Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 3846369, 8 pages
https://doi.org/10.1155/2022/3846369

Research Article

Memory-Efficient Algorithm for Scalar Multiplications on Twisted Edwards Curves for Isogeny-Based Cryptosystems

Sookyung Eom,[1] **Hyang-Sook Lee**,[2] **and Kyunghwan Song**[3]

[1]*Institute of Mathematical Sciences, Ewha Womans University, Seoul, Republic of Korea*
[2]*Department of Mathematics, Ewha Womans University, Seoul, Republic of Korea*
[3]*Department of Mathematics, Jeju National University, 102, Jejudaehak-ro, Jeju-si, Jeju, Republic of Korea*

Correspondence should be addressed to Sookyung Eom; esk9030@gmail.com and Hyang-Sook Lee; hsl@ewha.ac.kr

Received 1 November 2021; Revised 16 February 2022; Accepted 26 March 2022; Published 27 April 2022

Academic Editor: Salvatore Alfonzetti

Scalar multiplications are considered an essential aspect of implementations of isogeny-based cryptography. The efficiency of scalar multiplication depends on the equation of the underlying elliptic curves and the addition chain employed. Bos and Friedberger recently stated that, for larger scalar multiplication, addition-subtraction chains will become more useful for twisted Edwards curves because of the differential restriction on Montgomery curves in the setting of isogeny-based cryptosystem. Motivated by these comments, we attempt to increase the efficiency of scalar multiplication in twisted Edwards curves in terms of the memory of algorithms. In this paper, we present a double-base addition-subtraction chain algorithm with memory efficiency for scalar multiplication. The memory usage of this part is $O((\log n)^2/\log\log n)$, which is better than the result of the tree-based approach, which is $O((\log n)^2)$.

1. Introduction

1.1. Isogeny-Based Cryptosystem. The isogeny cryptosystem is currently considered one of the promising candidates for maintaining security in quantum computing environments. Couveignes [1] first introduced the concept of isogeny-based cryptosystems in 2006, and Charles et al. [2] first developed an isogeny cryptosystem based on supersingular elliptic curves in 2009. However, the ordinary elliptic curve was found to be vulnerable in an isogeny-based system by a subexponential quantum algorithm developed by Childs et al. [3]. Therefore, an isogeny-based cryptosystem is currently being studied at the supersingular curve rather than at the ordinary elliptic curves. One of the important works on the isogeny-based cryptosystem is the supersingular isogeny Diffie–Hellman (SIDH) key exchange that was introduced by Jao and Feo [4] in 2011. There have been a lot of related studies [5–8] examining this scheme. One example is the supersingular isogeny key encapsulation (SIKE) [9] submitted to the NIST (National Institute of Standards and Technology) competition on the standardisation of post-quantum cryptosystems. Practical hardware implementations [10–14] of SIKE have also been developed, and more are expected to be available in the future.

To achieve more efficient isogeny-based cryptosystems, it is necessary to investigate methods that improve the isogeny calculations or the scalar multiplications. In particular, fast exponentiation and inversion are critical in the process of the scalar multiplication of points over elliptic curve. Koziel et al. [15] improved upon the efficiency of isogeny-based cryptosystems by avoiding the computation of finite field inversions in the scalar multiplication of points. They also worked on the shape of smooth isogeny primes and presented new windowing methods to calculate fast addition chains. Recently, Bos and Friedberger [16] sped up the scalar multiplications on twisted Edward curves by using an efficient addition-subtraction chain algorithm.

1.2. Our Contribution. Motivated by the previous research stated in Section 1.1, our work proposes a new double-base

addition-subtraction chain algorithm and demonstrates scalar multiplication using the proposed chain for a twisted Edwards curve with projective coordinates. In particular, we focus on the memory efficiency aspect. Our algorithm leads to a memory-efficient system because it only needs $O((\log n)^2/\log\log n)$, which is better than the result of the tree-based approach, which is $O((\log n)^2)$ [17]. Therefore, in terms of memory efficiency, the approach based on a modified greedy algorithm is shown to be a good choice for scalar multiplication. These results represent an improvement upon the scalar multiplication of isogeny-based cryptosystems in the field of post-quantum cryptography. The experimental results also show that the length of the chain is much shorter, and as a result, for scalar multiplication in a isogeny-based cryptosystem, the complexity is reduced by 20% compared to the single-base number system.

1.3. Organization. The paper is organized as follows. In Section 2, we give preliminaries on isogenies and addition-subtraction chains on double-base number systems for elliptic curve isogeny-based cryptosystems. Section 3 introduces twisted Edwards curves and their various coordinates. In Section 4, we present our new algorithms and compare the memory usage with previous work [18] and suggest the experimental result for the computational complexity. Section 5 concludes our work.

