Vol.2020-CSEC-91 No.33
Vol.2020-SPT-40 No.33
Vol.2020-EIP-90 No.33
2020/11/26

2-ADD-Skip methodを用いた高速な同種写像計算アルゴリズムの改良

小寺 健太[1,a)]　鄭 振牟[2,b)]　宮地 充子[1,3,c)]

概要：同種写像暗号は耐量子暗号の候補の 1 つとして注目を浴びている．例えば commutative supersingular isogeny Diffie-Hellman（CSIDH）は非常に小さい公開鍵長をもつ鍵共有手法として知られている．しかし一方で他の手法と比べて実行速度が遅いという課題がある．そこで本研究では CSIDH において主要な計算である，同種写像計算に着目することで高速化を目指す．SCIS2020 において Kodera らは 2-ADD-Skip method を提案し，同種写像計算に必要となる楕円曲線上の加算の回数を削減した．本研究では，より効率的に 2-ADD-Skip method を活用する同種写像計算アルゴリズムを提案する．演算量の解析により Meyer らのアルゴリズムと比較しておよそ 12%高速であることを示す．さらに計算機実験により同種写像の次数 ℓ が $19 \leq \ell \leq 373$ を満たすとき，提案手法は Bernstein らの $\tilde{O}(\sqrt{\ell})$ のアルゴリズムよりも高速であることを示す．

キーワード：耐量子暗号，同種写像，Montgomery 曲線

1. はじめに

1.1 背景

現在広く用いられている RSA 暗号や楕円曲線暗号などの公開鍵暗号には，量子計算機を用いた多項式時間の攻撃手法が存在する．そこで近年，耐量子暗号と呼ばれる量子計算機を用いた攻撃に耐えうる暗号の研究が盛んに行われている．同種写像暗号は耐量子暗号の候補の 1 つであり，supersingular isogeny Diffie–Hellman（SIDH）[1] や commutative SIDH (CSIDH) [2] などが知られている．特に SIDH を元に作られた鍵カプセル化アルゴリズムに supersingular isogeny key encapsulation (SIKE) がある [3]．SIKE は米国標準技術研究所（NIST）が推進する耐量子暗号の標準化プロジェクトにおいて第 3 ラウンドの代替候補に残っており，候補内で最小の公開鍵長を持つ [4]．一方で 2018 年に提案された CSIDH は SIDH よりも更に小さい公開鍵長を持つことが知られている．しかしながら実行速度に課題があり，高速化手法の研究が必要となっている．

CSIDH は秘密鍵に応じた様々な奇数次数の同種写像を計算することで鍵共有を行う．CSIDH のアルゴリズムではスカラー倍算による同種写像の核の生成元の計算と，その生成元を用いた同種写像の計算が繰り返し行われる．Meyer らは同種写像の次数が大きいものから計算することでスカラー倍算の量を削減できることを示した [5]．また Meyer らは Elligator map [6] を活用するだけでなく，同種写像を次数によってグループ分けする SIMBA という手法を提案し生成元計算のコストを更に削減した [7]．さらに Hutchnson らは SIDH における strategy の概念を適用し，線形計画法などを用いて解析することで効率的なアルゴリズムを提案した [8]．秘密鍵空間の定義方法も注目されており，Meyer らが同種写像の次数の大きさに対応した重み付けを提案したほか [7]，Nakagawa らによって L_1 ノルム空間を用いることで高速に計算できることが示された [9]．

一方で，スカラー倍算や同種写像の計算自身を高速化する方針でも様々な研究が行われている．例えば Cervantes-Vázquez らは最適化された addition chain の組み合わせによってスカラー倍算を効率よく計算する手法を提案した [10]．ここで同種写像 $\phi : E \longrightarrow E'$ は 1．$\ker(\phi)$ に含まれる点を求める点計算，2．与えられた点 P について $\phi(P)$ を求める像計算，3．曲線 E' の係数計算の 3 つから構成される．Meyer らは twisted Edwards 曲線を用いることで係

[1] 大阪大学
Osaka University
[2] 金沢大学
Kanazawa University
[3] 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology
[a)] kodera@cy2sec.comm.eng.osaka-u.ac.jp
[b)] cheng@se.kanazawa-u.ac.jp
[c)] miyaji@comm.eng.osaka-u.ac.jp

ⓒ 2020 Information Processing Society of Japan

Vol.2020-CSEC-91 No.33
Vol.2020-SPT-40 No.33
Vol.2020-EIP-90 No.33
2020/11/26

数計算に要する演算量を削減した [5]．また Bernstein らは $\ker(\phi)$ の点計算により多くの 2 倍算を用いる手法を提案し，特定の条件下で演算量が削減できることを示した [11]．通常，次数 $\ell = 2d+1$ の同種写像においては，$\ker(\phi)$ のうち d 個の点計算を行う必要がある．しかし，より少数の点計算で同種写像を計算するアルゴリズムが Kodera ら [12] および Bernstein ら [13] によって独立に提案された．Kodera らは 2-ADD-Skip method と呼ばれる手法およびそれを活用するアルゴリズムを提案し，点計算の個数をおよそ 1/3 に削減した．一方 Bernstein らは終結式を活用することで，およそ $2\sqrt{\ell}$ 個の点計算を元に計算量 $\tilde{O}(\sqrt{\ell})$ のアルゴリズムを提案した．

また単純電力解析（SPA）を始めとするサイドチャネル攻撃に対して耐性を持たせるために，公開鍵計算のアルゴリズムを constant-time 化する研究も盛んである．例えば dummy 同種写像の提案や [7]，2 種の核の生成元を保持することで効率化を行った研究 [14]，さらにはより強い攻撃耐性に関する提案 [10] などが挙げられる．

本稿では同種写像暗号の高速化を目指し，主要な計算である同種写像の計算の高速化に取り組む．特に点計算の個数を削減するために Kodera らによって提案された 2-ADD-Skip method に注目する．2-ADD-Skip method を利用する既存の同種写像の計算アルゴリズムを改良することで，より効率的に同種写像を計算する手法を提案する．

以降の本稿の構成を示す．まず 2 章で楕円曲線やその加法公式を示し，3 章で同種写像計算と既存のアルゴリズムをまとめる．4 章で提案手法を示し，体上の演算量について解析する．5 章では計算機実験により提案手法を評価する．最後に 6 章で本稿を結論付ける．

1.2 成果

本稿では 2-ADD-Skip method を用いる同種写像計算アルゴリズムの改良に取り組む．既存のアルゴリズムにおける 2-ADD-Skip method の活用方法を拡張し，新たに定義されたパラメータ n を持つアルゴリズムを提案する．演算量の解析を行い，同種写像の次数 ℓ に対して最適である n の値を導出する．

さらに提案したアルゴリズムの性能評価のために，計算機実験を通して Meyer らのアルゴリズムおよび Bernstein らのアルゴリズムとの比較を行う．漸近的な計算量では Bernstein らのアルゴリズムに劣るものの，次数が $19 \leq \ell \leq 373$ を満たす同種写像においては提案したアルゴリズムが最も効率が良いことを実験的に示す．また上記 3 種の同種写像計算アルゴリズムを CSIDH-512 に適用した実験を行う．その結果，提案したアルゴリズムを用いた場合の計算量が最小であり，Meyer らのアルゴリズムを用いた場合に比べておよそ 5.2% の計算量を削減できることを明らかにする．

