0, s_1, s_2, s_3 < T_{14}$, where p and T_{14} are defined as in Example 6 in Sect. 3.1.1. Integers s_i $(0 \leq i \leq 3)$ have representations $s_i = (s_{(i,j-1)}, \ldots, s_{(i,0)})_2$.
Output: $[s]Q$.
1: Compute a look-up table $\mathtt{Tbl}_Q$ of Q; (use Algorithm 3)
2: Decompose s and compute integers s_0, s_1, s_2, and s_3, and compute an integer j, which is the maximum of the bitlengths of $s_0, s_1, s_2, and s_3$;
3: $T \leftarrow \mathtt{Tbl}_Q((s_{(3,j-1)}, s_{(2,j-1)}, s_{(1,j-1)}, s_{(0,j-1)})_2)$;
4: **for** l **from** $j - 2$ **down to** 0 **do**
5: $\quad T \leftarrow [2]T$;
6: $\quad$ **if** $(s_{(3,l)}, s_{(2,l)}, s_{(1,l)}, s_{(0,l)})_2 \neq 0$ **then**
7: $\quad\quad T \leftarrow T + \mathtt{Tbl}_Q((s_{(3,l)}, s_{(2,l)}, s_{(1,l)}, s_{(0,l)})_2)$;
8: $\quad$ **end if**
9: **end for**
10: **return** T;

3.4.1 Algorithms

We decided to implement the scalar multiplication using the binary method and simultaneous multiple point multiplication [10]. Simultaneous multiple point multiplication requires a look-up table for input points Q over $\mathbb{G}_2$ and P over $\mathbb{G}_1$ as well as a decomposition of an input scalar s.

In Example 6, the implemented binary method is given by Algorithm 1, and the simultaneous multiple point multiplication is given by Algorithm 2 over $\mathbb{G}_2$ and Algorithm 4 over $\mathbb{G}_1$. Look-up table construction is performed using Algorithms 3 and 5. Integer decomposition is straightforward, i.e., the division and the remainder are iterated by the divisor T_{14} over $\mathbb{G}_2$ and T_{12} over $\mathbb{G}_1$.

3.4.2 Efficiency Analysis

The look-up table in simultaneous multiple point multiplication requires 2^n entries, where n is the number of partitions. In the algorithms of the present study, all of the values at even position $2i$ entries are computed by values at position i entries using $\pi_{p^{14}}$ over $\mathbb{G}_2$, and $[\xi_4]$ over $\mathbb{G}_1$. Therefore, there are only $2^{n-1} - 1$ even position computations and $2^{n-1} - 1$ additions.

Algorithm 3 Look-up table construction of Q

Input: $Q \in \mathbb{G}_2$. Let $p, T_{14} \in \mathbb{Z}$ be as defined in Example 6 in Sect. 3.1.1, and let $\pi_{p^{14}}$ be a p^{14}-Frobenius endomorphism on E.
Output: $\mathtt{Tbl}_Q$.

