Pseudo 8–Sparse Multiplication for Efficient Ate–Based Pairing on Barreto–Naehrig Curve

Yuki Mori[1], Shoichi Akagi[1], Yasuyuki Nogami[1], and Masaaki Shirase[2]

[1] Graduate School of Natural Science and Technology, Okayama University
3-1-1, Tsushima-naka, Okayama, Okayama 700-8530, Japan
[2] Future University Hakodate, Japan
yasuyuki.nogami@okayama-u.ac.jp

Abstract. According to some recent implementation reports on Ate–based pairings such as optimal ate pairing with Barreto–Naehrig curve whose embedding degree is 12, *sparse multiplication* accelerates Miller's loop calculation in a pairing calculation. Especially, 7–sparse multiplication is available when the implementation uses affine coordinates, where 7–sparse means that the multiplicand or multiplier has 7 zeros among 12 coefficients. This paper extends it to *pseudo* 8*–sparse multiplication*. Then, some experimental results together with theoretic calculation costs are shown in order to evaluate its efficiency.

Keywords: sparse multiplication, pairing, Barreto–Naehrig curve.

1 Introduction

Recent Ate–based pairings such as R–ate [1], Optimal ate [2] and Xate [3] on Barreto–Naehrig (BN) curve have received much attention since they achieve quite efficient pairing calculations. Then, many researchers have tried to implement these Ate–based pairings as thoroughly efficient programs using mathematic and programmatic techniques such as Montgomery reduction (Montgomery representation), lazy reduction, Projective/Jacobian coordinates, sparse multiplication, and final exponentiation with Gröbner basis. Among these techniques, this paper focuses on *sparse multiplication*. Note here that pairings on BN curve are defined over $\mathbb{F}_{q^{12}}$ since the embedding degree of BN curve is 12, where q denotes the field characteristic throughout this paper.

Aranha et al. [4] and Grewal et al. [5] have well introduced the preceding techniques. According to their works, 6–sparse multiplication[1] with *projective coordinates* accelerates Miller's loop calculation that is a major calculation part together with *final exponentiation*. They have also introduced 7–sparse multiplication with *affine coordinates*. It seems that, from the viewpoint of efficiency, 7–sparse multiplication is better than 6–sparse multiplication though the difference of the adapted coordinates should be carefully taken into account. This paper proposes a more efficient sparse multiplication.

[1] It means that the multiplier/multiplicand has 6 *zeros* among 12 vector coefficients.

Z. Cao and F. Zhang (Eds.): Pairing 2013, LNCS 8365, pp. 186–198, 2014.

This paper first focuses on the fact that multiplying/dividing the result of Miller's loop calculation by an arbitrary non–zero element in $\mathbb{F}_q$ does not change the result of the pairing because of the following *final exponentiation*. Based on this fact, this paper achieves *pseudo* 8–sparse multiplication by dividing one of non–zero coefficients of the preceding 7–sparse multiplier with affine coordinates. According to the division, one of 5 non–zero coefficients becomes one and thus it contributes to a calculation efficiency. After that, in order to cancel the calculation overhead caused from the division, this paper applies *isomorphic* twist with a quadratic and cubic residue in $\mathbb{F}_q$, where note that *sextic* twist with a quadratic and cubic *non* residue in $\mathbb{F}_{q^2}$ is available for BN curves. Then, in order to evaluate the efficiency of *pseudo* 8–sparse multiplication, this paper shows some experimental results together with theoretic calculation costs.

Throughout this paper, $\mathbb{F}_q$ and $\mathbb{F}_{q^m}$ denote a prime field of characteristic q and its m–th extension field, respectively.

2 Preliminaries

This section briefly reviews Barreto–Naehrig (BN) curve [6], towering extension field with irreducible binomials [4], sextic twist [3], Ate pairing, and sparse multiplication (7–sparse multiplication) appeared in Miller's loop [4].

2.1 Barreto–Naehrig Curve

Barreto–Naehrig curve [7] that is well known to realize an efficient *asymmetric* pairing is defined in the form of

$$E : y^2 = x^3 + b, \quad b \in \mathbb{F}_q, \tag{1}$$

together with the following parameter settings,

$$q(\chi) = 36\chi^4 - 36\chi^3 + 24\chi^2 - 6\chi + 1, \tag{2a}$$
$$r(\chi) = 36\chi^4 - 36\chi^3 + 18\chi^2 - 6\chi + 1, \tag{2b}$$
$$t(\chi) = 6\chi^2 + 1, \tag{2c}$$

where χ is a certain integer[2]. This paper focuses on recent efficient Ate–based pairings such as optimal ate [2], R–ate [1], and Xate [3] pairings on BN curve.

Towering Extension Field with Irreducible Binomials $\mathbb{F}_{((q^2)^3)^2}$
In what follows, let $q - 1$ be divisible by 4 and c be a cubic and quadratic non residue in $\mathbb{F}_q$. Then, $\mathbb{F}_{q^{12}}$ is constructed as a tower field in the following representations.