2. 準備

2.1 Montgomery 曲線

K を体とする．K の標数が 2 でないとき，Montgomery 曲線とは，$a, b \in K, b(a^2-4) \neq 0$ について

$$M_{a,b}: \quad by^2 = x^3 + ax^2 + x$$

で与えられる [15]．無限遠点を $\mathcal{O}$，スカラー倍算を $[k]P$ と表記する．

Costello らの論文 [16] に従って，射影空間における Montgomery 曲線を以下で表す．

$$M_{A,B,C}: \quad BY^2Z = CX^3 + AX^2Z + CXZ^2$$

ここで $(A,B,C),\ (X,Y,Z) \in \mathbb{P}^2(K),\ C \neq 0,\ Z \neq 0,\ a = A/C,\ b = B/C,\ x = X/Z,\ y = Y/Z$ である．Montgomery 曲線には $\mathbb{P}^1(K)$ 上の点 $(X : Z)$ に関して効率のよい加法が存在する [15]．特に $C \neq 1$ となる場合は，その加法公式が以下で与えられる．$\mathbb{P}^1(K)$ 上の点を $(X_P : Z_P), (X_Q : Z_Q)$ とする．

- $P \neq Q$ のとき

$$\begin{aligned} X_{P+Q} &= Z_{P-Q}(X_PX_Q - Z_PZ_Q)^2 \\ Z_{P+Q} &= X_{P-Q}(X_PZ_Q - Z_PX_Q)^2 \end{aligned} \tag{1}$$

- $P = Q$ のとき

$$\begin{aligned} X_{[2]P} &= 4C(X_P+Z_P)^2(X_P-Z_P)^2, \\ Z_{[2]P} &= (4X_PZ_P)(4C(X_P-Z_P)^2(A+2C)(4X_nZ_n)), \\ 4X_PZ_P &= (X_P+Z_P)^2 - (X_P-Z_P)^2. \end{aligned} \tag{2}$$

式 (1), (2) についてそれぞれ関数を定義する．

$$\begin{aligned} &\texttt{ADD} : ((X_P : Z_P), (X_Q : Z_Q), (X_{P-Q} : Z_{P-Q})) \\ &\qquad \longrightarrow (X_{P+Q} : Z_{P+Q}), \\ &\texttt{DBL} : ((X_P : Z_P), (A : C)) \longrightarrow (X_{[2]P} : Z_{[2]P}). \end{aligned}$$

K 上の乗算，2 乗算，加算の計算量について，それぞれ $\mathbf{M}, \mathbf{S}, \mathbf{a}$ で表すと，ADD および DBL はそれぞれ $4\mathbf{M}+2\mathbf{S}+6\mathbf{a}$ および $4\mathbf{M}+2\mathbf{S}+8\mathbf{a}$ の演算で計算できる [5]．

2.2 楕円曲線間の同種写像

E および E' を楕円曲線とする．同種写像 $\phi : E \longrightarrow E'$ とは，$\phi(\mathcal{O}_\mathcal{E}) = \mathcal{O}_{E'}$ を満たす有理写像である．ϕ が分離的で $\ker(\phi)$ が位数 ℓ の巡回群であるとき ℓ-同種写像と表す．Φ を楕円曲線 E の有限部分群とする．このとき $\ker(\phi) = \Phi$ を満たす分離的な同種写像 $\phi : E \longrightarrow E'$ および楕円曲線 E' が同型を除いて一意に存在する．また E および Φ が与えられたとき，Vélu の公式によって ϕ および E' を計算できる [17]．本稿では，E 上の点 P について $\phi(P)$ を求める

ⓒ 2020 Information Processing Society of Japan

Vol.2020-CSEC-91 No.33
Vol.2020-SPT-40 No.33
Vol.2020-EIP-90 No.33
2020/11/26

ことを**像計算**，また E' の係数を計算することを**係数計算**という．また**同種写像計算**とは，像計算および係数計算を同時に行う計算を表す．

3. 既存研究

本章では Montgomery 曲線における奇数次数 $\ell = 2d+1$ の同種写像計算アルゴリズムについてまとめる．同種写像を $\phi : M_{A,B,C} \longrightarrow M_{A',B',C'}$ とおく．まず $\ker(\phi) = \langle P \rangle$ について $(X_i : Z_i) := \varphi_x([i]P)$ と定義する．さらに像計算について $(X : Z) \in M_{A,B,C}$ および $(X' : Z') := \phi((X : Z)) \in M_{A',B',C'}$ の表記を用いる．係数計算については曲線の係数を $(A : C), (A' : C')$ で表す．すなわち同種写像計算とは $(X_1 : Z_1), (A : C), (X : Z)$ が与えられ $(A' : C'), (X' : Z')$ を求める処理を指す．

Montgomery 曲線における同種写像計算は，Costello-Hisil の公式 [18] を用いて行われる．通常のアルゴリズムでは加法公式を用いて $(X_2 : Z_2), \ldots, (X_d : Z_d)$ の $d-1$ 点の座標がすべて計算される．この過程を点計算と呼び，1 度の 2 倍算と $d-2$ 回の加算が実行される．既存の改良手法として twisted Edwards 曲線を用いて係数計算を高速化する手法 [5]，点計算より多くの 2 倍算によって計算する手法 [11] を述べる．さらに，より少数の点計算から同種写像を計算する手法として，2-ADD-Skip method を用いた手法 [12]，および終結式を利用し $\tilde{O}(\sqrt{\ell})$ での計算を実現した手法 [13] を述べる．

3.1 Costello-Hisil の公式

Costello，Hisil らは Vélu の公式を元に Montgomery 曲線上の奇数次数同種写像の公式を導出した [18].

K を標数が 2 でない体とし，K 上の Montgomery 曲線を $M_{a,b} : by^2 = x^3 + ax^2 + x$，$P$ を位数 $\ell = 2d+1$ の点とする．さらに $\sigma = \sum_{i=1}^{d} x_{[i]P}, \tilde{\sigma} = \sum_{i=1}^{d} 1/x_{[i]P}, \pi = \prod_{i=1}^{d} x_{[i]P}$ とおく．ここで $x_{[i]P}$ は $[i]P$ の x 座標を表す．このとき $\ker(\phi) = \langle P \rangle$ を満たす Montgomery 曲線間の ℓ-同種写像 $\phi : M_{a,b} \longrightarrow M_{a',b'}$ は以下で与えられる．

$$a' = (6\tilde{\sigma} - 6\sigma + a)\pi^2, \quad b' = b\pi^2, \tag{3}$$

$$\phi : (x, y) \longmapsto (f(x), yf'(x)),$$
$$f(x) = x \cdot \prod_{i=1}^{d} \left(\frac{x \cdot x_{[i]P} - 1}{x - x_{[i]P}} \right)^2, \tag{4}$$

ここで $f'(x)$ はその微分を表す．

なお Renes は点 $(0,0)$ との加算表現を用いることで Vélu の公式を変換し，Costello-Hisil らの公式と等価な式を導出した [19]．その結果，核が奇数位数巡回群であるという制約を $(0,0) \notin \ker(\phi)$ に緩和することに成功した．