$\mathtt{Tbl}_Q((0,0,0,0)_2) \leftarrow O$;
$\mathtt{Tbl}_Q((0,0,0,1)_2) \leftarrow Q$;
$\mathtt{Tbl}_Q((0,0,1,0)_2) \leftarrow \pi_{p^{14}}(Q)$;
$\mathtt{Tbl}_Q((0,0,1,1)_2) \leftarrow \mathtt{Tbl}_Q((0,0,1,0)_2) + Q$;
$\mathtt{Tbl}_Q((0,1,0,0)_2) \leftarrow \pi_{p^{14}}(\mathtt{Tbl}_Q((0,0,1,0)_2))$;
$\mathtt{Tbl}_Q((0,1,0,1)_2) \leftarrow \mathtt{Tbl}_Q((0,1,0,0)_2) + Q$;
$\mathtt{Tbl}_Q((0,1,1,0)_2) \leftarrow \pi_{p^{14}}(\mathtt{Tbl}_Q((0,0,1,1)_2))$;
$\mathtt{Tbl}_Q((0,1,1,1)_2) \leftarrow \mathtt{Tbl}_Q((0,1,1,0)_2) + Q$;
$\mathtt{Tbl}_Q((1,0,0,0)_2) \leftarrow \pi_{p^{14}}(\mathtt{Tbl}_Q((0,1,0,0)_2))$;
$\mathtt{Tbl}_Q((1,0,0,1)_2) \leftarrow \mathtt{Tbl}_Q((1,0,0,0)_2) + Q$;
$\mathtt{Tbl}_Q((1,0,1,0)_2) \leftarrow \pi_{p^{14}}(\mathtt{Tbl}_Q((0,1,0,1)_2))$;
$\mathtt{Tbl}_Q((1,0,1,1)_2) \leftarrow \mathtt{Tbl}_Q((1,0,1,0)_2) + Q$;
$\mathtt{Tbl}_Q((1,1,0,0)_2) \leftarrow \pi_{p^{14}}(\mathtt{Tbl}_Q((0,1,1,0)_2))$;
$\mathtt{Tbl}_Q((1,1,0,1)_2) \leftarrow \mathtt{Tbl}_Q((1,1,0,0)_2) + Q$;
$\mathtt{Tbl}_Q((1,1,1,0)_2) \leftarrow \pi_{p^{14}}(\mathtt{Tbl}_Q((0,1,1,1)_2))$;
$\mathtt{Tbl}_Q((1,1,1,1)_2) \leftarrow \mathtt{Tbl}_Q((1,1,1,0)_2) + Q$;
return $\mathtt{Tbl}_Q$;

Algorithm 4 Simultaneous multiple point multiplication over $\mathbb{G}_1$

Input: $P \in \mathbb{G}_1$ and $s \in \mathbb{Z}$. An integer s is a positive integer such that $s = s_0 + s_1 T_{12}$ and $0 \le s_0, s_1 < T_{12}$, where p and T_{12} are as defined in Examples 6. Integers s_i $(i = 0, 1)$ have representations $s_i = (s_{(i,j-1)}, \ldots, s_{(i,0)})_2$.
Output: $[s]P$.

Compute a look-up table $\mathtt{Tbl}_P$ of P; (use Algorithm 5)
Decompose s and compute integers s_0 and s_1, and compute an integer j, which is the maximum of the bitlengths of s_0 and s_1;
$T \leftarrow \mathtt{Tbl}_P((s_{(1,j-1)}, s_{(0,j-1)})_2)$;
for l **from** $j-2$ **down to** 0 **do**
  $T \leftarrow [2]T$;
  if $(s_{(1,l)}, s_{(0,l)})_2 \neq 0$ **then**
    $T \leftarrow T + \mathtt{Tbl}_P((s_{(1,l)}, s_{(0,l)})_2)$;
  end if
end for
return T;

Algorithm 5 Look-up table construction of P

Input: $P \in \mathbb{G}_1$. Let $p, T_{12} \in \mathbb{Z}$, and let and $[\xi_4] : \mathbb{G}_1 \to \mathbb{G}_1$, all as defined in Examples 6.
Output: $\mathtt{Tbl}_P$.

$\mathtt{Tbl}_P((0,0)_2) \leftarrow O$;
$\mathtt{Tbl}_P((0,1)_2) \leftarrow P$;
$\mathtt{Tbl}_P((1,0)_2) \leftarrow [\xi_4]P$;
$\mathtt{Tbl}_P((1,1)_2) \leftarrow \mathtt{Tbl}_P((1,0)_2) + P$;
return $\mathtt{Tbl}_P$;

Let $s \in \mathbb{Z}$ be an input scalar, and let s' be a bitwise-or of all of the decompositions of s. This means that if s is decomposed as $s = \sum_{i=0}^{n-1} s_i T_m^i$, where T_m is the divisor, then $s' = s_0|s_1|\cdots|s_{n-1}$. Then, let $\mathrm{len}(s)$ and $\mathrm{len}(s')$ be the bitlengths of s and s', let $\mathrm{hw}(s)$ and $\mathrm{hw}(s')$ be the Hamming weights of s and s', and let $\mathrm{ave}(\mathrm{hw}(s))$ and $\mathrm{ave}(\mathrm{hw}(s'))$ be the averages of $\mathrm{hw}(s)$ and $\mathrm{hw}(s')$, respectively. Then, in the binary method, there are $\mathrm{len}(s) - 1$ doublings and $\mathrm{hw}(s) - 1$ additions. On the other hand, in simultaneous multiple point multiplication, there are $\mathrm{len}(s') - 1$ doublings and $\mathrm{hw}(s') - 1$ additions after table construction. Hence, the total cost estimations are