2. Preliminaries

In this section, we review briefly some concepts and properties of isogenies and addition-subtraction chains for elliptic curve isogeny-based cryptosystems.

2.1. Isogeny. Let E_1 and E_2 be two elliptic curves over a finite field $\mathbb{F}_q$. An isogeny ϕ: $E_1(\overline{\mathbb{F}_q}) \longrightarrow E_2(\overline{\mathbb{F}_q})$ is a nonconstant group homomorphism (E_1 and E_2) that is given by rational functions. This means that $\phi(P+Q) = \phi(P) + \phi(Q)$ for all $P, Q \in E_1(\overline{\mathbb{F}_q})$ and there exist rational functions r_1, r_2 such that if $\phi(x_1, y_1) = (x_2, y_2)$, then

$$\begin{aligned} x_2 &= r_1(x_1, y_1), \\ y_2 &= r_2(x_1, y_1), \end{aligned} \tag{1}$$

for all but finitely many $(x_1, y_1) \in E_1(\overline{\mathbb{F}_q})$ [19].

If the degree of ϕ as a group homomorphism is ℓ, the isogeny is called an ℓ-isogeny. If the degree of ϕ is equal to the cardinality of kernel of ϕ, ϕ is called separable. If there exists a separable isogeny between two curves, we say that they are isogenous. For an isogeny ϕ between two curves E_1, E_2 with $E_1 = E_2$, we say ϕ is an endomorphism. The set of endomorphisms of an elliptic curve, denoted by end (E), has a ring structure with the operations of point-wise addition and function composition.

2.2. Isogeny-Based Cryptosystems

Definition 1. A prime of the form $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1$ is called a smooth isogeny prime, where ℓ_A and ℓ_B and small primes, e_A and e_B, are positive integers, and f is a small cofactor to make p prime.

Isogeny-based cryptosystems use smooth isogeny primes, and the case where $\ell_A = 2$ and $\ell_B = 3$ is preferred since this case enjoys the fastest known isogeny computations [8]. Let E be an elliptic curve and let $m \in \mathbb{N}$.

Definition 2. Let E be an elliptic curve and let $m \in \mathbb{N}$. Then, $E[m]$ is called m-torsion subgroup of E, which is the set of points of order m:

$$E[m] = \{P \in E \mid [m]P = O\}. \tag{2}$$

An elliptic curve E over a field of characteristic p has a torsion subgroup $E[p^r]$ for which holds that either $E[p^r] = O$ or $E[p^r] = \mathbb{Z}/p^r\mathbb{Z}$ for all $r \geq 1$, the first case is called that E is supersingular and the other case is called E is ordinary. There are some hard problem candidate hard problem related to supersingular elliptic curves as follows [20].

Problem 1. Let p, ℓ be distinct prime numbers. Let E_1 and E_2 be two elliptic curves over $\mathbb{F}_{p^2}$ with $\#E_1(\mathbb{F}_{p^2}) = \#E_2(\mathbb{F}_{p^2}) = (p+1)^2$, chosen uniformly at random. Find $k \in \mathbb{N}$ and an isogeny of degree ℓ^k from E_1 to E_2.

We remark that the ordinary Problem 1 stated for the general curve is not known to be equivalent, and that there also exists a subexponential algorithm that computes the endomorphism ring of ordinary curves [21]. Although the best algorithm that computes isogenies is still exponential, there is a subexponential quantum algorithm that computes an isogeny between two ordinary curves [3]; therefore, studies on isogeny-based cryptography are focused on supersingular cases.

2.3. Chains for Scalar Multiplication. There are several candidates to reduce the cost of scalar multiplications on elliptic curve cryptosystems. In isogeny-based cryptosystems, the results of the scalar multiplication of elliptic curves are addition chains [15], addition-subtraction chains [16], etc. In this section, we introduce an addition-subtraction chain using double base.

Definition 3. Given two primes $p, q \in \mathbb{N}$, a double-base number system (DBNS) is a representation into which every positive integer n is represented as the linear combination with coefficients in $1, -1$ and sum of difference of $\{p, q\}$-integers, i.e., numbers of the form $p^a q^b$:

$$n = \sum_{i=0}^{\ell} d_i p^{a_i} q^{b_i}, \tag{3}$$

with $d_i \in \{-1, 1\}$ and $a_0 \geq a_1 \geq a_2 \geq \ldots \geq a_\ell \geq 0$ and $b_0 \geq b_1 \geq b_2 \geq \ldots \geq b_\ell \geq 0$.