射影空間 $\mathbb{P}^1(K)$ における計算公式は，上式に $x = X/Z, x_{[i]P} = X_i/Z_i, a = (A : C)$ を適用することで得られる．像計算式 (4) は式 (5) で与えられる．

$$(X' : Z') = (X \cdot (S_X)^2 : Z \cdot (S_Z)^2), \tag{5}$$

$$S_X = \prod_{i=1}^{d} (XX_i - Z_iZ),\ S_Z = \prod_{i=1}^{d} (XZ_i - X_iZ). \tag{6}$$

同様に，係数計算式 (3) は式 (7) で与えられる．

$$(A' : C') = (\tau(A - 3\sigma) : C) \tag{7}$$

なお $\tau = \prod_{i=1}^{\ell-1} \frac{X_i}{Z_i}$ および $\sigma = \sum_{i=1}^{\ell-1} \left(\frac{X_i}{Z_i} - \frac{Z_i}{X_i} \right)$ である．

3.2 係数計算における改善手法

まず Castryck らは $(A' : C')$ を効率的に計算するために新たな変数 T_i を用いた式変形を行った [2]．T_i は次の $\ell - 1$ 次多項式の係数によって定義された．$\sum_{i=0}^{\ell-1} T_i w^i = \prod_{i=1}^{\ell-1} (Z_i w + X_i)$．すなわち $T_0 = \prod_i X_i$, $T_1 = \sum_i (Z_i \prod_{j \neq i} X_j)$, $T_{\ell-2} = \sum_i (X_i \prod_{j \neq i} Z_j)$, $T_{\ell-1} = \prod_i Z_i$ を用いて，式 (8) を計算する．

$$(A' : C') = \left(AT_0T_{\ell-1} - 3C(T_0T_{\ell-2} - T_1T_{\ell-1}) : CT_{\ell-1}^2 \right). \tag{8}$$

Meyer，Reith らは twisted Edwards 曲線を用いて係数計算を行うことでさらに演算量を削減した [5]．体 K 上の twisted Edwards 曲線は係数 $a_{tE}, d_{tE} \in K$, $a_{tE}d_{tE} \neq 0$, $a_{tE} \neq d_{tE}$, $d_{tE} \neq 1$ を用いて $tE_{a_{tE},d_{tE}} : a_{tE}u^2 + v^2 = 1 + d_{tE}u^2v^2$ で与えられる [20]．Montgomery 曲線と twisted Edwards 曲線の間には同型写像が存在し，体上の加算のみで曲線の形を相互変換できる．

$$\begin{aligned} &a_{tE} = A + 2C,\ d_{tE} = A - 2C, \\ &(A : C) = (2(a_{tE} + d_{tE}) : a_{tE} - d_{tE})\,. \end{aligned} \tag{9}$$

さらに $\mathbb{P}^1(K)$ における Montgomery 曲線上の点 $(X : Z)$ は式 (10) によって対応する twisted Edwards 曲線の YZ 座標上の点 $(Y_{tE} : Z_{tE})$ に変換できる．

$$(X : Z) \longmapsto (Y_{tE} : Z_{tE}) = (X - Z : X + Z). \tag{10}$$

twisted Edwards 曲線における ℓ-同種写像の公式は Moody，Shumow らによって導出された [21]．特に YZ 座標上においては式 (11) および (12) で与えられる．

$$a'_{tE} = a_{tE}^{\ell} \cdot \pi_Z^8,\ d'_{tE} = d_{tE}^{\ell} \cdot \pi_Y^8. \tag{11}$$

$$\pi_Z = \prod_{i=1}^{d} Z_{tE,i}\ \pi_Y = \prod_{i=1}^{d} Y_{tE,i}. \tag{12}$$

式 (10) および (9) による変換が軽量であり，式 (8) に比べて式 (11) に要する演算量が小さいため係数計算を高速化できる．

ⓒ 2020 Information Processing Society of Japan

Vol.2020-CSEC-91 No.33
Vol.2020-SPT-40 No.33
Vol.2020-EIP-90 No.33
2020/11/26

Algorithm 1 同種写像計算アルゴリズム [5]

Input: ℓ, $(X:Z)$, $(X_1:Z_1)$, and $(A:C)$
Output: $(X':Z')$ and $(A':C')$
1: $(\pi_Y,\pi_Z)\leftarrow(X_1-Z_1,X_1+Z_1)$ // **2a**
2: $(t^+,t^-)\leftarrow(X+Z,X-Z)$ // **2a**
3: $(t_0,t_1)\leftarrow(t^-\cdot\pi_Z,t^+\cdot\pi_Y)$ // **2M**
4: $(S_X:S_Z)\leftarrow(t_0+t_1,t_0-t_1)$ // **2a**
5: **for** $i=2$ to $(\ell-1)/2$ **do**
6: **if** $i==2$ **then**
7: $(X_i:Z_i)\leftarrow\mathtt{DBL}((X_1:Z_1),(A:C))$ // $4\mathbf{M}+2\mathbf{S}+8\mathbf{a}$
8: **else**
9: $(X_i:Z_i)\leftarrow\mathtt{ADD}((X_{i-1}:Z_{i-1}),(X_1:Z_1),(X_{i-2}:Z_{i-2}))$ // $4\mathbf{M}+2\mathbf{S}+6\mathbf{a}$
10: **end if**
11: $(S_X,S_Z,\pi_Y,\pi_Z)\leftarrow\mathtt{UPDATE}(X_i:Z_i)$ // $6\mathbf{M}+4\mathbf{a}$
12: **end for**
13: $(X':Z')\leftarrow(X\cdot(S_X)^2:Z\cdot(S_Z)^2)$ // $2\mathbf{M}+2\mathbf{S}$
14: $(a'_{tE},d'_{tE})\leftarrow\left((A+2C)^\ell\cdot\pi_Z^8,(A-2C)^\ell\cdot\pi_Y^8\right)$ // $(2+\tilde{\ell})\mathbf{M}+(2\tilde{\ell}+6)\mathbf{S}+3\mathbf{a}$
15: $(A':C')\leftarrow(2(a'_{tE}+d'_{tE}),a'_{tE}-d'_{tE})$ // **3a**
16: **return** $(X':Z')$ and $(A':C')$

3.3 Montgomery 曲線上の同種写像計算アルゴリズム

同種写像計算のアルゴリズムとして像計算式 (5) および係数計算式 (11) を用いる Meyer らのアルゴリズム（Algorithm 1）について述べる．まず中間変数である S_X, S_Z, π_Y, π_Z を計算する．式 ((6)) および (12) が示すように，これらの中間変数は $(X:Z)$ および $(X_i:Z_i)$ に関する多項式である．したがって，$i=1,\dots,d$ についての繰り返し処理を用いて計算できる．

はじめに $(X_1:Z_1),(X:Z)$ を用いて全ての中間変数を初期化する（1-4 行目）．次に $i=2,\dots,d$ について加法公式によって点 $(X_i:Z_i)$ が計算され（6-10 行目）次式に従って中間変数の更新が行われる（11 行目）．

$$\begin{cases} S_X\leftarrow S_X\cdot((X-Z)\cdot(X_i+Z_i)+(X+Z)\cdot(X_i-Z_i)),\\ S_Z\leftarrow S_Z\cdot((X-Z)\cdot(X_i+Z_i)-(X+Z)\cdot(X_i-Z_i)); \end{cases} \tag{13}$$

$$\begin{cases} \pi_Y\leftarrow\pi_Y\cdot(X_i-Z_i),\\ \pi_Z\leftarrow\pi_Z\cdot(X_i+Z_i). \end{cases} \tag{14}$$