$$\begin{aligned}&\text{Cost of the Binary Method}\\ &\quad = (\mathrm{len}(s) - 1)\,\mathbf{DBL} + (\mathrm{hw}(s) - 1)\,\mathbf{ADD}, \qquad (2)\\ &\text{Cost of Simultaneous Multiple Point Multiplication}\\ &\quad = (2^{n-1} - 1)(\mathbf{EvenPos} + \mathbf{ADD})\\ &\qquad + (\mathrm{len}(s') - 1)\,\mathbf{DBL} + (\mathrm{hw}(s') - 1)\,\mathbf{ADD}, \qquad (3)\end{aligned}$$

where **DBL** and **ADD** are the costs of point doubling and addition, respectively, **EvenPos** is the cost of $\pi_{p^{14}}$ over $\mathbb{G}_2$ and $[\xi_4]$ over $\mathbb{G}_1$.

We assume that s is a uniformly distributed integer from 1 to $r - 1$. Approximately, the expectation of $\mathrm{len}(s)$ is $\mathrm{len}(r)$. In average cases, $\mathrm{hw}(s) \approx \mathrm{len}(s)/2$, $\mathrm{len}(s') \le \mathrm{len}(s)/n$, and $\mathrm{hw}(s') \ge \mathrm{len}(s')/2$. In addition, we assume that $\mathrm{len}(s') = \mathrm{len}(s)/n$, and $\mathrm{hw}(s') = \mathrm{len}(s')$. Then, based on these assumptions, we obtain the following inequality using Equations (2) and (3):

$$\begin{aligned}\mathrm{len}(r)\left\{\left(1 - \frac{1}{n}\right)\mathbf{DBL} + \left(\frac{1}{2} - \frac{1}{n}\right)\mathbf{ADD}\right\}\\ > \left(2^{n-1} - 1\right)(\mathbf{EvenPos} + \mathbf{ADD}). \qquad (4)\end{aligned}$$

In this inequality, the left-hand side denotes the improvement and the right-hand side denotes the overhead of the simultaneous multiple point multiplication, as compared with the binary method for average cases.

Finally, if s is close to or smaller than divisor T_{14} over $\mathbb{G}_2$ or T_{12} over $\mathbb{G}_1$, then the values given by Equation (2) would be smaller than the values given by Equation (3) by the overhead. However, we can neglect these cases. Because, from a cryptographic point of view, s must be large and uniformly distributed.

3.4.3 Experimental Results

We constructed the extension field $\mathbb{F}_{p^{16}} = \mathbb{F}_p[Z]/(Z^{16} - 2)$, elliptic curve $E/\mathbb{F}_p : Y^2 = X^3 + 6X$, and base points of $\mathbb{G}_1$, $\mathbb{G}_2$ (see Appendix A).

Based on the experimental programs of the preset study, the timings of **DBL**, **ADD**, and **EvenPos** are 0.15 msec, 0.15 msec, and 0.02 msec over $\mathbb{G}_2$ and 3.71 μsec, 4.32 μsec, and 0.52 μsec over $\mathbb{G}_1$, respectively. Additionally, timings of decomposition of input scalars are 2.0 μsec in Algorithm 2, and 0.7-1.6 μsec in Algorithm 4. These parts are cheaper than the others in both cases, hence, their costs are negligible. Then, in the case of $\mathbb{G}_2$, by inequality (4), we obtain:

$$\begin{aligned}\mathrm{LHS} &= 161 \cdot \left\{\left(1 - \frac{1}{4}\right) \cdot 0.15 + \left(\frac{1}{2} - \frac{1}{4}\right) \cdot 0.15\right\}\\ &= 24.15. \qquad (5)\end{aligned}$$

$$\mathrm{RHS} = 7 \cdot (0.02 + 0.15) = 1.19. \qquad (6)$$

Table 1 Properties of generated scalars.

