A Consideration of an Efficient Arithmetic over the Extension Field of Degree 3 for Elliptic Curve Pairing Cryptography

Xin Li, Yuta Kodera, Yoshinori Uetake, Takuya Kusaka, Yasuyuki Nogami
Okayama University, Okayama, Japan

ABSTRACT

This paper presents an efficient arithmetic in extension field based on Cyclic Vector Multiplication Algorithm that reduces calculation costs over cubic extension for elliptic curve pairing cryptography. In addition, we evaluate the calculation costs compared to Karatsuba-based method.

I. INTRODUCTION

Recently, pairing-based cryptography has received much attention since it has many innovative applications. Usually pairing requires pairing-friendly elliptic curves defined over the extension field i.e. $\mathbb{F}_{p^{18}}$ and $\mathbb{F}_{p^{24}}$. The efficiency of pairing depends on the efficient arithmetic in extension field. Therefore, we aim to reduce the number of $\mathbb{F}_p$ multiplications and additions for multiplication, squaring over $\mathbb{F}_{p^3}$, where $\mathbb{F}_{p^3}$ is the base field of $\mathbb{F}_{p^{18}}$ during sextic twist. This paper presents new algorithms based on cyclic vector multiplication algorithm (CVMA), which uses a special class of type-⟨k, m⟩ Gauss period normal bases, and has several advantages: it is easily parallelized; Frobenius mapping is easily carried out since its basis is a normal basis; and it is sufficiently practical and useful when parameters k and m are small [1]. Moreover, we evaluate the calculation costs compared with Karatsuba-based method.

II. THE ALGORITHMS

The cost for calculating elements in finite extension fields $\mathbb{F}_{p^m}$ is based on the following operations: multiplication (M), squaring (S), addition (or subtraction) (A), division by 2 (D_2), and inversion (I) in $\mathbb{F}_p$.

A) Karatsuba-based method

Karatsuba-based method uses polynomial basis $\{1, \alpha, \alpha^2\}$ defined by method uses polynomial $f(x) = x^3 - 2$, and f(x) needs to satisfy irreducible over $\mathbb{F}_p$ so that $\alpha \in \mathbb{F}_{p^3}$.

1. Multiplication

The Karatsuba method for multiplying two elements

$$A = a_0 + a_1\alpha + a_2\alpha^2,\ a_0, a_1, a_2 \in \mathbb{F}_p,$$

$$B = b_0 + b_1\alpha + b_2\alpha^2,\ b_0, b_1, b_2 \in \mathbb{F}_p,$$

is defined as

$$\begin{aligned} AB &= (a_0 + a_1\alpha + a_2\alpha^2)(b_0 + b_1\alpha + b_2\alpha^2) \\ &= a_0b_0 + 2(a_1b_2 + a_2b_1) + (a_0a_1 + a_1b_0 + 2a_2b_2)\alpha \\ &\quad + (a_0b_2 + a_2b_0 + a_1b_1)\alpha^2, \end{aligned}$$

and precomputes the values $T_0 = a_0b_0$, $T_1 = a_1b_1$, $T_2 = a_2b_2$, $T_3 = (a_1 + a_2)(b_1 + b_2)$, $T_4 = (a_0 + a_1)(b_0 + b_1)$, $T_5 = (a_0 + a_2)(b_0 + b_2)$ which costs 6M + 6A. Then the multiplication is performed as

$$\begin{aligned} AB &= T_0 + 2(-T_1 - T_2 + T_3) + (-T_0 - T_1 + 2T_2 + T_4)\alpha \\ &\quad + (-T_0 + T_1 - T_2 + T_5)\alpha^2, \end{aligned}$$

which takes 11A, for a total of 6M + 17A.

2. Squaring

Squaring is defined as

$$\begin{aligned} A^2 &= (a_0 + a_1\alpha + a_2\alpha^2)^2 \\ &= (a_0^2 + 4a_1a_2) + (2a_0a_1 + 2a_2^2)\alpha + (a_1^2 + 2a_0a_2)\alpha^2. \end{aligned}$$

For CH-SQR2 [2], we first precompute the values $T_0 = 2a_1$, $T_1 = a_0^2$, $T_2 = a_2^2$, $T_3 = T_0a_2$, $T_4 = T_0a_0$, $T_5 = (a_0 + a_1 + a_2)^2$, which costs 2M + 3S + 3A. Then the squaring is computed as

$$\begin{aligned} A^2 &= T_1 + 2T_3 + (2T_2 + T_4)\alpha \\ &\quad + (-T_1 - T_2 - T_3 - T_4 + T_5)\alpha^2, \end{aligned}$$

which costs 8A, giving a total cost of 2M + 3S + 11A.