ここで式 (13)，(14) の計算に対応する関数を定義する．

$$\mathtt{UPDATE}:(S_X,S_Z,\pi_Y,\pi_Z,(X+Z),(X-Z),(X_i:Z_i)) \longmapsto (S'_X,S'_Z,\pi'_Y,\pi'_Z).$$

以降では簡潔さの観点から $\mathtt{UPDATE}((X_i:Z_i))$ と表記する．この関数は $(4\mathbf{M}+4\mathbf{a})+2\mathbf{M}=6\mathbf{M}+4\mathbf{a}$ の演算によって計算できる．これは $X_i\pm Z_i$ の計算は 1 度でよく，また $X\pm Z$ の値を事前に計算できるからである（2 行目）．最後に中間変数を元に式 (5) および (11) を用いて $(X':Z'),(A':C')$ を計算する（13-15 行目）．

以上よりアルゴリズム全体では $(10d+\tilde{\ell}-4)\mathbf{M}+(2d+2\tilde{\ell}+6)\mathbf{S}+(10d+3)\mathbf{a}$ の演算が必要となる．ここで $\tilde{\ell}$ は ℓ のビット長を表し，ℓ 乗の計算には $(\tilde{\ell}/2)\mathbf{M}+\tilde{\ell}\mathbf{S}$ の演算が必要となることを仮定している．

なお $Z_1=1$ もしくは $C=1$ を満たす場合，ADD もしくは DBL の計算に必要となる乗算回数が減る．Bernstein らはこの事実から，核に含まれる $d-1$ 点の計算方法を工夫し演算量を削減することを提案した [11]．例えば Montgomery ladder のような手法で計算することで DBL の実行回数を増やすことができる．しかし，加法公式 (1)，(2) の Z 座標や同種写像の係数計算式（11）の C' の式が示すように一般には $Z_1\neq1$ かつ $C\neq1$ となる．したがって上記の工夫が有効となるのは逆元計算を行う場合となるが，本稿では議論しない．

3.4 2-ADD-Skip method による高速化

Algorithm 1 が示すように，通常のアルゴリズムでは $i=2,\dots,d$ について核に含まれる点 $(X_i:Z_i)$ の座標が加法公式によって逐次計算され，UPDATE の引数として利用される．Kodera らは 2-ADD-Skip method と呼ばれる更新手法を提案し，アルゴリズムにおける点計算の個数を削減した [12]．

2-ADD-Skip method とは，ある $m\neq n$ について $(X_{m+n}:Z_{m+n})$ および $(X_{m-n}:Z_{m-n})$ の 2 点に関する更新を $(X_m:Z_m)$ および $(X_n:Z_n)$ の 2 点の座標から計算する手法である．すなわち，2 点 $(X_{m+n}:Z_{m+n})$ および $(X_{m-n}:Z_{m-n})$ の点計算を省略できる．

まず，ある $m\neq n$ について $X_{m+n}X_{m-n}, Z_{m+n}Z_{m-n}, X_{m+n}Z_{m-n}+X_{m-n}Z_{m+n}$ の 3 つの値を式 (15) によって計算する．

$$\begin{cases} X_{m+n}X_{m-n}=C(X_nX_m-Z_nZ_m)^2,\\ Z_{m+n}Z_{m-n}=C(X_nZ_m-X_mZ_n)^2,\\ X_{m+n}Z_{m-n}+X_{m-n}Z_{m+n}\\ \quad=2C(X_nZ_m+X_mZ_n)(X_nX_m+Z_nZ_m)\\ \qquad+4AX_nX_mZ_nZ_m. \end{cases} \tag{15}$$

なお次の式変形により，式 (15) は $9\mathbf{M}+3\mathbf{S}+7\mathbf{a}$ の演算によって計算できる．

$$4AX_nX_mZ_nZ_m=A\left((X_nX_m+Z_nZ_m)^2-(X_nX_m-Z_nZ_m)^2\right).$$

次に，式 (16) および (17) を用いて像計算および係数計算に関する更新を計算する．

$$\begin{cases} S_X\leftarrow S_X\cdot((X^2)\cdot X_iX_j-(XZ)\cdot(X_iZ_j+X_jZ_i)\\ \qquad\qquad+(Z^2)\cdot Z_iZ_j),\\ S_Z\leftarrow S_Z\cdot((X^2)\cdot Z_iZ_j-(XZ)\cdot(X_iZ_j+X_jZ_i)\\ \qquad\qquad+(Z^2)\cdot X_iX_j); \end{cases} \tag{16}$$

$$\begin{cases} \pi_Y\leftarrow\pi_Y\cdot(X_iX_j+Z_iZ_j-(X_iZ_j+X_jZ_i)),\\ \pi_Z\leftarrow\pi_Z\cdot(X_iX_j+Z_iZ_j+(X_iZ_j+X_jZ_i)). \end{cases} \tag{17}$$

ⓒ 2020 Information Processing Society of Japan

Vol.2020-CSEC-91 No.33
Vol.2020-SPT-40 No.33
Vol.2020-EIP-90 No.33
2020/11/26

表 1 2 点 $(X_i : Z_i), (X_j : Z_j)$ に関する点計算と更新の演算量の比較

	(i) 既存手法	(ii) 2-ADD-Skip method	差分 (i)-(ii)
$(X_i : Z_i), (X_j : Z_j)$ の点計算 もしくは $X_iX_j, Z_iZ_j, X_iZ_j + X_jZ_i$ の計算	$8\mathbf{M} + 4\mathbf{S} + 12\mathbf{a}$	$9\mathbf{M} + 3\mathbf{S} + 7\mathbf{a}$	$-1\mathbf{M} + 1\mathbf{S} + 5\mathbf{a}$
像計算 に関する更新	$8\mathbf{M} + 8\mathbf{a}$	$7\mathbf{M} + 4\mathbf{a}$	$1\mathbf{M} + 4\mathbf{a}$
係数計算 に関する更新	$4\mathbf{M}$	$2\mathbf{M} + 3\mathbf{a}$	$2\mathbf{M} - 3\mathbf{a}$
合計	$20\mathbf{M} + 4\mathbf{S} + 20\mathbf{a}$	$18\mathbf{M} + 3\mathbf{S} + 14\mathbf{a}$	$2\mathbf{M} + 1\mathbf{S} + 6\mathbf{a}$

いま，X^2, XZ, Z^2 がアルゴリズムの冒頭で計算済みであるとすると，式 (16) および (17) が要する演算量はそれぞれ $7\mathbf{M} + 4\mathbf{a}$ および $2\mathbf{M} + 3\mathbf{a}$ となる．

UPDATE に対応する関数として，式 (16), (17), (15) をまとめたものを 2ADDSKIP と定義する．

$$\texttt{2ADDSKIP}: (S_X, S_Z, \pi_Y, \pi_Z, X^2, XZ, Z^2, (X_m : Z_m), (X_n : Z_n)) \longmapsto (S'_X, S'_Z, \pi'_Y, \pi'_Z)$$