No.	1	2	3	4
Bitlength	160	140	120	100
Hamming weight	78	71	61	55
Max bitlength over $\mathbb{G}_2$	40	40	40	40
Hamming weight over $\mathbb{G}_2$	38	33	36	35
Max bitlength over $\mathbb{G}_1$	80	80	79	80
Hamming weight over $\mathbb{G}_1$	52	54	48	52

No.	5	6	7	8
Bitlength	80	60	40	20
Hamming weight	36	32	18	9
Max bitlength over $\mathbb{G}_2$	40	40	40	20
Hamming weight over $\mathbb{G}_2$	29	25	18	9
Max bitlength over $\mathbb{G}_1$	80	60	40	20
Hamming weight over $\mathbb{G}_1$	36	32	18	9

Table 2 Timings of Algorithms 1, 2, and 3 over $\mathbb{G}_2$ [msec].

No.	1	2	3	4
Alg. 1	52.96	48.27	41.34	35.49
Alg. 2	22.64	21.13	21.50	21.22
Alg. 3	1.97	1.98	1.98	1.98

No.	5	6	7	8
Alg. 1	26.06	20.81	12.76	6.11
Alg. 2	18.92	16.75	14.47	7.96
Alg. 3	1.98	1.98	1.98	1.98

On the other hand, in the case of $\mathbb{G}_1$:

$$\begin{aligned}\text{LHS} &= 161 \cdot \left\{\left(1-\frac{1}{2}\right)\cdot 3.71 + \left(\frac{1}{2}-\frac{1}{2}\right)\cdot 4.32\right\} \\ &= 298.655. \end{aligned} \tag{7}$$

$$\text{RHS} = 1 \cdot (0.52 + 4.32) = 4.84. \tag{8}$$

In the both cases, inequality (4) is satisfied.

There are many dependencies on software platforms, e.g., CPUs, compilers, libraries, etc. As such, we measure the actual running times of the binary method and simultaneous multiple point multiplication for scalars of various length in order to demonstrate the validity of the experimental programs.

For these experiments, we generated eight different bitlength positive integers at random for input into Algorithms 1, 2, and 4. Table 1 shows the properties of the generated integers. The "Max bitlength over $\mathbb{G}_2$" and "Max bitlength over $\mathbb{G}_1$" rows show the bitlengths of $\max\{s_i : 0 \le i \le 3, s = s_0 + s_1T_{14} + s_2T_{14}^2 + s_3T_{14}^3\}$ for $\mathbb{G}_2$ and $\max\{s_i : 0 \le i \le 1, s = s_0 + s_1T_{12}\}$ for $\mathbb{G}_1$, where s is an input scalar. These maximum integers correspond to the loop length in Algorithms 2 and 4. The "Hamming weight over $\mathbb{G}_2$" and "Hamming weight over $\mathbb{G}_1$" rows show the Hamming weights of $s_0|s_1|s_2|s_3$ over $\mathbb{G}_2$ and $s_0|s_1$ over $\mathbb{G}_1$, which correspond to the number of point additions in Algorithms 2 and 4 for an input scalar s.

Tables 2 and 4 give the running times over $\mathbb{G}_2$ and $\mathbb{G}_1$ of our experimental program for generated scalars. Tables 3 and 5 show the ratios of the time required for Algorithm 1 to the time required for Algorithm 2 and the time required for 4, respectively. Based on Table 3, with scalar Nos. 1 through 6 in Table 1, Algorithm 2 is faster than Algorithm 1, whereas Algorithm 2 is slower than Algorithm 1 with scalar Nos. 7 and 8. Based on Table 5, with scalar Nos. 1 through 4, Algorithm 4 is faster than Algorithm 1, whereas the opposite is true for scalar Nos. 5 through 8.

Table 3 Ratio of timings of Algorithms 2 and 1 over $\mathbb{G}_2$.

No.	1	2	3	4
Alg. 2/Alg. 1	0.43	0.44	0.52	0.60

No.	5	6	7	8
Alg. 2/Alg. 1	0.73	0.80	1.13	1.30

Table 4 Timings of Algorithms 1, 4, and 5 over $\mathbb{G}_1$ [μsec].