The representation of an integer n is not unique in both unsigned case and signed case. The most representative algorithm for expressing arbitrary n by DBNS is the greedy algorithm. The greedy algorithm is as follows.

Theorem 1 (see [22]). *Algorithm 1 terminates after* $k \in \mathcal{O}(\log n/\log \log n)$ *steps.*

In Algorithm 1, lines 1–3 give the limitation for the input parameter a_0 and line 4 gives the value of the parameter b_0. The parameter a_0 is used when the exponentiation a_1 of 2 is the biggest term in the expansion, that is, $0 \le a_\ell \le a_{\ell-1} \le \cdots \le a_1 \le a_0$. The value of b_0 also gives limit of b_i 's similarly.

3. Twisted Edwards Curves

3.1. Twisted Edwards Curves. Let k be a field of odd characteristic, and $a, d \in k$ with $a\,d(a-d) \neq 0$.

Definition 4 (see [24]). The twisted Edwards curve with coefficients a and d is defined by the equation

$$E_{E,a,d}: ax^2 + y^2 = 1 + dx^2y^2. \tag{4}$$

An Edwards curve is a twisted Edwards curve with $a = 1$.

Edwards curves and twisted Edwards curves have been broadly used in cryptography [24]. Every twisted Edwards curve is birationally equivalent to an elliptic curve in Montgomery curve [24].

Definition 5 (see [24]). The twisted Edwards curve $E_{E,a,d}: ax^2 + y^2 = 1 + dx^2y^2$ is a quadratic twist of the Edwards curve $E_{E,1,d/a}: \tilde{x}^2 + \tilde{y}^2 = 1 + (d/a)\tilde{x}^2\tilde{y}^2$. The map $(\tilde{x}, \tilde{y}) \mapsto (x, y) = (\tilde{x}/\sqrt{a}, \tilde{y})$ is an isomorphism from $E_{E,1,d/a}$ to $E_{E,a,d}$ over $k(\sqrt{a})$. If a is a square in k and $d \in k^*/(k^*)^2$ is not square in k, then $E_{E,a,d}$ is isomorphic to $E_{E,1,d/a}$ over k but not isomorphic to E over k [25]. Generally, $E_{E,a,d}$ is a quadratic twist of $E_{E,\overline{a},\overline{d}}$ for $\overline{a}, \overline{d}$ satisfying $\overline{d}/\overline{a} = d/a$. Conversely, every quadratic twist of a twisted Edwards curve is isomorphic to a twisted Edwards curve. That is, the set of twisted Edwards curves is invariant under quadratic twists.

Let $P = (x_1, y_2)$ and $Q = (x_2, y_2)$ be points on a twisted Edwards curve $E_{E,a,d}$. The sum of these points $(x_1, y_1), (x_2, y_2)$ on $E_{E,a,d}$ is

$$P + Q = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2}\right). \tag{5}$$

$(0, 1)$ is the neutral element of the group, and the opposite of (x_1, y_1) is $(-x_1, y_1)$. These formulas also work for doubling, that is, if $P = Q$. In particular,

$$2P = \left(\frac{2x_1y_1}{ax_1^2 + y_1^2}, \frac{y_1^2 - ax_1^2}{2 - ax_1^2 - y_1^2}\right). \tag{6}$$

In [6], the twisted Edwards curves have simple completeness and support the ADD function. The twisted Edwards model has been considered for implementations because it offers a single simple group law formula that works for all possible inputs. When one can exploit precomputation, the ADD operation used in scalar multiplication for Edwards and twisted Edwards curves[6] can play an important role in improving the speed of scalar multiplication, particularly in the ephemeral Diffie–Hellman key-exchange protocol.

3.2. Coordinates. Bernstein et al. [24] offered efficient formulas for addition and doubling for Edwards curves and twisted Edwards curves. They introduced various coordinates for Edwards curves and twisted Edwards curves.

Definition 6. A projective point (x, y, z) on $ax^2 + y^2 = 1 + dx^2y^2$ is represented as X, Y, Z, T satisfying the following relations: $x = X/Z$, $y = Y/Z$, $xy = T/Z$. The coordinates of the point $(X: Y: Z: T)$ are called the extended twisted Edwards coordinates. The identity element is represented by $(0: 1: 1: 0)$. The opposite of a point is $(-X: Y: Z: -T)$.

Definition 7. The following equation:

$$\left(aX^2 + Y^2\right)Z^2 = Z^4 + dX^2Y^2, \tag{7}$$

is called the projective twisted Edwards curve to avoid inversions. For $Z_1 \neq 0$, the homogeneous point $(X_1: Y_1: Z_1)$ represents the affine point $(X_1/Z_1, Y_1/Z_1)$ on $E_{E,a,d}$.