簡単のため，$(X_{m+n} : Z_{m+n})$ および $(X_{m-n} : Z_{m-n})$ の 2 点に関する更新を $\texttt{2ADDSKIP}((X_m : Z_m), (X_n : Z_n))$ で表す．

表 1 は 2 点に関する点計算と更新に必要な演算量をまとめたものである．2-ADD-Skip method を用いるごとに $2\mathbf{M} + 1\mathbf{S} + 6\mathbf{a}$ の演算を削減できることが分かる．

Kodera らは 2-ADD-Skip method を用いた同種写像計算アルゴリズムとして Algorithm 2 を提案した．同種写像を正しく計算するためには，点 $(X_i : Z_i)$ に関する更新を $i = 2, \ldots, d$ について漏れなく重複なく行う必要がある．$\texttt{UPDATE}((X_m : Z_m))$ および $\texttt{2ADDSKIP}((X_m : Z_m), (X_1 : Z_1))$ によって $i = m-1, m, m+1$ の連続した 3 点に関する更新が計算できる．さらに $((X_{m+3} : Z_{m+3}))$ について同様な処理を行うことで連続した 6 点に関する更新が計算できる．したがって，m の初期値を d によって適切に決定することで漏れなく重複なく更新を計算できる．すなわち，整数 $q \geq 0, 0 \geq r < 3$ について $d = 3q + r + 1$ と書けるとき，$m = 3 + r$ を初期値とすればよい．Algorithm 2 は q 回の 2-ADD-Skip method を実行するため，Algorithm 1 に比べておよそ $\lfloor (\ell - 2)/6 \rfloor (2\mathbf{M} + 1\mathbf{S} + 6\mathbf{a})$ の演算を削減できる．

3.5 $\tilde{O}(\sqrt{\ell})$ での同種写像計算

Bernstein らは次数 ℓ の同種写像 ϕ を $\tilde{O}(\sqrt{\ell})$ で計算する手法を提案した [13]．$\ker \phi = \langle P \rangle$ について核多項式は

$$\Psi(X) = \prod_{i=1}^{(\ell-1)/2} (X - x_{[i]P})$$

で与えられる．式 (4) が

$$f(x) = \frac{x^\ell \cdot \Psi(1/x)^2}{\Psi(x)^2}$$

Algorithm 2 2-ADD-Skip method を用いた同種写像計算アルゴリズム [12]

Input: $\ell \geq 9$, $(X : Z)$, $(X_1 : Z_1)$, and $(A : C)$
Output: $(X' : Z')$ and $(A' : C')$

$(\pi_Y, \pi_Z) \leftarrow (X_1 - Z_1, X_1 + Z_1)$ // $2\mathbf{a}$
$(t^+, t^-) \leftarrow (X + Z, X - Z)$ // $2\mathbf{a}$
$(t_0, t_1) \leftarrow (t^- \cdot \pi_Z, t^+ \cdot \pi_Y)$ // $2\mathbf{M}$
$(S_X : S_Z) \leftarrow (t_0 + t_1, t_0 - t_1)$ // $2\mathbf{a}$
$(XX, XZ, ZZ) \leftarrow (X^2, XZ, X \cdot Z, Z^2)$ // $\mathbf{M} + 2\mathbf{S}$
$d \leftarrow (\ell - 1)/2$
$(q, r) \leftarrow (\lfloor (d-1)/3 \rfloor, (d-1) \mod 3)$
$(X_2 : Z_2) \leftarrow \texttt{DBL}((X_1 : Z_1), (A : C))$ // $4\mathbf{M} + 2\mathbf{S} + 8\mathbf{a}$
$(X_3 : Z_3) \leftarrow \texttt{ADD}((X_1 : Z_1), (X_2 : Z_2), (X_1 : Z_1))$ // $4\mathbf{M} + 2\mathbf{S} + 8\mathbf{a}$
for $i = 1$ to q **do**
  $m_i \leftarrow 3 * i + r$
  if $i == 1$ **then**
    if $r == 1$ **then**
      $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow \texttt{UPDATE}((X_2 : Z_2))$ // $6\mathbf{M} + 4\mathbf{a}$
      $(X_{m_1} : Z_{m_1}) \leftarrow \texttt{ADD}((X_3 : Z_3), (X_1 : Z_1), (X_2 : Z_2))$ // $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$
    else if $r == 2$ **then**
      $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow \texttt{UPDATE}((X_2 : Z_2))$ // $6\mathbf{M} + 4\mathbf{a}$
      $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow \texttt{UPDATE}((X_3 : Z_3))$ // $6\mathbf{M} + 4\mathbf{a}$
      $(X_{m_1} : Z_{m_1}) \leftarrow \texttt{ADD}((X_3 : Z_3), (X_2 : Z_2), (X_1 : Z_1))$ // $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$
    end if
  else
    $(X_{m_i} : Z_{m_i}) \leftarrow \texttt{ADD}((X_{m_{i-1}} : Z_{m_{i-1}}), (X_3 : Z_3), (X_{m_{i-2}} : Z_{m_{i-2}}))$ // $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$
  end if
  $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow \texttt{UPDATE}((X_{m_i} : Z_{m_i}))$ // $6\mathbf{M} + 4\mathbf{a}$
  $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow \texttt{2ADDSKIP}((X_{m_i} : Z_{m_i}), (X_1 : Z_1))$ // $18\mathbf{M} + 3\mathbf{S} + 14\mathbf{a}$
end for
$(X' : Z') \leftarrow (X \cdot (S_X)^2 : Z \cdot (S_Z)^2)$ // $2\mathbf{M} + 2\mathbf{S}$
$(a'_{tE}, d'_{tE}) \leftarrow \left((A + 2C)^\ell \cdot \pi_Z^8, (A - 2C)^\ell \cdot \pi_Y^8\right)$ // $(2 + \tilde{\ell})\mathbf{M} + (2\tilde{\ell} + 6)\mathbf{S} + 3\mathbf{a}$
$(A' : C') \leftarrow (2(a'_{tE} + d'_{tE}), a'_{tE} - d'_{tE})$ // $3\mathbf{a}$
return $(X' : Z')$ and $(A' : C')$

と変形できることから，与えられた X について $\Psi(X)$ を効率よく計算することが重要となる．彼らはおよそ $\sqrt{\ell}$ 次の 2 つの多項式の集結式によって $\Psi(X)$ を計算し，$\tilde{O}(\sqrt{\ell})$ での同種写像計算を実現した．

4. 提案

4.1 2-ADD-Skip method を用いたアルゴリズムの一般化

Algorithm 2 では，ある点 $(X_{m_i} : Z_{m_i})$ に対して $\texttt{2ADDSKIP}((X_{m_i} : Z_{m_i}), (X_1 : Z_1))$ が計算されていた．しかし第 2 引数は $(X_1 : Z_1)$ に限らず n 点に一般化できる．

まず点集合と点に関してブロック更新という操作を定義する．n 点の集合 $N = \{(X_1 : Z_1), \ldots, (X_n : Z_n)\}$ および点 $(X_{m_i} : Z_{m_i}) \notin N$ に関するブロック更新とは，$\texttt{2ADDSKIP}((X_{m_i} : Z_{m_i}), (X_j : Z_j)), (X_j : Z_j) \in N$ による