For CH-SQR3 [2], we first precompute the values $T_0 = a_0^2$, $T_1 = (T_6 + a_1)^2$, $T_2 = (T_6 - a_1)^2$, $T_3 = 2a_1a_2$, $T_4 = a_2^2$, $T_5 = (T_1 + T_2) / 2$, $T_6 = a_0 + a_2$, $T_7 = -T_4 + T_5$, which costs 1M + 4S + 6A + $1D_2$. Then the squaring is computed as

$$A^2 = (T_0 + 2T_1) + (-T_1 + T_3 + T_6 - T_7)\alpha + (-T_0 + T_7)\alpha^2,$$

which takes 6A, giving a total cost of 1M + 4S + 12A + $1D_2$.

3. Frobenius mapping

The Frobenius mapping is defined as

$$A^p = (a_0 + a_1\alpha + a_2\alpha^2)^p = a_0 + a_1\alpha^p + a_2\alpha^{2p}.$$

Since $\alpha^p = \alpha^{p-1}\alpha = (\alpha^3)^{\frac{p-1}{3}}\alpha = 2^{\frac{p-1}{3}}\alpha$, and let $\varepsilon = 2^{\frac{p-1}{3}}$,

$$A^p = a_0 + \varepsilon a_1\alpha + \varepsilon^2 a_2\alpha^2.$$

Therefore, the cost of Frobenius mapping is 2M.

4. Inversion

Inversion is defined as $A^{-1} = s^{-1}\bar{A}$, but $s = A\bar{A}$ and $\bar{A} = A^pA^{p^2}$. According to $\varepsilon = 2^{\frac{p-1}{3}}$, we can get $\varepsilon^3 = 1$ and $\varepsilon^2 = -\varepsilon - 1$, thus

$$A^p = a_0 + \varepsilon a_1\alpha + (-a_2 - \varepsilon a_2)\alpha^2,$$

$$A^{p^2} = a_0 + (-a_1 - \varepsilon a_1)\alpha + \varepsilon a_2\alpha^2,$$

and $\bar{A}$ is defined as

$$\begin{aligned} \bar{A} &= (\bar{a}_0, \bar{a}_1, \bar{a}_2) \\ &= (a_0^2 - 2a_1a_2) + (2a_2^2 - a_0a_1)\alpha + (a_1^2 - a_0a_2)\alpha^2, \end{aligned}$$

which costs 3M + 3S + 5A. Then, s defined as

$$s = a_0\bar{a}_0 + 2(a_1\bar{a}_2 + a_2\bar{a}_1),$$

which costs 3M + 3A. Finally,

$$A^{-1} = s^{-1}\bar{a}_0 + s^{-1}\bar{a}_1\alpha + s^{-1}\bar{a}_2\alpha^2,$$

which costs 3M + 1I, and the total cost of inversion is 9M + 3S + 8A + 1I.

B) Proposed method based on CVMA

CVMA uses the basis $\{\tau_1, \tau_2, \tau_3\} = \{\omega_1 + \omega^{-1}, \omega_2 + \omega^{-2}, \omega_3 + \omega^{-3}\}$ defined by modular polynomial $\Phi_7(x) = (x^7 - 1)/(x - 1)$ and its root $\omega \in \mathbb{F}_{p^3}$. In this case, $p \not\equiv 1,6 \pmod 7$ such that $\{\tau_1, \tau_2, \tau_3\}$ becomes a basis. It realizes efficient multiplication, squaring, and Frobenius mapping [3].