No.	1	2	3	4
Alg. 1	1274.78	1133.09	967.74	834.30
Alg. 4	753.77	765.54	706.87	735.56
Alg. 5	6.00	6.00	6.00	6.00

No.	5	6	7	8
Alg. 1	611.87	487.27	300.77	144.94
Alg. 4	620.76	495.13	308.31	152.34
Alg. 5	6.00	6.00	6.01	6.00

Table 5 Ratio of timings of Algorithms 4 and 1 over $\mathbb{G}_1$.

No.	1	2	3	4
Alg. 4/Alg. 1	0.59	0.68	0.73	0.88

No.	5	6	7	8
Alg. 4/Alg. 1	1.01	1.02	1.03	1.05

Based on the experimental results, Algorithms 2 and 4 are faster than Algorithm 1 in many cases, and the advantage of $\mathbb{G}_1$ scalar multiplication is slightly less than the advantage of $\mathbb{G}_2$ scalar multiplication.

4. Conclusion

We proposed elliptic curve scalar multiplication methods using the concepts of Ate$_i$ pairing and optimal pairing. We can reduce the number of group operations over an elliptic curve by using the proposed methods. The proposed methods can also be applied to elliptic curves generated by the Cocks–Pinch method if the Ate$_i$ pairing or optimal pairing is efficiently computed over these curves.

References

[1] P.S.L.M. Barreto, S. Galbraith, C. ÓhÉigeartaigh, and M. Scott, "Efficient pairing computation on supersingular abelian varieties," Des. Codes Cryptogr., vol.42, no.3, pp.239–271, 2007.
[2] D. Boneh and M. Franklin, "Identity-based Encryption from the Weil pairing," CRYPTO 2001, LNCS, vol.2139, pp.213–229, Springer, 2001.
[3] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from Weil pairing," ASIACRYPT 2001, LNCS, vol.2248, pp.514–532, Springer, 2001.
[4] P.S.L.M. Barreto and M. Naehrig, "Pairing-friendly elliptic curve of prime order," SAC 2005, LNCS 3897, pp.319–331, 2006.

[5] D. Freeman, "Constructing pairing-friendly elliptic curves with embedding degree 10," ANTS 2006, LNCS, vol.3076, pp.452–465, Springer, 2006.
[6] D. Freeman, M. Scott, and E. Teske, "A taxonomy of pairing-friendly elliptic curves," J. Cryptology, vol.23, no.2, pp.224–280, 2010.
[7] S.D. Galbraith, "Pairings," in Advances in Elliptic Curve Cryptography, ed. I. Blake, G. Seroussi, and N. Smart, Chapter IX , Cambridge University Press, 2005.
[8] S.D. Galbraith, X. Lin, and M. Scott, "Endomorphisms for faster elliptic curve cryptography on a large class of curves," EUROCRYPT 2009, vol.5479, pp.518–535, 2009.
[9] R.P. Gallant, R.J. Lambert, and S.A. Vanstone, "Faster point multiplication on elliptic curves with efficient endomorphisms," CRYPTO 2001, LNCS, vol.2139, pp.190–200, 2001.
[10] D. Hankerson, A.J. Menezes, and S. Vanstone, Guide to Elliptic Curve Cryptography, Springer-Verlag New York, Secaucus, NJ, USA, 2004.
[11] F. Hess, N.P. Smart, and F. Vercauteren, "The eta pairing revisited," IEEE Trans. Inf. Theory, vol.52, no.10, pp.4595–4602, 2006.
[12] T. Kobayashi, H. Morita, K. Kobayashi, and F. Hoshino, "Fast elliptic curve algorithm combining frobenius map and table reference to adapt to higher characteristic," EUROCRYPT'99, LNCS, vol.1592, pp.176–189, 1999.
[13] N. Koblitz, "Elliptic curve cryptosystems," Math. Comp., vol.48, no.177, pp.203–209, 1997.
[14] V.S. Miller, "Use of elliptic curves in cryptography," CRYPTO'85, LNCS, vol.218, pp.417–426, Springer, 1985.
[15] Y. Nogami, M. Akane, Y. Sakemi, H. Kato, and Y. Morikawa, "Integer variable χ-based Ate pairing," Pairing 2008, LNCS, vol.5209, pp.178–191, Springer, 2008.
[16] Y. Nogami, Y. Sakemi, T. Okimoto, K. Nekado, M. Akane, and Y. Morikawa, "Scalar multiplication using frobenius expansion over twisted elliptic curve for Ate pairing based cryptography," IEICE Trans. Fundamentals, vol.E92-A, no.1, pp.182–189, Jan. 2009.
[17] R. Sakai, K. Ohgishi, and M. Kasahara, "Cryptosystems based on pairing," SCIS 2000, 2000.
[18] Y. Sakemi, Y. Nogami, K. Okeya, H. Kato, and Y. Morikawa, "Skew frobenius map and efficient scalar multiplication for pairing-based cryptography," CANS 2008, LNCS, vol.5339, pp.226–239, Springer, 2008.
[19] M. Scott, "Faster pairings using an elliptic curve with an efficient endomorphism," INDOCRYPT2005, LNCS, vol.3797, pp.258–269, Springer, 2005.
[20] Victor Shoup, NTL: A library for doing number theory, Available at http://www.shoup.net/ntl/
[21] C.-A. Zhao, F. Zhang, and J. Huang "A note on the Ate pairing," Int. J. Information Security, vol.6, no.7, pp.379–382, 2008.
[22] F. Vercauteren, "Optimal pairings," IEEE Trans. Inf. Theory, vol.56, no.1, pp.455–461, 2010.
[23] L.C. Washington, Elliptic Curves, Number theory and cryptography, Chapman & Hall/CRC, 2003.