[2] There are some conditions such as q to be a prime number for defining $\mathbb{F}_q$.

$$\begin{cases} \mathbb{F}_{q^2} = \mathbb{F}_q[i]/(i^2 - \beta), \text{ where } \beta = c. \\ \mathbb{F}_{q^6} = \mathbb{F}_{q^2}[v]/(v^3 - \xi), \text{ where } \xi = i. \\ \mathbb{F}_{q^{12}} = \mathbb{F}_{q^6}[w]/(w^2 - v). \end{cases} \tag{3}$$

According to most of previous works such as Aranha et al. [4], the above v is used for the following *sextic twist* of BN curve.

Sextic Twist. For BN curve E defined above, *sextic twisted curve* E' together with a certain quadratic and cubic non residue $z \in \mathbb{F}_{q^2}$ and an isomorphic mapping ψ_6 are given as follows [3].

$$\begin{aligned} E' :\ & y^2 = x^3 + bz, \\ \psi_6 :\ & E'(\mathbb{F}_{q^2})[r] \longmapsto E(\mathbb{F}_{q^{12}})[r] \cap \mathrm{Ker}(\pi_q - [q]), \\ & (x, y) \longmapsto (z^{-1/3}x, z^{-1/2}y). \end{aligned} \tag{4}$$

where $\mathrm{Ker}(\cdot)$ and π_q respectively denote the kernel of the mapping $\cdot$ and Frobenius mapping for rational point as

$$\pi_q : (x, y) \longmapsto (x^q, y^q). \tag{5}$$

In addition, its order $\#E'(\mathbb{F}_{q^2})$ is also divisible by r that is the order of BN curve E over $\mathbb{F}_q$. Thus, some efficient pairings [4] have made the best use of the sextic twisted *subfield* curve $E'(\mathbb{F}_{q^2})$ based on the isomorphic twist. In this paper, $E'(\mathbb{F}_{q^2})[r]$ shown in Eq. (4) is denoted by $\mathbb{G}'_2$ such as shown in Alg. 1.

When $\hat{z}$ is a Quadratic and Cubic Residue in $\mathbb{F}_q$
Consider the following curve $\hat{E}(\mathbb{F}_q)$ and mapping.

$$\begin{aligned} \hat{E} :\ & y^2 = x^3 + b\hat{z}, \\ & \hat{E}(\mathbb{F}_q)[r] \longmapsto E(\mathbb{F}_q)[r], \\ & (x, y) \longmapsto (\hat{z}^{-1/3}x, \hat{z}^{-1/2}y), \\ & \text{where } \hat{z}, \hat{z}^{-1/2}, \hat{z}^{-1/3} \in \mathbb{F}_q. \end{aligned} \tag{6}$$

Throughout this paper, it should be carefully noted that $E(\mathbb{F}_q)$ and $\hat{E}(\mathbb{F}_q)$ are isomorphic[3] since $\hat{z}$ is a quadratic and cubic *residue* in $\mathbb{F}_q$.

[3] $E(\mathbb{F}_{q^2})$ and $\hat{E}(\mathbb{F}_{q^2})$ furthermore $E(\mathbb{F}_{q^{12}})$ and $\hat{E}(\mathbb{F}_{q^{12}})$ are also isomorphic.

2.2 Pairings

In what follows, let the embedding degree be k. For example, $k = 12$ in the case of BN curve. As previously introduced, this paper focuses on Ate–based pairings.

Ate Pairing. Suppose the following two groups and Ate pairing notation.

$$\mathbb{G}_1 = E(\mathbb{F}_{q^k})[r] \cap \mathrm{Ker}(\pi_q - [1]),$$
$$\mathbb{G}_2 = E(\mathbb{F}_{q^k})[r] \cap \mathrm{Ker}(\pi_q - [q]),$$

$$\alpha : \ \mathbb{G}_2 \times \mathbb{G}_1 \to \mathbb{F}^*_{q^k}/(\mathbb{F}^*_{q^k})^r. \tag{7}$$

In the case of BN curve, the above $\mathbb{G}_1$ is just $E(\mathbb{F}_q)$. Then, let $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$, Ate pairing $\alpha(Q, P)$ is given as follows.

$$\alpha(Q, P) = f_{t-1,Q}(P)^{\frac{q^k-1}{r}}, \tag{8}$$

where $f_{t-1,Q}(P)$ is the output of Miller's algorithm. After calculating the *final exponentiation*, the bilinearity of Ate pairing holds.

In the case of Xate pairing $\zeta(Q, P)$ on BN curve defined by

$$\zeta(Q, P) = \Big\{ f_{\chi,Q}(P)^{(1+q^3)(1+q^{10})} \cdot l_{\chi Q, \pi_q^3(\chi Q)}(P) \cdot l_{\chi Q + \pi_q^3(\chi Q), \pi_q^{10}(\chi Q + \pi_q^3(\chi Q))}(P) \Big\}^{\frac{q^k-1}{r}}. \tag{9}$$

where χ is the setting integer parameter shown at Eqs. (2), the calculation procedure becomes as shown in Alg. 1. In what follows, the calculation steps from 1 to 6 shown in Alg. 1 is called Miller's loop. In addition, it is found that steps 3 and 5 in Alg. 1 are key to accelerating a pairing calculation. As one of such accelerating techniques, *sparse multiplication* has been introduced and thus a lot of related works have been reported [4], [5].