1. Multiplication

The CVMA-based method for multiplying two elements A, B is defined as

$$\begin{aligned} AB &= (a_1\tau_1 + a_2\tau_2 + a_3\tau_3)(b_1\tau_1 + b_2\tau_2 + b_3\tau_3) \\ &= (a_1b_2 + a_2b_1 + a_2b_3 + a_3b_2 - 2a_1b_1 - 2a_2b_2 - a_3b_3)\tau_1 \\ &\quad + (a_1b_3 + a_3b_1 + a_2b_3 + a_3b_2 - a_1b_1 - 2a_2b_2 - 2a_3b_3)\tau_2 \\ &\quad + (a_1b_2 + a_2b_1 + a_1b_3 + a_3b_1 - 2a_1b_1 - a_2b_2 - 2a_3b_3)\tau_3, \end{aligned}$$

and precomputes the values $T_1 = a_1b_1$, $T_2 = a_2b_2$, $T_3 = a_3b_3$, $T_4 = (a_1 - a_2)(b_2 - b_1)$, $T_5 = (a_2 - a_3)(b_3 - b_2)$, $T_6 = (a_1 - a_3)(b_3 - b_1)$, which costs 6M + 6A. Then the multiplication is performed as

$$\begin{aligned} AB &= (-T_1 + T_4 + T_5)\tau_1 + (-T_2 + T_5 + T_6)\tau_2 \\ &\quad + (-T_3 + T_4 + T_6)\tau_3, \end{aligned}$$

which costs 6A, for a total of 6M + 12A.

2. Squaring

The squaring is defined as

$$\begin{aligned} A^2 &= (a_1\tau_1 + a_2\tau_2 + a_3\tau_3)^2 \\ &= (-2a_1^2 - 2a_2^2 - a_3^2 + 2a_1a_2 + 2a_2a_3)\tau_1 \\ &\quad + (-a_1^2 - 2a_2^2 - 2a_3^2 + 2a_1a_3 + 2a_2a_3)\tau_2 \\ &\quad + (-2a_1^2 - a_2^2 - 2a_3^2 + 2a_1a_2 + 2a_1a_3)\tau_3, \end{aligned}$$

and precomputes the values $T_0 = (a_1 - a_2)^2$, $T_1 = a_3 - a_1$, $T_2 = T_1^2$, $T_3 = a_3^2$, $T_4 = 2a_2$, $T_5 = T_1T_4$, $T_6 = T_4(a_3 - a_2)$, $T_7 = -T_2 - T_3$, which costs 2M + 3S + 5A. Then the squaring is computed as

$$A^2 = (-2T_0 - T_3 + T_5) + (T_6 + T_7)\alpha + (-T_0 + T_7)\alpha^2,$$

which costs 5A, giving a total cost of 2M + 3S + 10A.

Or we precompute $T_0 = a_2 + a_3$, $T_1 = (T_0 - a_1)^2$, $T_2 = (T_0 + a_1)^2$, $T_3 = (T_1 + T_2)/2$, $T_4 = a_2^2$, $T_5 = 2a_2a_3$, $T_6 = (a_1 - a_2)^2$, $T_7 = 2T_5 - T_3$, $T_8 = T_7 - T_1$, which costs 1M + 4S + 9A + $1D_2$. Then the squaring is computed as

$$A^2 = (-T_6 + T_7)\tau_1 + (-T_4 + T_5 + T_6 + T_8)\tau_2 + (T_4 + T_8)\tau_3,$$

which costs 5A, giving a total cost of 1M + 4S + 14A + $1D_2$.

3. Frobenius mapping

The Frobenius mapping is defined as

$$A^p = (a_1\tau_1 + a_2\tau_2 + a_3\tau_3)^p = a_1\tau_1^p + a_2\tau_2^p + a_3\tau_3^p$$

For example, if $p \equiv 5 \pmod 7$,

$$\begin{aligned} \tau_1^p &= \omega^5 + \omega^{-5} = \omega^2 + \omega^{-2} = \tau_2, \\ \tau_2^p &= \omega^{10} + \omega^{-10} = \omega^3 + \omega^{-3} = \tau_3, \\ \tau_3^p &= \omega^{15} + \omega^{-15} = \omega^1 + \omega^{-1} = \tau_1, \end{aligned}$$

thus $A^p = a_3\tau_1 + a_1\tau_2 + a_2\tau_3$. The cost is 0, because we Just need to shift the coefficients. Similarly, the cost of other p is also 0.