ⓒ 2020 Information Processing Society of Japan

Vol.2020-CSEC-91 No.33
Vol.2020-SPT-40 No.33
Vol.2020-EIP-90 No.33
2020/11/26

n 回の計算と 1 回の UPDATE$((X_{m_i} : Z_{m_i}))$ から構成される．図 1 が示すように，ブロック更新によって連続した $2n+1$ 点の更新が計算できる．

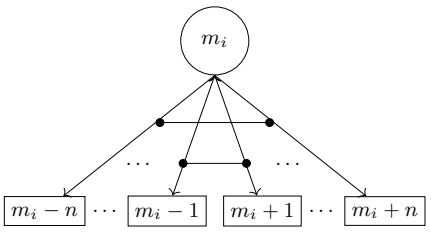


図 1 集合 $N = \{(X_1 : Z_1), \ldots, (X_n : Z_n)\}$ および 点 $(X_{m_i} : Z_{m_i}) \notin N$ に関するブロック更新

したがって Algorithm 3 のように，ブロック更新を組み合わせることで核に含まれる d 点に関する更新を抜け漏れなく計算できる．同種写像の次数 $2d+1 \geq 9$ および整数 $n \geq 1$ が与えられたとき，$q \geq 1$, $0 \leq r < 2n+1$ および $d = q(2n+1)+r+n$ を満たす整数 q, r が一意に決定する．点集合 M, N, R を次のように定義する．

$$M = \{(X_{m_i} : Z_{m_i}) \mid m_i = i(2n+1)+r, 1 \leq i \leq q\},$$
$$N = \{(X_i : Z_i) \mid 1 \leq i \leq n\},$$
$$R = \{(X_{n+i} : Z_{n+i}) \mid 1 < i \leq r\}.$$

Algorithm 3 は集合 $(N \cup R) \setminus \{(X_1, Z_1)\}$ に含まれる点に関して UPDATE を計算したのち（8-14 行目），N および $(X_{m_i} : Z_{m_i}) \in M$ に関するブロック更新（18-33 行目）を行うことで d 点の更新を計算している．すなわち合計 nq 回の 2-ADD-Skip method が実行されており，Algorithm 1 と比べておよそ $2nq$ 点の点計算が省略されている．なお 2-ADD-Skip method を 1 度以上実行するために $d > 4$ を仮定している．また次数 ℓ が素数かつ $q > 0$ であるとき，$r \neq 0$ であることを利用して記述を簡略化している．

4.2 集合 N の要素数の最適化

同種写像の次数 ℓ に対する集合 N の要素数 n を最適化するために Algorithm 3 の演算量を計算する．まず ADD や DBL を用いて点計算を行う点の集合は $U = (M \cup N \cup R) \setminus \{(X_1 : Z_1)\}$ で表すことができる．ただしここでは簡単のため 15 行目の演算が行われないものと仮定している．また U に含まれる点について UPDATE が計算され，さらに直積 $M \times N$ の点のペアについて 2ADDSKIP が計算される．すなわち Algorithm 3 の演算量 Cost(Alg.3) は以下で与えられる．

$$\text{Cost(Alg.3)} = (q+n+r-1) \cdot (\text{Cost(ADD)} + \text{Cost(UPDATE)}) + nq \cdot \text{Cost(2ADDSKIP)} + C.$$

ここで C はアルゴリズム冒頭（1 – 5 行目）および末尾（33 – 35 行目）で計算される演算量を表す．

一方，2-ADD-Skip method を用いない Algorithm 1 の

Algorithm 3 2-ADD-Skip method を用いた同種写像計算アルゴリズムの一般化

Input: $\ell \geq 9$, n, $(X : Z)$, $(X_1 : Z_1)$, and $(A : C)$
Output: $(X' : Z')$ and $(A' : C')$
1: $(\pi_Y, \pi_Z) \leftarrow (X_1 - Z_1, X_1 + Z_1)$ // **2a**
2: $(t^+, t^-) \leftarrow (X + Z, X - Z)$ // **2a**
3: $(t_0, t_1) \leftarrow (t^- \cdot \pi_Z, t^+ \cdot \pi_Y)$ // **2M**
4: $(S_X : S_Z) \leftarrow (t_0 + t_1, t_0 - t_1)$ // **2a**
5: $(XX, XZ, ZZ) \leftarrow (X^2, XZ, X \cdot Z, Z^2)$ // $\mathbf{M} + 2\mathbf{S}$
6: $d \leftarrow (\ell - 1)/2$
7: $(q, r) \leftarrow (\lfloor (d-n)/(2n+1) \rfloor, (d-n) \mod (2n+1))$
// $N \cup R$ に含まれる点の点計算および更新
8: $(X_2 : Z_2) \leftarrow$ DBL$((X_1 : Z_1), (A : C))$ // $4\mathbf{M} + 2\mathbf{S} + 8\mathbf{a}$
9: $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow$ UPDATE$((X_2 : Z_2)$ // $6\mathbf{M} + 4\mathbf{a}$
10: **for** $i = 3$ to $n + r$ **do**
11: $(X_i : Z_i) \leftarrow$ ADD$((X_{i-1} : Z_{i-1}), (X_1 : Z_1), (X_{i-2} : Z_{i-2}))$
// $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$
12: $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow$ UPDATE$((X_i : Z_i))$ // $6\mathbf{M} + 4\mathbf{a}$
13: **end for**
14: **if** $2n + 1 > n + r$ **then**
15: $(X_{2n+1} : Z_{2n+1}) \leftarrow$ ADD$((X_n : Z_n), (X_{n+1} : Z_{n+1}), (X_1 : Z_1))$ // $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$
16: **end if**
17: **for** $i = 1$ to q **do**
18: $m_i \leftarrow i(2n+1) + r$
19: **if** $i == 1$ **then**
20: **if** $r == 1$ **then**
21: $(X_{m_1} : Z_{m_1}) \leftarrow$ DBL$((X_{n+1} : Z_{n+1}), (A : C))$ // $4\mathbf{M} + 2\mathbf{S} + 8\mathbf{a}$
22: **else**
23: $(X_{m_1} : Z_{m_1}) \leftarrow$ ADD$((X_{n+r} : Z_{n+r}), (X_{n+1} : Z_{n+1}), (X_{r-1} : Z_{r-1}))$ // $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$
24: **end if**
25: **else**
26: $(X_{m_i} : Z_{m_i}) \leftarrow$ ADD$((X_{m_{i-1}} : Z_{m_{i-1}}), (X_{2n+1} : Z_{2n+1}), (X_{m_{i-2}} : Z_{m_{i-2}}))$ // $4\mathbf{M} + 2\mathbf{S} + 6\mathbf{a}$
27: **end if**
// N と $(X_{m_i} : Z_{m_i})$ に関するブロック更新
28: $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow$ UPDATE$((X_{m_i} : Z_{m_i}))$ // $6\mathbf{M} + 4\mathbf{a}$
29: **for** $j = 1$ to n **do**
30: $(S_X, S_Z, \pi_Y, \pi_Z) \leftarrow$ 2ADDSKIP$((X_{m_i} : Z_{m_i}), (X_j : Z_j))$
// $18\mathbf{M} + 3\mathbf{S} + 14\mathbf{a}$
31: **end for**
32: **end for**
33: $(X' : Z') \leftarrow (X \cdot (S_X)^2 : Z \cdot (S_Z)^2)$ // $2\mathbf{M} + 2\mathbf{S}$
34: $(a'_{tE}, d'_{tE}) \leftarrow \left((A+2C)^\ell \cdot \pi_Z^8, (A-2C)^\ell \cdot \pi_Y^8\right)$ // $(2+\tilde{\ell})\mathbf{M} + (2\tilde{\ell}+6)\mathbf{S} + 3\mathbf{a}$
35: $(A' : C') \leftarrow (2(a'_{tE} + d'_{tE}), a'_{tE} - d'_{tE})$ // $3\mathbf{a}$
36: **return** $(X' : Z')$ and $(A' : C')$