7–sparse Multiplication in Miller's Loop on Affine Coordinates
According to Grewal et al.'s work [5], in the case of adapting affine coordinates for representing rational points, the *doubling* phase (step 3) and *addition* phase (step 5) in Miller's loop are efficiently carried out by the following calculations. In what follows, let $P = (x_P, y_P) \in E(\mathbb{F}_q)$, $T = (x, y)$, and $Q = (x_2, y_2) \in E'(\mathbb{F}_{q^2})$ be given in affine coordinates, and let $T + Q = (x_3, y_3)$ be the sum of T and Q.

***Doubling* phase (when $T = Q$)**

$$A = \frac{1}{2y},\ B = 3x^2,\ C = AB,\ D = 2x,\ x_3 = C^2 - D,$$
$$E = Cx - y,\ y_3 = E - Cx_3,\ F = C\bar{x}_P,$$

$$l_{T,T}(P) = y_P + Fw + Ew^3 = y_P - Cx_P w + Ew^3, \tag{10a}$$

where $\bar{x}_P = -x_P$ will be precomputed. □

Algorithm 1. Xate pairing on BN curves (generalized for $\chi < 0$)

Input: $P \in \mathbb{G}_1,\ Q \in \mathbb{G}'_2,\ \chi$
Output: $\zeta(Q, P)$
1 $T \leftarrow Q, f \leftarrow 1$
2 **for** $i = \lfloor \log_2(|\chi|) \rfloor - 1$ **downto** 0 **do**
3 $\quad f \leftarrow f^2 \cdot l_{T,T}(P),\ \ T \leftarrow 2T;$ (*see* Doubling phase Eq. (10a))
4 $\quad$ **if** $|\chi|_i = 1$ **then**
5 $\quad\quad f \leftarrow f \cdot l_{T,Q}(P),\ \ T \leftarrow T + Q;$ (*see* Addition phase Eq. (10b))
6 $\quad$ **if** $|\chi|_i = -1$ **then**
7 $\quad\quad f \leftarrow f \cdot l_{T,-Q}(P),\ \ T \leftarrow T - Q;$ (*see* Addition phase Eq. (10b))
8 **end for**
9 **if** $\chi < 0$ **then**
10 $\quad T \leftarrow -T, f \leftarrow f^{-1}$
11 $f \leftarrow f \cdot \pi_q^3(f),\ \ Q_1 \leftarrow \pi_q^3(T)$
12 $f \leftarrow f \cdot l_{T,Q_1}(P),\ \ Q_2 \leftarrow T + Q_1$
13 $f \leftarrow f \cdot \pi_q^{10}(f),\ \ T \leftarrow \pi_q^{10}(Q_2)$
14 $f \leftarrow f \cdot l_{T,Q_2}(P)$
15 $f \leftarrow \mathrm{FinalExp}(f) (= f \leftarrow f^{(q^k-1)/r})$
16 **return** f

***Addition* phase (when $T \neq Q$)**

$$A = \frac{1}{x_2 - x},\ B = y_2 - y,\ C = AB,\ D = x + x_2,\ x_3 = C^2 - D,$$

$$E = Cx - y,\ y_3 = E - Cx_3,\ F = C\bar{x}_P,$$

$$l_{T,Q}(P) = y_P + Fw + Ew^3 = y_P - Cx_P w + Ew^3, \tag{10b}$$

where $\bar{x}_P = -x_P$ will be precomputed. □

As shown in Eqs. (10), since 1, w, and $w^3 = vw$ are basis elements of $\mathbb{F}_{q^{12}}$ for $\mathbb{F}_{q^2}$ as previously introduced, it is found that 7 coefficients among 12 of the vector representation of $l_{\psi_6(T),\psi_6(T)}(P) \in \mathbb{F}_{q^{12}}$ are equal to zero at least. In other words, only 5 coefficients $y_P \in \mathbb{F}_q$, $Cx_P \in \mathbb{F}_{q^2}$, and $E \in \mathbb{F}_{q^2}$ are possible to be non–zero. $l_{\psi_6(T),\psi_6(Q)}(P)$ also has the same property. Thus, the calculation of multiplying $l_{\psi_6(T),\psi_6(T)}(P)$ or $l_{\psi_6(T),\psi_6(Q)}(P)$ is called *sparse multiplication*, in this case especially 7–sparse multiplication, that accelerates Miller's loop calculation as shown in Alg. 1. This paper proposes *pseudo* 8–sparse multiplication.

3 Main Proposal

This paper proposes the following two ideas in order to realize an efficient 8–sparse multiplication for Ate–based pairing on BN curve such as Ate, optimal ate [2], R–ate [1], and Xate [3] pairings.