Definition 8. Another way to avoid inversions is using a point $(X_1: Y_1: Z_1)$ with $X_1Y_1Z_1 \neq 0$ to represent the affine point $(Z_1/X_1, Z_1/Y_1)$ on $E_{E,a,d}$. These points are called the inverted twisted Edwards coordinates.

The following Table 1 shows the complexity of addition and doubling calculated according to the considered coordinates of the curves, that DBL is point doubling, ADD is point addition, mADD is mixed point addition: addition of an input that has been scaled to have Z-coordinate 1, mDBL is mixed doubling: doubling of an input that has been scaled to have Z-coordinate 1.

Bos and Friedberger [16] proposed an algorithm to compute scalar multiplications on twisted Edwards curves using addition-subtraction chains. Their results show improved speed of scalar multiplication compared to the Montgomery curves most commonly used in isogeny-based cryptography systems. They also show that using larger scalar values in scalar multiplication leads to better performance. However, their chains are difficult to find for such large numbers, and the operations are generally more expensive; therefore, it is unclear if twisted Edwards curves can become faster in their setting. The present work was motivated by the results of Bos and Friedberger's research. We propose the use of a double-base addition-subtraction chain on twisted Edwards curves with projective coordinates, which led us to think about the efficiency of scalar multiplication.

4. Memory-Efficient Algorithm for Scalar Multiplications

In terms of addition and doubling operations in twisted Edwards curves, we use the following notations.

(1) ADD (resp. SUB): the cost of point addition (resp. subtraction)

(2) M: number of field multiplications

(3) D: number of field doublings

(4) S: number of field squarings

Input: a_0, $S = \{1, -1\}$, N
Output: three sequences $(d_i, a_i, b_i)_{0 \le i \le \ell}$ such that $N = \sum_{i=0}^{\ell} d_i 2^{a_i} 3^{b_i}$ with $d_i \in S$, $a_0 \ge \cdots \ge a_\ell$, $b_0 \ge \cdots \ge b_\ell$

(1) **if** $a_0 > [\log_3 N]$ **then**
(2) **return fail**
(3) **end if**
(4) $b_0 \leftarrow \lceil (\log_2 N - a_0)\log_2 3 \rceil$
(5) $(i, t, s) \leftarrow (1, N, 1)$
(6) **while** $t > 0$ **do**
(7) find the best approximation $z = d_i 2^{a_i} 3^{b_i}$ of t with $di \in S$, $0 \le a_i \le a_{i-1}$, and $0 \le b_i \le b_{i-1}$
(8) $d_i \leftarrow s \times d_i$
(9) **if** $t < z$ **then**
(10) $s \leftarrow -s$
(11) **end if**
(12) $t \leftarrow |t - z|$
(13) $i \leftarrow i + 1$
(14) **end while**
(15) **return** $(d_i, a_i, b_i)_{i \le i \le \ell}$

Algorithm 1: The greedy algorithm for double-base number system [23].

Table 1: Costs of addition and doubling operations in twisted Edwards curve.

Curve, representation	DBL	ADD	mADD	mDBL
Twisted Edwards curve with projective coordinates	7	11	10	6
Twisted Edwards curve with inverted coordinates	7	10	9	6
Twisted Edwards curve with extended coordinates	8	9	8	7

(5) T: number of field triplings

(6) I: number of field inversions

Bos and Friedberger [16] showed that scalar multiplications are more effective in twisted Edwards curves than they are in Montgomery curves for isogeny-based protocols. In this section, we introduce an algorithm that improves upon those of Bos and Friedberger [16] and Yu et al. [18] in terms of memory. Our method first uses double-base addition-subtraction chains. Secondly, we use twisted Edwards curves with projective coordinates to improve the addition and subtraction calculations determined by the chains in isogeny-based cryptosystems.

We show that the following algorithm generates addition-subtraction chains using a double base to improve the total complexity. Addition-subtraction chains with double base can generate chains that are much shorter than a single-base chain. This means that the number of operations in terms of addition and subtraction calculations can be reduced. A decrease in the length of the chain suggests a more optimized structure. After a short chain with double base is achieved using Algorithm 2, we use the twisted Edwards curves with projective coordinates to reduce the complexity of the addition and subtraction calculation, since the projective twisted Edwards curves allow for computing inversions to be avoided. That is, the projective twisted Edward curves have $\mathrm{ADD} = \mathrm{SUB} = 10M + 1S = 11M$ for $S = 1M$, $D = 3M + 4S = 7M$, $T = 9M + 3S = 12M$, $I = 100M$ (see the twisted Edwards curve projective row in the 1st table in [26]). We can thus see that the total complexity can be reduced by adjusting the number of additions and subtractions. Therefore, in terms of total complexity, our algorithm decreases the addition part because the length of the chain is decreased.