4. Inversion

Inversion is defined as $A^{-1} = -s^{-1}\bar{A}$, but $s = A\bar{A}$ and $\bar{A} = A^pA^{p^2}$. If $p \equiv 5 \pmod 7$,

$$\begin{aligned} A^p &= (a_3, a_1, a_2) = a_3\tau_1 + a_1\tau_2 + a_2\tau_3, \\ A^{p^2} &= (a_2, a_3, a_1) = a_2\tau_1 + a_3\tau_2 + a_1\tau_3, \end{aligned}$$

and $\bar{A}$ is defined as

$$\begin{aligned} \bar{A} &= (\bar{a}_0, \bar{a}_1, \bar{a}_2) \\ &= (a_1^2 + a_3^2 - 2a_1a_3 - a_2a_3)\tau_1 \\ &\quad + (a_1^2 + a_2^2 - 2a_1a_2 - a_1a_3)\tau_2 \\ &\quad + (a_2^2 + a_3^2 - 2a_2a_3 - a_1a_2)\tau_3, \end{aligned}$$

and precompute the values $T_1 = a_3 - a_1$, $T_2 = a_1 - a_2$, $T_3 = a_3 - a_2$, $T_4 = T_1^2$, $T_5 = T_2^2$, $T_6 = T_3^2$, $T_7 = a_2a_3$, $T_8 = a_1a_3$, $T_9 = a_1a_2$, which costs 3M + 3S + 3A. Then $\bar{A}$ is performed as

$$\bar{A} = (T_4 - T_7)\tau_1 + (T_5 - T_8)\tau_2 + (T_6 - T_9)\tau_3$$

which costs 3A, for a total of 3M + 3S + 6A.

Second, s is defined as

$$\begin{aligned} s &= \bar{a}_1(-2a_1 + a_2) + \bar{a}_2(a_1 - 2a_2 + a_3) + \bar{a}_3(-a_2 + a_3) \\ &= \bar{a}_1(-a_1 - T_2) + \bar{a}_2(T_2 + T_3) + \bar{a}_3(-T_3), \end{aligned}$$

which costs 3M + 4A. Finally,

$$A^{-1} = -s^{-1}\bar{a}_0 - s^{-1}\bar{a}_1\alpha - s^{-1}\bar{a}_2\alpha^2,$$

which costs 3M + 1I, the total cost of inversion is 9M + 3S + 10A + 1I.

Table 1 lists the costs for a multiplication, squaring, Frobenius mapping and inversion.

Table 1 Summary of calculation costs

Operation	Karatsuba-based	Proposal
Multiplication	6M + 17A	6M + 12A
Squaring(1)	2M + 3S + 11A	2M + 3S + 10A
Squaring(2)	1M + 4S + 12A + $1D_2$	1M + 4S + 14A + $1D_2$
Frobenius mapping	2M	0
Inversion	9M + 3S + 8A + 1I	9M + 3S + 10A + 1I

III. EXPERIMENTAL RESULT

The performance of multiplication, squaring and inversion was measured on a Intel(R) Core i5-6500 CPU 3.20GHz 3.20GHz with 8.00GB of RAM, running GNU/Windows 10. Programs were coded in C and compiled with the GNU C Compiler version 6.3.0. The prime number which used is 256-bit and satisfies conditions of Karatsuba-based method and CVMA-based method. We ran 100,000 times and measured the average time required, the result is shown in Table 2.

Table 2 Timings of calculation

Operation	Karatsuba-based (μs)	CVMA-based (μs)
Multiplication	0.17	**0.15**
Squaring(1)	0.16	0.16
Squaring(2)	**0.17**	0.18
Inversion	**9.29**	9.30

IV. CONCLUSION

In elliptic pairing cryptography, we calculate pairing by Miller's algorithm, which uses more multiplications than squarings. In addition, Frobenius mapping of CVMA costs 0, which used not only for inversion but also for pairing calculation and scalar multiplication. Therefore, CVMA-based method is more efficient than Karatsuba-based method.

V. REFERENCES

[1] K. Hidehiro, Y. Nogami, T. Yoshida, and Y. Morikawa, "Cyclic vector multiplication algorithm based on a special class of Gauss period normal basis." ETRI journal 29.6, 2007, pp. 769-778.

[2] J. Chung, M. A. Hasan, "Asymmetric squaring formulae", 18th IEEE Symposium on Computer Arithmetic, 2007, pp. 113-122.

[3] N. Kenta, "Proposals of Multiplication and Inversion Methods in Extension Field for Scalable Asymmetric-key and Fast Symmetric-key Cryptosystems", 2013.