1. As shown in Eqs. (10), one of non–zero coefficients is $y_P \in \mathbb{F}_q$. This coefficient does not change through Miller's loop calculation. Thus, dividing both sides of those equations by y_p, the coefficient becomes 1. It leads to a more efficient sparse multiplication by $l_{\psi_6(T),\psi_6(T)}(P)$ or $l_{\psi_6(T),\psi_6(Q)}(P)$. In this paper, it is called *pseudo* 8*–sparse multiplication.*
2. The above division by y_P causes a little more calculation cost for the other non–zero coefficients in the Miller's loop as it is. Applying the map introduced in Eqs. (6), such an additional cost in Miller's loop is canceled.

As shown in Eq. (10a) and Eq. (10b), they are basically the same. Thus, using Eq. (10a) in what follows, these ideas are introduced in detail.

3.1 Pseudo 8–Sparse Multiplication

Note that y_P shown in Eq. (10a) is an non–zero[4] element in $\mathbb{F}_q$. Thus, dividing both sides of Eq. (10a) by y_P,

$$y_P^{-1} l_{T,T}(P) = 1 - C(x_P y_P^{-1})w + E y_P^{-1} w^3. \tag{11}$$

Even if replacing $l_{T,T}(P)$ by the above $y_P^{-1} l_{T,T}(P)$, the calculation result of the pairing does not change because *final exponentiation* cancels $y_P^{-1} \in \mathbb{F}_p$. Then, as shown above, one of the non–zero coefficients becomes 1 and it realizes more efficient vector multiplications in Miller's loop. This paper calls it *pseudo* 8–sparse multiplication. The detailed calculation procedure of pseudo 8–sparse multiplication is introduced in **App**. A.

3.2 Line Evaluation in Miller's Loop

Comparing the line evaluations Eq. (10a) and Eq. (11), it is found that the latter needs a little more calculation cost for Ey_P^{-1} even though $x_P y_P^{-1}$ and y_P^{-1} can be precomputed. In what follows, an approach to cancel $x_P y_P^{-1}$ is introduced.

In brief, based on $P(x_P, y_P)$, the map introduced in Eqs. (6) can find a certain isomorphic rational point $\hat{P}(x_{\hat{P}}, y_{\hat{P}}) \in \hat{E}(\mathbb{F}_q)$ such that

$$x_{\hat{P}} y_{\hat{P}}^{-1} = 1 \tag{12}$$

by letting the twist parameter z of Eq. (4) be $\hat{z} = (x_P y_P^{-1})^6$ of Eqs. (6), where $\hat{E}$ denotes the BN curve defined by Eqs. (6). Of course, this $\hat{z}$ is a quadratic and

[4] $P(x_P, y_P) \in E(\mathbb{F}_q)$ for pairing on BN curve is selected such that $x_P \neq 0$ and $y_P \neq 0$.

cubic residue in $\mathbb{F}_p$ and thus it yields the map. According to Eq. (4), such z is obtained by solving the following equation from the input $P(x_P, y_P)$.

$$z^{1/3} x_P = z^{1/2} y_P. \tag{13}$$

Then, $\hat{P}(x_{\hat{P}}, y_{\hat{P}}) \in \hat{E}(\mathbb{F}_q)$ is given by

$$\hat{P}(x_{\hat{P}}, y_{\hat{P}}) = (x_P^3 y_P^{-2}, x_P^3 y_P^{-2}). \tag{14}$$

Since the x and y coordinates of $\hat{P}$ are the same, $x_{\hat{P}} y_{\hat{P}}^{-1} = 1$. Therefore, corresponding to the the map introduced in Eqs. (6), first mapping not only P to $\hat{P}$ shown above but also Q to $\hat{Q}$ shown below,

$$\hat{Q}(x_{\hat{Q}}, y_{\hat{Q}}) = (x_P^2 y_P^{-2} x_Q, x_P^3 y_P^{-3} y_Q). \tag{15}$$

the line evaluations Eq. (10a) becomes

$$\begin{aligned} \hat{l}_{\hat{T},\hat{T}}(\hat{P}) = y_{\hat{P}}^{-1} l_{\hat{T},\hat{T}}(\hat{P}) &= 1 - C(x_{\hat{P}} y_{\hat{P}}^{-1}) w + E y_{\hat{P}}^{-1} w^3 \\ &= 1 - Cw + E(x_P^{-3} y_P^2) w^3. \end{aligned} \tag{16}$$

Eq. (10b) becomes the same. Compared to Eq. (11), the second term of the right–hand side has become simple because $x_{\hat{P}} y_{\hat{P}}^{-1} = 1$.

Computing $\hat{P}$, $\hat{Q}$, and $x_P^{-3} y_P^2$ using x_P^{-1} and y_P^{-1} will be an overhead; however, Miller's loop calculation becomes efficient together with pseudo 8–sparse multiplication. Alg. 2 shows the proposed algorithm for which x_P^{-1} and y_P^{-1} thus need to be once calculated[5].

4 Cost Evaluation and Experimental Result

In order to show the efficiency of the proposal, this section shows some experimental results with evaluating the calculation costs.

In what follows, "Grewal's work" means optimal ate pairing with affine coordinates and 7–sparse multiplication (*see* the detail [5]). "This work" means Xate pairing with affine coordinates and 8–sparse multiplication.