We provide a modified greedy algorithm for isogeny-based cryptography, which is used to produce a double-base addition-subtraction chain for each given N to compute the scalar multiplication NP, where P is a point in a twisted Edwards curve with projective coordinates over F_{q^2}. Furthermore, to avoid the large inversion part of the calculation, we work on twisted Edwards curve with projective coordinates.

In Algorithm 2, checking the integers α and β for all $0 \le \alpha \le a$ and $0 \le \beta \le b$ is very inefficient, since it is necessary to repeat $O(\log n)$ times to find and compare the values $|t - 2^x 3^y|$ where $0 \le y \le b$ for fixed $0 \le x \le a$, whose complexity is $O(\log n)$. As an efficient way to obtain α and β, we suggest Algorithm 3, which is an application of the binary search [27] and which only requires $O(\log\log n)$ iterations.

Note that the required number of steps of Algorithm 2 is equivalent to that of Algorithm 1, in big-O notation manner. By combining Algorithm 3, we obtain an addition-subtraction multiplication chain c for N whose number of iterations is $O(\log n/\log\log n \cdot \log\log n) = O(\log n)$. It is equivalent to many existing results such as those in [6, 17, 22] which are focusing on the speed to find a chain.

For example, when $N = 72641$, the double-base chain is

$$\begin{aligned} 72641 &= 2^7 3^6 - 2^5 3^6 + 2^5 3^4 + 2^3 3^2 \\ &= 2^1 3^1 \left(2^2 3^1 \left(2^2 3^2 \left(3^2 \left(2^2 - 1\right) + 1\right) + 1\right) + 1\right) - 1\big) - 1. \end{aligned} \tag{8}$$

Input: N
Output: an addition-subtraction multiplication chain c for N
(1) $b \leftarrow \lceil \log_3 N/2 \rceil$
(2) $a \leftarrow \lceil (\log_3 N - b)\log_2 3 \rceil$
(3) $(i, t, s) \leftarrow (0, N, 1)$
(4) **while** $t \neq 0$ **do**
(5) Find the pair of integers (α, β) with $0 \leq \alpha \leq a, 0 \leq \beta \leq b$ such that $2^\alpha 3^\beta$ is the closest number to t
(6) $v[i] \leftarrow (s, \alpha, \beta)$
(7) $t \leftarrow t - s \cdot 2^\alpha 3^\beta$
(8) **if** $t > 0$ **then**
(9) $s \leftarrow 1$
(10) **else if** $t < 0$ **then**
(11) $s \leftarrow -1$
(12) **else**
(13) return $(v[0], \ldots, v[i])$
(14) **end if**
(15) $i \leftarrow i + 1$
(16) $(a, b) \leftarrow (\alpha, \beta)$
(17) **end while**

Algorithm 2: A modified greedy algorithm for isogeny-based cryptography.

In this case, the number of doubling is 7, tripling is 6, and addition (or subtraction) is 5. Therefore, the complexity is $7 \cdot 7 + 12 \cdot 6 + 11 \cdot 5 = 176M$. A single-based chain [28] is

$$\begin{aligned} 72641 &= 2^{17} - 2^{16} + 2^{13} - 2^{11} + 2^{10} - 2^7 + 2^6 - 2^0 \\ &= 2^6\left(2\left(2^3\left(2\left(2^2\left(2^3(2-1)+1\right)-1\right)+1\right)-1\right)+1\right)-1. \end{aligned} \tag{9}$$

In this case, the number of doubling is 17, and addition (or subtraction) is 7. Therefore, the complexity is $7 \cdot 17 + 11 \cdot 7 = 196M$.

To find a short chain, various techniques are considered, such as a brute force approach [16] or a tree-based approach [29]. However, these approaches require huge amounts of memory for the following reasons. First, these approaches use recursive calls. These approaches also search for and store the chains in the intermediate stage, which are useless in the final stage, thus representing a burden on memory. As the number of chains required to conduct scalar multiplications increases, the burden on memory increases dramatically because the number of chains under consideration increases exponentially. By contrast, Algorithm 2 provides a memory-efficient system because it requires a total of $3 \times (l + 1)$ array. The burden on memory does not increase exponentially, as it only increases linearly. Therefore, in terms of memory efficiency, the approach based on a modified greedy algorithm is a good choice for scalar multiplication.