計算量は以下で与えられる．

$$\text{Cost(Alg.1)} = (d-1) \cdot (\text{Cost(ADD)} + \text{Cost(UPDATE)}) + C.$$

より正確には Algorithm 3 の 5 行目の計算（$1\mathbf{M} + 2\mathbf{S}$）が存在しないが，簡単のために等しく C を用いる．$d = q(2n+1)+r+n$ が成立することから，2-ADD-Skip method の実行によって削減することができる演算量は

$$\begin{aligned}&\mathrm{Cost}(\text{Alg.1}) - \mathrm{Cost}(\text{Alg.3})\\ &= 2nq \cdot (\mathrm{Cost}(\texttt{ADD}) + \mathrm{Cost}(\texttt{UPDATE})) - nq \cdot \mathrm{Cost}(\texttt{2ADDSKIP})\\ &= nq \cdot (2\mathbf{M} + 1\mathbf{S} + 6\mathbf{a}).\end{aligned}$$

したがって，与えられた同種写像の次数 $\ell = 2d+1$ に対して集合 N の要素数 n の最適化とは 2ADDSKIP の計算回数 nq が最大化に帰着できる．

簡単のため，床関数を用いずに $q = \frac{d-n}{2n+1}$ とおく．このとき関数

$$f(n) = nq = n \cdot \frac{d-n}{2n+1}$$

は $n > 0$ および $d > 3$ の条件下において

$$n_M = \frac{\sqrt{\ell} - 1}{2}$$

で唯一の極大値を持つ．したがって f の最大値は $(\ell + 1 - 2\sqrt{\ell})/4$ であり，このとき $q = n_M$ が成立する．以上より ℓ-同種写像に対して，集合 N および M の要素数がおよそ $\frac{\sqrt{\ell}-1}{2}$ である Algorithm 3 によっておよそ

$$\frac{(\ell + 1 - 2\sqrt{\ell})}{4} \cdot (2\mathbf{M} + 1\mathbf{S} + 6\mathbf{a})$$

の演算量を削減できる．

さらに $\mathbf{S} = \mathbf{M}$ および $\mathbf{a} = 0\mathbf{M}$ としたとき n の最適化前である Algorithm 2 の場合

$$\begin{aligned}\frac{\mathrm{Cost}(\text{Alg.1}) - \mathrm{Cost}(\text{Alg.2})}{\mathrm{Cost}(\text{Alg.1})} &\approx \frac{((\ell-2)/6) \cdot 3\mathbf{M}}{(d-1) \cdot 12\mathbf{M} + C}\\ &\approx \frac{\ell - 2}{12\ell - 1 + 12\log\ell}\end{aligned}$$

と書けることから，ℓ が十分に大きいときおよそ 8% の高速化が期待できる．これに対し n を最適化することによって，

$$\begin{aligned}\frac{\mathrm{Cost}(\text{Alg.1}) - \mathrm{Cost}(\text{Alg.3})}{\mathrm{Cost}(\text{Alg.1})} &\approx \frac{f(n_M) \cdot 3\mathbf{M}}{(d-1) \cdot 12\mathbf{M} + C}\\ &\approx \frac{3(\ell + 1 - 2\sqrt{\ell})}{24\ell - 1 + 12\log\ell}\end{aligned}$$

期待できる高速化の割合を 12% に改善できることが示された．

5. 評価

本稿で提案したアルゴリズムの性能を評価するために，様々な次数 ℓ の同種写像計算および CSIDH における計算について計算機実験を行った．提案したアルゴリズムは https://velusqrt.isogeny.org [13] において公開されたソースコードを元に C 言語で実装された．また計算に要するクロック数について，Meyer らのアルゴリズム [5] および Bernstein らのアルゴリズム [13] との比較を行った．なお本実験は Intel Core i7-8569U の Coffee Lake プロセッサにおいて Turbo Boost を無効にして行われた．

提案したアルゴリズムのパラメータ n について，同種写像の次数 ℓ ごとにクロック数が最小となる値を事前に計算し利用した．なおそれらの最適値は 4.2 章で導出した n_M に近い値であることが確認された．

5.1 ℓ-同種写像計算

図 2 は様々な次数 ℓ の同種写像計算に要する計算量を両対数グラフを用いてまとめたものである．対象とする素数 ℓ は，CSIDH-512 で用いられる最小の奇素数 73 個および 587 とした．Bernstein らの論文と同様に，横軸は次数 ℓ に対応し，縦軸は同種写像計算に要するクロック数について 15 回の実験における中央値を $\ell + 2$ で割った値に対応している．また緑，赤，青の点はそれぞれ提案したアルゴリズム，Bernstein らのアルゴリズム，Meyer らのアルゴリズムを表す．

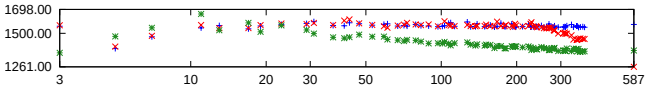


図 2 ℓ-同種写像計算に要する計算量

次数が $19 \le \ell \le 373$ を満たすとき，提案したアルゴリズムが最も効率良く同種写像を計算できることが示された．また，提案したアルゴリズムは Meyer らのアルゴリズムと比較して漸近的におよそ 12% の計算量を削減できることが確認できた．これは 4.2 章の結果と一致する．

5.2 CSIDH における公開鍵計算

図 3 は CSIDH-512 において与えられた秘密鍵に対応するイデアル類群における群作用の計算に要する計算量をまとめたものである．本実験では Bernstein らの論文と同様に，ランダムに生成された 65 種の秘密鍵についてそれぞれ 15 回の公開鍵計算を実行した．横軸は秘密鍵に対応し，計算量の中央値について昇順に整列している．また縦軸はクロック数に対応している．図における点の色が表すアルゴリズムは先述の通りである．なお全ての実装において constant-time 化は考慮されていないことに注意する．

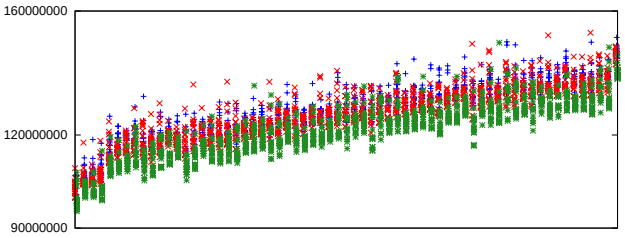


図 3 CSIDH-512 における公開鍵計算に要する計算量

CSIDH-512 において提案したアルゴリズムによる同種写像計算を用いた場合，クロック数が最小となることが示された．また提案したアルゴリズムおよび Bernsterin らの