4.1 Parameter Settings and Computational Environment

This paper has set the following parameters (*see* Sec. 2.1).

$$\begin{aligned} \chi &= -4611686018425225214, && \text{(17a)} \\ &= -2^{62} + 2^{21} + 2^{16} + 2, \\ &\quad \text{where } r(\chi) \text{ becomes a 254–bit prime,} \\ b &= c = 2, && \text{(17b)} \\ z &= i^{-1}. && \text{(17c)} \end{aligned}$$

[5] They are obtained by one $\mathbb{F}_q$–inversion using Montgomery trick.

Algorithm 2. Proposed Xate pairing on BN curves (generalized for $\chi < 0$)

Input: $P(x_P, y_P) \in \mathbb{G}_1,\ Q(x_Q, y_Q) \in \mathbb{G}'_2,\ \chi$
Output: $\zeta(Q, P)$
1 Compute x_P^{-1} and y_P^{-1} ; (they are used at steps 3 and 4)
2 Compute $x_P^{-3} y_P^2$; (it is used at steps 7 and 9 with Eq. (16))
3 $\hat{P} \leftarrow \text{Mapping}(P)$; (*see* Eq. (14))
4 $\hat{Q} \leftarrow \text{Mapping}(Q)$; (*see* Eq. (15))
5 $\hat{T} \leftarrow \hat{Q}, f \leftarrow 1$
6 **for** $i = \lfloor \log_2(|\chi|) \rfloor - 1$ **downto** 0 **do**
7 $\quad f \leftarrow f^2 \cdot \hat{l}_{\hat{T},\hat{T}}(\hat{P}), \hat{T} \leftarrow 2\hat{T}$; (*see* Eq. (16))
8 $\quad$ **if** $|\chi|_i = 1$ **then**
9 $\quad\quad f \leftarrow f \cdot \hat{l}_{\hat{T},\hat{Q}}(\hat{P}),\ \hat{T} \leftarrow \hat{T} + \hat{Q}$; (*see* Eq. (16))
10 $\quad$ **if** $|\chi|_i = -1$ **then**
11 $\quad\quad f \leftarrow f \cdot \hat{l}_{\hat{T},-\hat{Q}}(\hat{P}),\ \hat{T} \leftarrow \hat{T} - \hat{Q}$; (*see* Eq. (16))
12 **end for**
13 **if** $\chi < 0$ **then**
14 $\quad \hat{T} \leftarrow -\hat{T},\ f \leftarrow f^{-1}$
15 $f \leftarrow f \cdot \pi_q^3(f),\ \hat{Q}_1 \leftarrow \pi_q^3(\hat{T})$
16 $f \leftarrow f \cdot \hat{l}_{\hat{T},\hat{Q}_1}(\hat{P}),\ \hat{Q}_2 \leftarrow \hat{T} + \hat{Q}_1$; (*see* Eq. (16))
17 $f \leftarrow f \cdot \pi_q^{10}(f),\ \hat{T} \leftarrow \pi_q^{10}(\hat{Q}_2)$
18 $f \leftarrow f \cdot \hat{l}_{\hat{T},\hat{Q}_2}(\hat{P})$; (*see* Eq. (16))
19 $f \leftarrow \text{FinalExp}(f) (= f \leftarrow f^{(q^k-1)/r})$
20 **return** f

Table 1 shows the computational environments.

Table 1. Computing environment

	PC	iPad2	iPhone5
CPU	Core 2 Duo* E8135 2.66GHz	Apple A5* 1.0GHz	Apple A6* 1.3GHz
OS	Mac OS X 10.7.2	iOS 6.1.3	iOS 6.1.4
Library	GMP 5.1.2	gmp4osx (GMP 5.0.5)	gmp4osx (GMP 5.0.5)
Compiler	g++ 4.2.1	g++ 4.2.1	g++ 4.2.1
Programming Language	C++	C++ and Objective-C	C++ and Objective-C

∗ Only single core is used though it has two cores.

4.2 Cost Evaluation

In the same manner of Aranha et al. [4] and Grewal et al. [5], this paper uses the following notations for evaluating the calculation costs. Thus, the following paragraph is almost the same of that of Grewal et al.'s [5].

Notation and Definitions (*see also* Grewal et al.'s instruction [5]) Throughout this paper, lower case variables denote single–precision integers, upper case variables denote double–precision integers. The operation + represents addition without reduction, and $\oplus$ represents addition with reduction (*see* Alg. alg:sparse). The quantities m, s, a, i and r denote the times for multiplication, squaring, addition, inversion, and modular reduction in $\mathbb{F}_q$, respectively. Likewise, $\tilde{m}, \tilde{s}, \tilde{a}, \tilde{i}$ and $\tilde{r}$ denote the times for multiplication, squaring, addition, inversion, and reduction in $\mathbb{F}_{q^2}$, respectively, and $m_u, s_u, \tilde{m}_u$ and $\tilde{s}_u$ denote the times for multiplication and squaring without reduction in the corresponding fields. Finally, m_β and m_ξ m_v denote the times for multiplication by the quantities β and ξ, respectively (*see* the preceding *towering extension field*).