In Algorithm 3, first, the values of powers of 3 are stored to find the number $2^\alpha 3^\beta$, which is closest to t faster, and t is then compared with the following values: $2^0 3^i, 2^1 3^i, \ldots, 2^a 3^i$ with the rules in binary search. For example, let $t = 53$ and say that we have to find the closest number to t in the following numbers: $2^0 3^1, 2^1 3^1, \ldots, 2^{10} 3^1$ First, take $2^5 3^1 = 96$ and compare it to t. Since $t < 96$, we only need to compare t to $2^0 3^1, \ldots, 2^4 3^1$, and then take $2^2 3^1 = 12$ and compare it with t. Since $t > 12$, we only need to compare t with $2^3 3^1, 2^4 3^1$. Finally, take $2^4 3^1 = 48$ and compare it to t. Since $t > 48$, the remaining step is to compare $|t - 2^4 3^1|$ to $|t - 2^5 3^1|$.

Hence, we obtain that $2^4 3^1 = 48$ is the closest number to t when the power of 3 is fixed to 1. Now, we can find the numbers of the form $2^M 3^i$ closest to t for each fixed i, and by checking for all i, we obtain the closest number to t between 0 and $2^a 3^b$. Hence, we find that $2^4 3^1 = 48$ is the closest number to t when the power of 3 is fixed to 1. Now, we can find the numbers of the form $2^M 3^i$ that are closest to t for each fixed i, and by checking for all i, we obtain the closest number to t between 0 and $2^a 3^b$.

The memory of our algorithm is compared to the amount of memory in Yu et al. [18], a recent study of the same double-base addition-subtraction chain. Because Yu et al.'s results are also more recent than those of Bos and Friedberger, and Bos and Friedberger's results still show no results on the amount of memory in the algorithm. The part that takes up the most memory is line 4 and line 5 in Algorithm 2. The memory usage of this part is (the chain length) $O(\log n)$, that is, $O((\log n)^2/\log\log n)$. It is better than the memory usage of the tree-based approach, that is, $O((\log n)^2)$ [18].

To do the practical experiment, we let n be a fixed positive integer and generate ten random numbers between 2^{n-1} and 2^n. In other words, we generate ten n-bit integers N, and then we write

(i) $\#M$: the average value of the number of multiplications for scalar multiplication,
(ii) $\#M$/bit: the number of multiplications per number of bits,
(iii) Improvement: the percentage of improvements

relative to the result of the binary representation for each case in Table 2. Our algorithm obtains a complexity result

Input: t
Output: (α, β) with $0 \le \alpha \le a, 0 \le \beta \le b$ such that $2^\alpha 3^\beta$ is the closest number to t
(1) $i \leftarrow b, v \leftarrow (0,0,0)$
(2) $x \leftarrow (3^0, 3^1, \ldots, 3^b)$
(3) **while** $i \neq 0$ **do**
(4) $F \leftarrow 0$, $L \leftarrow a$
(5) **while** $F \le L$ **do**
(6) $M \leftarrow \lceil (F + L)/2 \rceil$
(7) **if** $2^M 3^{x[i]} < t$ **then**
(8) $F \leftarrow M + 1$
(9) **else if** $2^M 3^{x[i]} > t$ **then**
(10) $L \leftarrow M - 1$
(11) **else**
(12) $vtmp \leftarrow (|\pi t - 2^M 3^{x[i]}|, M, x[i])$
(13) **end if**
(14) **end while**
(15) **if** $2^M 3^{x[i]} < t$ **then**
(16) **if** $t < 3 \cdot 2^{M-2} 3^{x[i]}$ **then**
(17) $vtmp \leftarrow (|t - 2^{M-1} 3^{x[i]}|, M - 1, x[i])$
(18) **else**
(19) $vtmp \leftarrow (|t - 2^M 3^{x[i]}|, M, x[i])$
(20) **end if**
(21) **else**
(22) if $t < 3 \cdot 2^{M-1} 3^{x[i]}$ then
(23) $vtmp \leftarrow (|t - 2^M 3^{x[i]}|, M, x[i])$
(24) **else**
(25) $vtmp \leftarrow (|t - 2^{M+1} 3^{x[i]}|, M + 1, x[i])$
(26) **end if**
(27) **end if**
(28) **if** $v = (0,0,0)$ **or** $v[0] > vtmp[0]$ **then**
(29) $v \leftarrow vtmp$
(30) **end if**
(31) $i \leftarrow i - 1$
(32) **end while**
(33) return $(v[1], v[2])$

Algorithm 3: An efficient way to obtain α and β in Algorithm 2.