Appendix A: Experimental Parameters

A.1 Example 6

Here, we show the base points for the experiments in Sect. 3.4.

$P = (x_P, y_P) \in \mathbb{G}_1$,

$$\begin{aligned} x_P &= \texttt{0x0f720233 9155123c a8cab4c2 e0f6b61e\backslash} \\ &\quad \texttt{d52126cf aee68927 4e8894e9 41374038\backslash} \\ &\quad \texttt{27176d3d 845456ae}, \end{aligned}$$

$$\begin{aligned} y_P &= \texttt{0x1650bc43 b01e0363 af20a1b5 f0ecad0b\backslash} \\ &\quad \texttt{ae13a57f b17d9eb3 e566a878 66ffa16e\backslash} \\ &\quad \texttt{002e58c1 392c7565}, \end{aligned}$$

$Q = (x_Q, y_Q) \in \mathbb{G}_2$,

$$\begin{aligned} x_Q &= (\texttt{0x0cffa802 8cb3ba43 93c6463c c669c129\backslash} \\ &\quad \texttt{16cd9418 70114fa8 0917b9ab a3a30943\backslash} \\ &\quad \texttt{e1d85445 59522267})Z^2 + \\ &\quad (\texttt{0x17fac36d 36bd0394 91b9fd1c 2c1f3323\backslash} \\ &\quad \texttt{ae693a89 734288cf c6f4275a d1662bad\backslash} \\ &\quad \texttt{b3bbe232 455eb74d})Z^6 + \\ &\quad (\texttt{0x1b7a7017 60f78fe9 e9f66ed9 3f70ab53\backslash} \\ &\quad \texttt{0f6021a0 c25ab9eb 89f5ca7c b548dcc5\backslash} \\ &\quad \texttt{175dbae6 773e100a})Z^{10} + \\ &\quad (\texttt{0x14300d3b b0428ae9 07e65870 c0a57066\backslash} \\ &\quad \texttt{4fbc3714 7f0aec54 387d9982 3f7aa0fc\backslash} \\ &\quad \texttt{8b1fc5fb 914f4123})Z^{14}, \end{aligned}$$