First, Table 2 shows the calculation costs for the arithmetics in $E'(\mathbb{F}_{q^2})$, $\mathbb{F}_{q^2}$, and $\mathbb{F}_{q^{12}}$. Since their constructions are slightly different though both are based on *towering extension field* technique, the calculation costs are slightly different. Basically, the number of multiplications such as m and $\tilde{m}_u$ are the same though those of additions such as a and $\tilde{a}$ are different; however, 7–sparse multiplication and *pseudo* 8–sparse multiplication have the difference of $6m_u$. It leads to the main contribution of this paper.

Based on these fundamental arithmetics, Table 3 shows the calculation costs for pairings by Grewal et al.'s work and this paper in which that of final exponentiation is excluded[6]. Instead of 7–sparse multiplication, pseudo 8–sparse multiplication is applied 66 times in Xate pairing calculation excluding final exponentiation. Thus, as shown in Table 3, the difference of $66 \times 6m_u = 396m_u$ has occurred between the pairings excluding final exponentiation. According to the calculation costs of pairings, it is found that pseudo 8–sparse multiplication has reduced a few hundreds of m_u's. For iPad 2 and iPhone 5, since the relation of $69\tilde{i} + 204a + s + 2i \leq 178\tilde{m}_u + 326\tilde{s} + 229\tilde{r} + 2056\tilde{a} + 131m$, This work is faster than the Xate pairing using projective coordinates.

4.3 Experimental Result

Table 4 shows the calculation times of Xate pairing including(excluding) final exponentiation. They are the averages of 100,000 and 9,000 iterations of pairing on PC and iOS devices (iPad 2 and iPhone 5), respectively. According to the experimental results, pseudo 8–sparse multiplication contributes to a few percent acceleration of Previous work, which the Xate pairing uses affine coordinates and uses 7–sparse multiplication. It seems to be very small but makes the recent marvelous implementations of pairing [4], [5] a little more efficient.

[6] Because the calculation cost of final exponentiation is almost the same.

Table 2. Operation counts for 254-bit prime fields

$E'(\mathbb{F}_{q^2})$ Arithmetics	Grewal's work [5]	This work
Doubling/Line Evaluation	$\tilde{i} + 3\tilde{m}_u + 2\tilde{s}_u + 5\tilde{r} + 7\tilde{a} + 2m$	$\tilde{i} + 3\tilde{m}_u + 2\tilde{s}_u + 5\tilde{r} + 7\tilde{a} + 2m$
Addition/Line Evaluation	$\tilde{i} + 3\tilde{m}_u + 1\tilde{s}_u + 4\tilde{r} + 6\tilde{a} + 2m$	$\tilde{i} + 3\tilde{m}_u + 1\tilde{s}_u + 4\tilde{r} + 6\tilde{a} + 2m$
q–power Frobenius	$2\tilde{m} + 2a$	–
q^2–power Frobenius	$4m$	–
q^3–power Frobenius	–	$4a + 2m$
q^{10}–power Frobenius	–	$2a + 2m$
$\mathbb{F}_{q^2}$ Arithmetics	Grewal's work [5]	This work
Add/Subtr./Nega.	$\tilde{a} = 2a$	$\tilde{a} = 2a$
Multiplication	$\tilde{m} = 3m_u + 2r + 8a$	$\tilde{m} = 3m_u + 2r + 8a$
Squaring	$\tilde{s} = 2m_u + 2r + 3a$	$\tilde{s} = 2m_u + 2r + 6a$
Multiplication by β	$m_\beta = a$	$m_\beta = a$
Multiplication by ξ	$m_\xi = 2a$	$m_\xi = a$
$\mathbb{F}_{q^{12}}$ Arithmetics	Grewal's work [5]	This work
Multiplication	$18\tilde{m}_u + 6\tilde{r} + 110\tilde{a}$	$18\tilde{m}_u + 6\tilde{r} + 96\tilde{a} + a$
7–sparse Mult.	$10\tilde{m}_u + 6\tilde{r} + 47\tilde{a} + \mathbf{6}m_u + a$	–
Pseudo 8–sparse Mult.	–	$10\tilde{m}_u + 6\tilde{r} + 37\tilde{a} + 3a$
Squaring	$12\tilde{m}_u + 6\tilde{r} + 73\tilde{a}$	$12\tilde{m}_u + 6\tilde{r} + 63\tilde{a}$
q–power Frobenius	$5\tilde{m}_u + 6a$	$a + 10m$
q^2–power Frobenius	$10\tilde{m}_u + 2\tilde{a}$	$2a + 8m$
q^3–power Frobenius	–	$3a + 6m$
q^6–power Frobenius	–	$3\tilde{a}$
q^{10}–power Frobenius	–	$2a + 8m$

* : Add./Subtr./Nega./Mult. denote Addition/Subtraction/Negation/Multiplication.