Table 2: The comparison of the different models.

# of bits	$\#M$		$\#M$/bit		Improvement (%)
	Binary	Modified greedy	Binary	Modified greedy	
$(N = 17^3)$	$108.5M$	$123M$	8.35	9.46	−13.36
$(N = 19^3)$	$112.75M$	$128M$	8.67	9.85	−13.53
160	$1990.1M$	$1559.6M$	12.44	9.75	+21.63
224	$2787.9M$	$2168.4M$	12.45	9.68	+22.22
256	$3198.9M$	$2480.3M$	12.5	9.69	+22.46
384	$4847.3M$	$3711.8M$	12.62	9.67	+23.43
512	$6370.3M$	$5020.9M$	12.44	9.81	+21.18

that improves by more than 20% for large scalars, i.e., above 160 bits in scalar multiplication, which is comparable to that of Al Musa [17].

5. Conclusion

In this paper, we first use double-base addition-subtraction chains. We also use twisted Edwards curves with projective coordinates to improve the addition and subtraction calculations in the chain. Our algorithm generates addition-subtraction chains using double base to improve the total complexity. The memory used by our algorithm to execute scalar multiplication is better than the result of the tree-based approach [17]. This memory efficiency leads to efficient scalar multiplication. That is, it can also be applied to efficient scalar multiplexing in isogeny-based cryptosystems. We also show experimental results demonstrating that the length of the chain is much shorter; therefore, for scalar

multiplication in an isogeny-based cryptosystem, the complexity is reduced by 20% compared to the single-base number system. These results are also expected to contribute to research improving the efficiency of SIDH and SIKE-based protocols, which comprise one of the candidate groups for post-quantum cryptosystems. We can also consider implementing it in terms of efficiency in multi-system and network environments such as those in [30–33].

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

Sookyung Eom was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (nos. 2019R1C1C1002403, 2019R1A6A1A11 051177, and 2019R1A6A1A11051177). Hyang-Sook Lee was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) (no. 2018R1A2A1A05079095). Kyunghwan Song was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (no. 2020R1I1A1A01070546).

References

[1] J. M. Couveignes, "Hard homogeneous spaces," *IACR Cryptology ePrint Archive*, vol. 291, 2006.

[2] D. X. Charles, K. E. Lauter, and E. Z. Goren, "Cryptographic hash functions from expander graphs," *Journal of Cryptology*, vol. 22, no. 1, pp. 93–113, 2009.

[3] A. Childs, D. Jao, and V. Soukharev, "Constructing elliptic curve isogenies in quantum subexponential time," *Journal of Mathematical Cryptology*, vol. 8, no. 1, pp. 1–29, 2014.

[4] L. Feo and D. Jao, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," in *Proceedings of the PQCrypto 2011 International Workshop on Post-Quantum Cryptography*, pp. 19–34, Springer, Taipei, Taiwan, November 2011.

[5] G. H. M. Zanon, "Faster key compression for isogeny-based cryptosystems," *IEEE Transactions on Computers*, vol. 68, pp. 688–701, 2019.

[6] C. Costello, D. Jao, P. Lango, M. Naehrig, J. Renes, and D. Urbanik, "Efficient compression of SIDH public keys," in *Proceedings of the EUROCRYPT 2017 Advances in Cryptology*, pp. 679–706, Paris, France, May 2017.

[7] S. Galvraith, C. Petit, B. Shanim, and Y. B. Ti, "On the security of supersingular isogeny cryptosystems," in *Proceedings of the ASIACRYPT 2016 International Conference on the Theory and Application of Cryptology and Information Security*, pp. 63–91, Springer, December 2016.

[8] C. Costello, P. Longa, and M. Naehrig, "Efficient algorithms for supersingular isogeny Diffie-Hellman," in *Proceedings of the Advances in Cryptology - CRYPTO 2016*, pp. 572–601, Barbara, CA, USA, August 2016.

[9] D. Jao, "Supersingular isogeny key encapsulation," *Submission to the NIST Post-Quantum Standardization project*, 2017.

[10] H. Seo, A. Jalali, and R. Azarderakhsh, "Optimized SIKE round 2 on 64-bit ARM," *Cryptology ePrint Archive*, vol. 67, no. 8, pp. 2659–2671, 2019.

[11] B. Koziel, A. B. Ackie, R. El Khatib, R. Azarderakhsh, and M. M. Kermani, "SIKE'd up: fast and secure hardware architectures for supersingular isogeny key encapsulation," *Cryptology ePrint Archive*, vol. 67, no. 12, pp. 129–146, 2019.