$$\begin{aligned} y_Q &= (\texttt{0x18827b34 b8692150 8639514e 35326fcf\backslash} \\ &\quad \texttt{36512749 67c8aaf5 9a9b5b75 247d5103\backslash} \\ &\quad \texttt{528960f9 f630bbda})Z^3 + \\ &\quad (\texttt{0x150743ec 414d19b3 0540624c ba6579f4\backslash} \\ &\quad \texttt{211e86bb ecb764b5 85b7cf08 8fe8fb6c\backslash} \\ &\quad \texttt{0e0ae5b0 5eca3f05})Z^7 + \\ &\quad (\texttt{0x0f56e12b 6329c6f9 ba84115e 835d645d\backslash} \\ &\quad \texttt{8f7b6ba2 9a9b7365 e3dbb183 fc2e769d\backslash} \\ &\quad \texttt{524b782b c7207eee})Z^{11} + \\ &\quad (\texttt{0x047f969a 789264cd 1253852e cfe49606\backslash} \\ &\quad \texttt{d0bb31d4 f131e6c1 9542bd54 e79afc27\backslash} \\ &\quad \texttt{4741fffb 23babab7})Z^{15}. \end{aligned}$$

Appendix B: Overview of Ate$_i$ and Optimal Pairings

B.1 Tate Pairing

Let $P \in E(\mathbb{F}_{q^k})[r] := \{P_0 \in E(\mathbb{F}_{q^k}) : [r]P_0 = O\}$ and $Q \in E(\mathbb{F}_{q^k})$. Choose a point $R \in E(\mathbb{F}_{q^k})$ such that $D := (Q + R) - (R)$ and $r(P) - r(O)$ have disjoint support. Then, the (reduced-)Tate pairing is defined by

$$\begin{aligned} &\tau : E(\mathbb{F}_{q^k})[r] \times E(\mathbb{F}_{q^k})/rE(\mathbb{F}_{q^k}) \to \mu_r, \\ &\quad \tau(P, Q) = f_{r,P}(D)^{(q^k-1)/r}, \end{aligned}$$

where $f_{r,P}$ is a rational function with $\mathrm{div}(f_{r,P}) = r(P) - r(O)$. The (reduced-)Tate pairing $\tau(P, Q)$ is bilinear and non-degenerate.

Here, we briefly review Ate$_i$ pairing and optimal pairing. We use these concepts to reduce Miller's loop for scalar multiplication.

B.2 Ate$_i$ Pairing

The Ate$_i$ pairing proposed by [21] is an improvement of the original Ate pairing [11]. Let $T_i := q^i \pmod r$ for $i = 1, 2, \cdots, k-1$. For each i, we define the following quantities in a manner similar to that for the Ate pairing [11], where a_i is the smallest positive integer such that $T_i^{a_i} \equiv 1 \pmod r$. In addition, $N_i := \gcd(T_i^{a_i} - 1, q^k - 1)$, and L_i is a positive integer such that $T_i^{a_i} - 1 = L_i N_i$. The Ate$_i$ pairing (on $\mathbb{G}_2 \times \mathbb{G}_1$) is defined by $f_{T_i,Q}(P)$ for $Q \in \mathbb{G}_2$ and $P \in \mathbb{G}_1$.

The Ate$_i$ pairing on $\mathbb{G}_1 \times \mathbb{G}_2$ is defined by $f_{T_i,P}(Q)$ ($P \in$

$\mathbb{G}_1$ and $Q \in \mathbb{G}_2$) for supersingular elliptic curves. When E is ordinary, we need to use the twist of E in order to define the Ate pairing on $\mathbb{G}_1 \times \mathbb{G}_2$.

Let E and E' be ordinary elliptic curves over $\mathbb{F}_q$. We refer to the curve E' as a twist of degree d of E if there exists an isomorphism $\psi : E' \to E$ defined over $\mathbb{F}_{q^d}$ and d is minimal with this property. We hereinafter consider $\mathbb{F}_q$ with characteristic $p \geq 5$. Then, only $d = 2, 3, 4, 6$ are possible (see [11] for explicit forms of twists of elliptic curves with characteristic $p \geq 5$).

We next define the Ate$_i$ pairing on $\mathbb{G}_1 \times \mathbb{G}_2$ for ordinary elliptic curves. Let $e := k/\gcd(k, d)$ and $S_{e,i} := T_i^e \pmod r = q^{ie} \pmod r$ for an ordinary elliptic curve E with embedding degree k. Then, the Ate$_i$ pairing on $\mathbb{G}_1 \times \mathbb{G}_2$, which is referred to as the twisted Ate$_i$ pairing, is defined by $f_{S_{e,i},P}(Q)$ ($e = k/m$, $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$) and $f_{S_{e,i},P}(\psi(Q'))$ ($P \in \mathbb{G}_1$ and $Q' \in \psi^{-1}(\mathbb{G}_2)$).