Table 3. Calculation cost of pairings excluding final exponentiation

Method	Calculation cost*
Projective	$1835\tilde{m}_u + 458\tilde{s}_u + 1359\tilde{r} + 9118\tilde{a} + 25a + 308m$
Grewal's work [5]	$70\tilde{i} + 1628\tilde{m}_u + 135\tilde{s}_u + 1120\tilde{r} + 7618\tilde{a} + 69a + 144m + \mathbf{396}m_u$
This work	$69\tilde{i} + 1657\tilde{m}_u + 132\tilde{s}_u + 1130\tilde{r} + 7062\tilde{a} + 229a + 177m + s + 2i$

* : "Projective" means that the Xate pairing uses projective coordinates, and thus 6–sparse multiplication is only available in its Miller's loop.

Table 4. Calculation time of Xate pairing

	Calculation time of Xate pairing* [ms]		
Method	PC	iPad 2	iPhone 5
Previous work	1.48(0.9)	12.3(7.4)	9.97(5.8)
This work	1.46(0.89)	12.1(7.2)	9.84(5.7)

* : In the parenthesis, the calculation time excluding *final exponentiation* is shown. In other words, it is the calculation time for steps 1 to 15 on Alg. 2.

** : "Previous" means that the Xate pairing uses 7–sparse multiplication and affine coordinates.

Table 5. Calculation time of multi–Xate pairing

	Calculation time of multi–pairing on PC [ms]		
	PC		
# pairings	This work	Previous work*	Projective**
1	1.46	1.48	1.31
2	1.86	1.92	1.78
3	2.25	2.30	2.28
4	2.65	2.70	2.75
5	3.02	3.12	3.24
6	3.40	3.51	3.70
7	3.82	3.89	4.18
8	4.18	4.29	4.64
9	4.62	4.71	5.12
10	4.96	5.09	5.58

	Calculation time of multi–pairing on iPhone 5 [ms]		
# pairings	This work	Previous work*	Projective**
1	9.83	9.97	9.91
2	13.0	13.4	13.9
3	16.1	16.8	17.7
4	19.2	20.1	21.5
5	22.5	23.4	25.3
6	25.5	26.7	29.1
7	28.7	30.1	32.9
8	31.8	33.4	36.6
9	34.8	36.7	40.5
10	38.1	40.0	44.3

* : "Projective" means that the Xate pairing uses projective coordinates, and thus 6–sparse multiplication is only available in its Miller's loop.

** : "Previous" means that the Xate pairing uses 7–sparse multiplication and affine coordinates.

By the way, the proposed pseudo 8–sparse multiplication is not able to accelerate *final exponentiation*. Thus, it yields a greater effect for *multi–pairing* than a single pairing because *multi–pairing* can combine the final exponentiations as

$$\prod_{i=1}^{N} \alpha(Q_i, P_i) = \prod_{i=1}^{N} (f_{t-1,Q_i}(P_i))^{(p^k-1)/r} = \left(\prod_{i=1}^{N} f_{t-1,Q_i}(P_i) \right)^{(p^k-1)/r}. \quad (18)$$

In addition, squarings at step 6 in Alg. 2, for example, can also be combined. Table 5 shows the calculation time for N multi–pairing. They are the averages of 12,500 and 4,500 iterations of N multi–pairing on PC and iOS devices (iPad 2

and iPhone 5), respectively. Compared to the case with 6–sparse multiplication and projective coordinates, that with pseudo 8–sparse multiplication and affine coordinates becomes more efficient as the number N becomes larger.

5 Conclusion and Future Works

This paper has proposed *pseudo* 8–sparse multiplication for accelerating Ate–based pairing with affine coordinates on Barreto–Naehrig (BN) curve. According to the calculation costs and experimental results shown in this paper, the proposal made recent efficient pairings such as optimal ate and Xate pairings more efficient, especially together with multi–pairing technique.

As a future work, it should be considered to apply such a sparse multiplication for the other pairings together with some twist techniques.

Acknowledgement. This research was supported by KAKENHI Grant–in–Aid for Scientific Research (B) Number 25280047.

References

1. Lee, E., Lee, H.-S., Park, C.-M.: Efficient and generalized pairing computation on abelian varieties. IEEE Transactions on Information Theory 55(4), 1793–1803 (2009)
2. Vercauteren, F.: Optimal pairings. IEEE Transactions on Information Theory 56(1), 455–461 (2010)
3. Nogami, Y., Sakemi, Y., Kato, H., Akane, M., Morikawa, Y.: Integer Variable χ-based Cross Twisted Ate Pairing and Its Optimization for Barreto-Naehrig Curve. IEICE Transactions on Fundamentals of Electronics 2009(8), 1859–1867 (2009)
4. Aranha, D.F., Karabina, K., Longa, P., Gebotys, C.H., López, J.: Faster explicit formulas for computing pairings over ordinary curves. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 48–68. Springer, Heidelberg (2011)
5. Grewal, G., Azarderakhsh, R., Longa, P., Hu, S., Jao, D.: Efficient Implementation of Bilinear Pairings on ARM Processors. Cryptology ePrint Archive, Vol. 2012:408 (2012)
6. Barreto, P.S.L.M., Naehrig, M.: Pairing-Friendly Elliptic Curves of Prime Order. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 319–331. Springer, Heidelberg (2006)
7. Freeman, D., Scott, M., Teske, E.: A Taxonomy of Pairing-Friendly Elliptic Curves. Journal of Cryptology 23, 224–280 (2006)

A Pseudo 8–Sparse Multiplication

The calculation procedure of pseudo 8–sparse multiplication becomes as follows.