アルゴリズムを用いた場合，Meyer らのアルゴリズムを用いた場合と比較してそれぞれ 5.2%，1.0% の割合で高速に計算できることが確認できた．

6. 結論

本稿では同種写像暗号 CSIDH の高速化を目指し，奇数次数の ℓ-同種写像の効率的な計算手法について考察した．特に 2-ADD-Skip method を用いる同種写像計算アルゴリズムの改良に取り組んだ．既存のアルゴリズムにおける 2-ADD-Skip method の活用方法を拡張し，パラメータ n を持つ新たなアルゴリズムを提案した．さらに演算量を解析することで次数 ℓ に対して最適となる n の値を導出した．その結果 Meyer らのアルゴリズムと比較しておよそ $\frac{(\ell+1-2\sqrt{\ell})}{4} \cdot (2\mathbf{M} + 1\mathbf{S} + 6\mathbf{a})$ の演算を削減できることを示した．

また提案したアルゴリズムの性能評価のために，計算機実験を通して Meyer ら，Bernstein らのアルゴリズムとの比較を行った．漸近的な計算量は Bernstein らのアルゴリズムに劣るものの，次数が $19 \leq \ell \leq 373$ を満たす同種写像においては提案したアルゴリズムが最小のクロック数で計算できることを示した．さらに同種写像計算アルゴリズムを CSIDH-512 に適用したとき，提案したアルゴリズムを用いた場合の計算量が最小であることを明らかにした．提案したアルゴリズムを用いることで，Meyer らのアルゴリズムを用いた場合に比べておよそ 5.2% のクロック数を削減できる．

謝辞 本研究は JSPS 科研費 JP1910400 の助成を受けたものです．また本研究の一部は文部科学省「Society5.0 に対応した高度技術人材育成事業成長分野を支える情報技術人材の育成拠点の形成 (enPiT)」さらに文部科学省の平成 30 年度「Society 5.0 実現化研究拠点支援事業」の助成を受けています．

参考文献

[1] Feo, L. D., Jao, D. and Plût, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies, *J. Mathematical Cryptology*, Vol. 8, No. 3, pp. 209–247 (2014).

[2] Castryck, W., Lange, T., Martindale, C., Panny, L. and Renes, J.: CSIDH: An Efficient Post-Quantum Commutative Group Action, *ASIACRYPT 2018*, LNCS, Vol. 11274, Springer, pp. 395–427 (2018).

[3] Jao, D., Azarderakhsh, R., Campagna, M., Costello, C., Feo, L. D., Hess, B., Jalali, A., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Pereira, G., Renes, J., Soukharev, V. and Urbanik, D.: Supersingular isogeny key encapsulation. Submission to the NIST Post- Quantum Cryptography Standardization project. , https://sike.org, accessed Sep. 18. 2020.

[4] Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Liu, Y.-K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A. and Smith-Tone, D.: Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process, *NISTIR 8240* (2019).

[5] Meyer, M. and Reith, S.: A Faster Way to the CSIDH, *INDOCRYPT 2018* (Chakraborty, D. and Iwata, T., eds.), LNCS, Vol. 11356, Springer, pp. 137–152 (2018).

[6] Bernstein, D. J., Hamburg, M., Krasnova, A. and Lange, T.: Elligator: elliptic-curve points indistinguishable from uniform random strings, *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, ACM, pp. 967–980 (2013).

[7] Meyer, M., Campos, F. and Reith, S.: On Lions and Elligators: An Efficient Constant-Time Implementation of CSIDH, *PQCrypto 2019* (Ding, J. and Steinwandt, R., eds.), LNCS, Vol. 11505, Springer, pp. 307–325 (2019).

[8] Hutchinson, A., LeGrow, J. T., Koziel, B. and Azarderakhsh, R.: Further Optimizations of CSIDH: A Systematic Approach to Efficient Strategies, Permutations, and Bound Vectors, *IACR Cryptol. ePrint Arch.*, Vol. 2019, p. 1121 (2019).

[9] Nakagawa, K., Onuki, H., Takayasu, A. and Takagi, T.: L_1-Norm Ball for CSIDH: Optimal Strategy for Choosing the Secret Key Space, *IACR Cryptology ePrint Archive*, Vol. 2020, p. 181 (2020).

[10] Cervantes-Vázquez, D., Chenu, M., Chi-Domínguez, J., Feo, L. D., Rodríguez-Henríquez, F. and Smith, B.: Stronger and Faster Side-Channel Protections for CSIDH, *LATINCRYPT 2019*, LNCS, Vol. 11774, Springer, pp. 173–193 (2019).

[11] Bernstein, D. J., Lange, T., Martindale, C. and Panny, L.: Quantum Circuits for the CSIDH: Optimizing Quantum Evaluation of Isogenies, *EUROCRYPT 2019*, LNCS, Vol. 11477, Springer, pp. 409–441 (2019).

[12] 小寺健太，鄭 振牟，宮地充子：Montgomery 曲線における同種写像計算の高速化について，*SCIS2020*.

[13] Bernstein, D. J., Feo, L. D., Leroux, A. and Smith, B.: Faster computation of isogenies of large prime degree, *IACR Cryptol. ePrint Arch.*, Vol. 2020, p. 341 (2020).

[14] Onuki, H., Aikawa, Y., Yamazaki, T. and Takagi, T.: (Short Paper) A Faster Constant-Time Algorithm of CSIDH Keeping Two Points, *IWSEC 2019*, LNCS, Vol. 11689, Springer, pp. 23–33 (2019).

[15] Montgomery, P. L.: Speeding the Pollard and elliptic curve methods of factorization, *Mathematics of Computation*, Vol. 48, pp. 243–264 (1987).

[16] Costello, C., Longa, P. and Naehrig, M.: Efficient Algorithms for Supersingular Isogeny Diffie-Hellman, *CRYPTO 2016*, LNCS, Vol. 9814, Springer, pp. 572–601 (2016).

[17] Vélu, J.: Isogènies entre courbes elliptiques, *C, R, Acad. Sci. Paris Sér. A-B*, Vol. 273, pp. A238–A241 (1971).

[18] Costello, C. and Hisil, H.: A Simple and Compact Algorithm for SIDH with Arbitrary Degree Isogenies, *ASIACRYPT 2017*, LNCS, Vol. 10625, Springer, pp. 303–329 (2017).

[19] Renes, J.: Computing Isogenies Between Montgomery Curves Using the Action of (0, 0), *PQCrypto 2018*, LNCS, Vol. 10786, Springer, pp. 229–247 (2018).

[20] Bernstein, D. J., Birkner, P., Joye, M., Lange, T. and Peters, C.: Twisted Edwards Curves, *AFRICACRYPT 2008*, Springer, pp. 389–405 (2008).

[21] Moody, D. and Shumow, D.: Analogues of Vélu's formulas for isogenies on alternate models of elliptic curves, *Math. Comput.*, Vol. 85, No. 300, pp. 1929–1951 (2016).

ⓒ 2020 Information Processing Society of Japan