The Ate$_i$ and twisted Ate$_i$ pairings have an exponential relationship with the Tate pairing (see [21]).

The length of Miller's loop for computing $f_{T_i,Q}(P)$ is $\log_2 |T_i|$. If $T_n := \min\{T_i : i = 1, 2, \cdots, k-1, 0 \leq T_i \leq r-1\}$. Then, $f_{T_n,Q}(P)$ can be computed faster than the Ate pairing $f_{T,Q}(P)$.

B.3 Optimal Pairing

Optimal pairing is defined by Vercauteren [22] as a pairing that can be computed in $\log_2 r/\phi(k) + \epsilon(k)$ Miller loop iterations ($\phi(k)$ is the Euler function of k, and $\epsilon(k) \leq \log_2 k$). Vercauteren [22] presented a method of computing optimal pairing:

Theorem 2 ([22] Theorem 1.) *Let $\lambda = mr$, where $r \nmid m$. We express λ as $\lambda = \sum_{i=0}^{l} c_i q^i$. Then,*

$$a_{[c_0,c_1,\cdots,c_l]} : \mathbb{G}_2 \times \mathbb{G}_1 \to \mu_r$$

$$(Q, P) \mapsto \left(\prod_{i=0}^{l} f_{c_i,Q}^{q^i}(P) \cdot \prod_{i=0}^{l-1} \frac{l_{[s_{i+1}]Q,[c_i q^i]Q}(P)}{v_{[s_i]Q}(P)} \right)^{\frac{q^k-1}{r}},$$

where $s_i = \sum_{j=i}^{l} c_j q^j$ defines a bilinear map. Furthermore, if

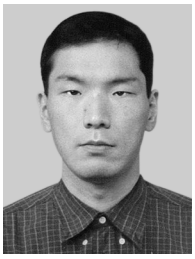
$$mkq^{k-1} \not\equiv \frac{q^k - 1}{r} \sum_{i=0}^{l} i c_i q^{i-1} \pmod r,$$

then $a_{[c_0,c_1,\cdots,c_l]}(Q, P)$ is non-degenerate.

Note that we may consider $l = \phi(k) - 1$ because $r \mid \Phi_k(q)$, where $\Phi_k(X)$ is the k-th cyclotomic polynomial.

Remark 3 If we apply optimal pairing concepts to twisted Ate pairings, then we obtain $a^{\text{twist}}_{[c_0,c_1,\ldots,c_l]} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mu_r$, as follows:

If we use the degree d twist E' of E defined over $\mathbb{F}_{q^e}$, where $e = k/\gcd(k, d)$, in Theorem 2, then we replace q by q^e and exchange P with Q. Then, there exists $[\xi_d] \in \mathrm{Aut}(E)$ defined over $\mathbb{F}_q$ such that $[\xi_d]P = [q^e]P$ for all $P \in \mathbb{G}_1$.



Naoki Kanayama received his B.E., B.S., M.S. and D.S. degrees from Waseda University, Tokyo, Japan, in 1994, 1996, 1998 and 2003, respectively. In 2003–2006, he was a post-doctoral fellow of the Japan Society for the Promotion of Science. From 2006, he is a research fellow at University of Tsukuba. Dr. Kanayama is a member of the Japan Society for Industrial and Applied Mathematics.

Tadanori Teruya received his B.S. and M.E. degrees from University of Tsukuba in 2007 and 2009, respectively. His current research interests include cryptography and information security.

Eiji Okamoto received his B.S., M.S. and Ph.D. degrees in electronics engineering from the Tokyo Institute of Technology in 1973, 1975 and 1978, respectively. He worked and studied communication theory and cryptography for NEC central research laboratories since 1978. From 1991 he became a professor at Japan Advanced Institute of Science and Technology, then at Toho University. Now he is a professor at Graduate School of Systems and Information Engineering, University of Tsukuba. His research interests are cryptography and information security. He is a coeditor-in-chief of International Journal of Information Security.