[12] H. Seo, A. Jalali, and R. Azarderakhsh, "SIKE Round 2 speed record on ARM Cortex-M4," *Cryptology and Network Security*, pp. 39–60, 2019.

[13] S. Jaques and J. M. Schanck, "Quantum cryptanalysis in the RAM model: claw-finding attacks on SIKE," in *Proceedings of the Advances in Cryptology - CRYPTO 2019*, pp. 32–61, Santa Barbara, CA, USA, August 2019.

[14] J. W. Bos and S. J. Friedberger, "Faster modular arithmetic for isogeny-based crypto on embedded devices," *Journal of Cryptographic Engineering*, vol. 10, pp. 1–13, 2019.

[15] B. Koziel, R. Azarderakhsh, D. Jao, and M. M. Kermani, "On fast calculation of addition chains for isogeny-based cryptography," in *Proceedings of the Inscrypt 2016, Information Security and Cryptology*, pp. 323–342, Springer, Beijing, China, November 2016.

[16] J. W. Bos and S. J. Friedberger, "Arithmetic considerations for isogeny-based cryptography," *IEEE Transactions on Computers*, vol. 68, pp. 979–990, 2018.

[17] S. Al Musa, *Multi-base chains for faster elliptic curve cryptography*, University of Wisconsin, Milwaukee, WA, USA, 2018.

[18] W. Yu, S. Al Musa, and B. Li, "Double-base chains for scalar multiplications on elliptic curves," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, Zagreb, Croatia, May 2020.

[19] L. Washington, *Elliptic Curves: Number Theory and Cryptography*, Chapman&Hall/CRC, Boca Raton, FL, USA, 2nd edition, 2008.

[20] S. Galbraith, C. Petit, and J. Silva, "Signature schemes based on supersingular isogeny problems," in *Proceedings of the AsiaCrypt 2017 Advances in Cryptology*, pp. 3–33, Springer, Hong Kong, China, November 2017.

[21] G. Bisson and A. V. Sutherland, "Computing the endomorphism ring of an ordinary elliptic curve over a finite field," *Journal of Number Theory*, vol. 131, no. 5, pp. 815–831, 2011.

[22] D. Vassil, I. Laurent, and M. Pradeep, "The double-base number system and its application to elliptic curve cryptography," *Mathematics of Computation*, vol. 77, pp. 1075–1104, 2008.

[23] C. Doche and L. Imbert, "Extended double-base number system with applications to elliptic curve cryptography," in *Proceedings of the INDOCRYPT 2006 7th International Conference on Cryptology in India*, pp. 335–348, Springer, Kolkata, India, December 2006.

[24] D. J. Bernstein et al., "Twisted Edwards curves," in *Proceedings of the International Conference on Cryptology in Africa*, pp. 389–405, Springer, Casablanca, Morocco, June 2008.

[25] F. Najman, "The number of twists with large torsion of an elliptic curve, revista de la real academia de ciencias exactas, físicas y naturales," vol. 109, no. 2, pp. 535–547, 2015.

[26] D. J. Bernstein and T. Lange, "Explicit-formulas database," 2009, http://hyperelliptic.org/EFD/g1p/index.html.

[27] D. Knuth, "The art of computer programming," *Google Scholar Digital Library*, vol. 3, 1998.
[28] D. M. Gordon, "A survey of fast exponentiation methods," *Journal of Algorithms*, vol. 27, no. 1, pp. 129–146, 1998.
[29] C. Doche and L. Habsieger, "A tree-based approach for computing double-base chains," in *Proceedings of the Australasian Conference on Information Security and Privacy*, pp. 433–446, Springer, Wollongong, Australia, July 2008.
[30] X. Lu and H. Li, "An improved stability theorem for nonlinear systems on time scales with application to multi-agent systems," *IEEE Transactions on Circuits and Systems*, vol. 67, pp. 3277–3281, 2020.
[31] X. Lu and H. Li, "A hybrid control approach to Hinfin; problem of nonlinear descriptor systems with actuator saturation," *IEEE Transactions on Automatic Control*, vol. 66, pp. 4960–4966, 2020.
[32] H. Li, X. Yang, and S. Wang, "Robustness for stability and stabilization of Boolean networks with stochastic function perturbations," *Information Sciences*, vol. 582, pp. 833–849, 2022.
[33] H. Li, S. Wang, X. Li, and G. Zhao, "Perturbation analysis for controllability of logical control networks," *SIAM Journal on Control and Optimization*, vol. 58, pp. 3632–3657, 2020.