Algorithm 3. Pseudo 8–sparse multiplication

Input: $a, b \in \mathbb{F}_{q^{12}}$,
$a = (a_0 + a_1 v + a_2 v^2) + (a_3 + a_4 v + a_5 v^2)w$, $b = 1 + (b_3 + b_4 v)w$,
where $a_j, b_k \in \mathbb{F}_{q^2} (j = 0, \cdots, 5, k = 3, 4)$,
Output: $c = ab = (c_0 + c_1 v + c_2 v^2) + (c_3 + c_4 v + c_5 v^2)w \in \mathbb{F}_{q^{12}}$

1	$D_0 \leftarrow a_3 \times b_3, D_1 \leftarrow a_4 \times b_4, S_0 \leftarrow a_5 \times b_3$;	$(3\tilde{m}_u)$
2	$T_0 \leftarrow S_0 + D_1$;	$(2\tilde{a})$
3	$T_1 \leftarrow T_0 \times i$;	(m_ξ)
4	$c_0 \leftarrow \mathrm{MontRed}(T_1)$;	$(\tilde{r})$
5	$T_0 \leftarrow a_5 \times b_4$;	$(\tilde{m}_u)$
6	$S_0 \leftarrow S_0 + T_0$;	$(2\tilde{a})$
7	$T_1 \leftarrow T_0 \times i$;	(m_ξ)
8	$c_0 \leftarrow c_0 \oplus a_0$;	$(\tilde{a})$
9	$T_1 \leftarrow T_1 + D_0$;	$(2\tilde{a})$
10	$c_1 \leftarrow \mathrm{MontRed}(T_1)$;	$(\tilde{r})$
11	$t_0 \leftarrow a_3 + a_4, s_0 \leftarrow b_4 + b_3$;	$(2\tilde{a})$
12	$T_1 \leftarrow t_0 \times s_0$;	$(\tilde{m}_u)$
13	$c_1 \leftarrow c_1 \oplus a_1$;	$(\tilde{a})$
14	$T_1 \leftarrow T_1 - D_0 - D_1$;	$(4\tilde{a})$
15	$c_2 \leftarrow \mathrm{MontRed}(T_1)$;	$(\tilde{r})$
16	$T_0 \leftarrow a_2 \times b_4$;	$(\tilde{m}_u)$
17	$c_2 \leftarrow c_2 \oplus a_2$;	$(\tilde{a})$
18	$S_0 \leftarrow S_0 + T_0$;	$(2\tilde{a})$
19	$T_1 \leftarrow T_0 \times i$;	(m_ξ)
20	$t_0 \leftarrow a_0 + a_3, t_{1,0} \leftarrow b_{3,0} + 1, t_{1,1} \leftarrow b_{3,1}$;	$(\tilde{a} + a)$
21	$T_0 \leftarrow t_0 \times t_1$;	$(\tilde{m}_u)$
22	$T_0 \leftarrow T_0 - D_0$;	$(2\tilde{a})$
23	$T_1 \leftarrow T_1 + T_0$;	$(2\tilde{a})$
24	$c_3 \leftarrow \mathrm{MontRed}(T_1)$;	$(\tilde{r})$
25	$T_1 \leftarrow a_1 \times b_3$;	$(\tilde{m}_u)$
26	$S_0 \leftarrow S_0 + T_1$;	$(2\tilde{a})$
27	$c_3 \leftarrow c_3 - a_0, t_0 \leftarrow a_0 + a_4$;	$(2\tilde{a})$
28	$t_{1,0} \leftarrow b_{4,0} + 1, t_{1,1} \leftarrow b_{4,1}$;	(a)
29	$T_0 \leftarrow t_0 \times t_1$;	$(\tilde{m}_u)$
30	$T_0 \leftarrow T_0 - D_1$;	$(2\tilde{a})$
31	$T_1 \leftarrow T_1 + T_0$;	$(2\tilde{a})$
32	$c_4 \leftarrow \mathrm{MontRed}(T_1)$;	$(\tilde{r})$
33	$t_0 \leftarrow a_1 + a_2, s_{0,0} \leftarrow s_{0,0} + 1$;	$(\tilde{a} + a)$
34	$t_0 \leftarrow t_0 + a_5$;	$(\tilde{a})$
35	$T_1 \leftarrow s_0 \times t_0$;	$(\tilde{m}_u)$
36	$T_1 \leftarrow T_1 - S_0$;	$(2\tilde{a})$
37	$c_5 \leftarrow \mathrm{MontRed}(T_1)$;	$(\tilde{r})$
38	$t_0 \leftarrow a_1 \oplus a_2$;	$(\tilde{a})$
39	$c_4 \leftarrow c_4 - a_0$;	$(\tilde{a})$
40	$c_5 \leftarrow c_5 - t_0$;	$(\tilde{a})$
41	Return $c = (c_0 + c_1 v + c_2 v^2) + (c_3 + c_4 v + c_5 v^2)w$	