Vol.2010-CSEC-51 No.10
2010/12/10

効率的な 3 倍算公式を用いたスカラー倍算手法の提案

笹原　大揮[†1]　宮地充子[†1]

現在，公開鍵暗号として RSA が広く使われているが，鍵長が 2048 ビット必要である．これに対して楕円曲線暗号の場合，224 ビットで同等の安全性を実現できるため，楕円曲線暗号は注目されている．楕円曲線の主な演算はスカラー倍算と呼ばれ，大きく事前計算テーブルを利用する手法と利用しない手法に分かれる．本研究では始めに効率的な 3 倍算公式と連続 3 倍算公式を提案する．その演算公式を用いて事前計算テーブル生成手法と事前計算を利用しないスカラー倍算手法の提案を行う．

Scalar Multiplication Method using efficient Tripling Formurae

Taiki Sasahara[†1] and Atsuko Miyaji[†1]

Recently, RSA method is widely used as Public Key Cryptosystem, but it's key length need 2048 bits. Elliptic curve cryptography introduced independently by Koblitz and Miller in the mid-eighties. Elliptic Curve Cryptosystem(ECC) with a key size of 224 bits provides the same level of security as RSA with a key size of2048 bits. It is expected that ECC will become popular for many information security applications in the near future. A Main operation on ECC is calculation of scalar multiplication. The scalar multiplication is to calulate kP from a integer k and a point P on EC. It can be classified into a method using precomputation point and a method unusing precomputation point. In this paper, we propose an efficient Tripling formula and a Tripling formula for consecutive 0. Using new operations, we propose new scalar multiplication method.

1. はじめに

楕円曲線暗号は次世代の公開鍵暗号として注目を集めている．RSA 等の既存の公開鍵暗号よりも短い鍵で署名を作成できる．その事はスマートカードのような計算能力，メモリに制限があるデバイスで活用が見込まれている．楕円曲線暗号の安全性は楕円離散対数問題の困難さに基づいている．楕円曲線暗号の中心となる計算はスカラー倍算と呼ばれ，点 $P, Q \in E(K)$，整数 $k \in \mathbb{Z}$ に対して $Q = kP$ の計算である．このスカラー倍算の高速化手法の研究が活発に行われている．高速化アプローチは大きく分けて以下の 4 点が挙げられる．一点目はワイエルシュトラス標準形から変数変換を行い，得意な演算が実行できるようパラメータを変換する．二点目は Affine 座標が逆元の計算を必要とし，逆元の計算に計算コストがかかるため，射影座標上で演算を行うため，射影座標の構築を行う．3 点目は整数 k の表現方法を 2 進展開から 3 進展開や 2 進と 3 進の組み合わせ等に変化させ効率よくスカラー倍算を行う．最後にスカラー倍算手法の改良である．上記高速アプローチの既存研究は数多く存在しているが，今回挙げる既存研究は次の通りとなる．Meloni は 2007 年に同一 Z 座標 (ZADD) を用いた加法公式を提案し，$P' + \cdots + P' + 2P$ というシーケンスで kP を計算している．これは $2P$ を計算する中で $2P$ の Z 座標と同じ P' を計算し，ZADD を k のビット長回繰り返す[6]．また，宮地は 2008 年に射影座標上での 2 倍点の計算が連続する時に有効となる連続 2 倍算公式を提案している．この演算は一回目の 2 倍点の計算を利用して 2 回目以降の aZ_2^4 の計算量を減らしている[7]．さらに Longa らは 2008 年に射影座標上での 3 倍点の計算を $6M + 10S$ で計算できる 3 倍算公式を提案し，連続 3 倍算は $6wM + 10wS$ の計算で実行できるとしている[4]．また，Ciet らは 2006 年に Ternary/Binary というスカラー倍算を提案している．これは整数 k を 6 で割り，2 か 3 で割り切れるなら 2 倍算，3 倍算を行い，割り切れなければ 3 倍算を行った後，2 倍加算を行うというアルゴリズムである[1]．そして，横川，宮地，笹原は 2-and-3 倍算公式 (DT) を提案している．これは 3 倍点の計算の中で $+1M$ することで 2 倍点を計算できる演算となっている[8],[9]．最近では CHES2010 にて Goundar らにより同一 Z 座標を用いた加算に加え，共役加算を提案している．また 2 倍加算をスカラー倍算の主たる演算とすることでサイドチャネル攻撃対策を盛り込んだスカラー倍算手法を提案している[3]．本研究ではさらなるスカラー倍算高速化手法の構築を目的とする．

本論文では Affine 座標上での効率的な 3 倍算公式，同一 Z 座標を用いた 3 倍算公式，同一 Z 座標となる連続 3 倍算公式，同一 Z 座標となる 2-and-3 倍算公式と入力と関係ない点を加算後の点と同一 Z 座標となる CoADD3P を提案する．この提案した演算を用いてスカラー倍算の高速化を図る．まず，2 章で本論文中で使う用語と計算量削減アイディア，楕円曲線について説明する．3 章では既存の加法公式とスカラー倍算手法を紹介し，4 章で計算

†1 〒 923-1292 石川県能美市旭台 1-1 北陸先端科学技術大学院大学 情報科学研究科 III 棟 8F
Japan Advanced Institute of Science and Technology 1-1, Asahidai, Nomi, Ishikawa, Japan

量削減を実現した新しい加法公式とその加法公式を用いたスカラー倍算手法を提案する．5 章で既存のスカラー倍算手法と比較を行い，6 章で結論と今後の課題という構成になっている．

2. 準　　備

この章では楕円曲線と加法公式について説明する．まず，この論文中で使用する記号を表 2 に示す．次に Jacobian 座標の演算について説明する．また，各演算に対する計算量の求め方は乗算 M，2 乗算 S，逆元 I の計算回数で評価する．乗算と 2 乗算では 2 乗算の計算量の方が小さく，小さな数との乗算・加法・減算は乗算・2 乗算に比べ小さいので無視できる．さらに，一般的に計算量削減の方法には Z_1Z_2 の計算を以下のように置き換える方法 rule(2.1) がある．

rule (2.1)
$$2Z_1Z_2 = (Z_1 + Z_2)^2 - Z_1^2 - Z_2^2 \tag{2.1}$$

これは $1M$ を $1S$ に計算量を削減する．そして，既存研究を 3 節で紹介する．

記号	意味
$C_1 + C_1$	C_1 上での点加算
$2C_1$	C_1 上での 2 倍算
$3C_1$	C_1 上での 3 倍算
$C_1 \to C_2$	C_1 は C_2 の点として演算
$C_1 + C_2 \to C_3$	C_1 と C_2 上の点の加算は C_3 の点として演算
$C_1 \to C_2, C_3$	C_1 上の点は C_2 と C_3 の点として演算
t(演算)	演算の計算量

C_1, C_2 と C_3 はそれぞれ Affine($\mathcal{A}$) や Jacobian($\mathcal{J}$) などの座標.

表 2.1　記号の定義

Jacobian 座標 ($\mathcal{J}$)

$\mathcal{A}$ は加法公式の度に逆元計算が必要である. 逆元計算は乗算に比べ時間がかかるため，逆元計算を利用しない射影座標が用いられる場合が多い．一般的な射影座標として $\mathcal{J}$ の楕円曲線の方程式は,

$$E_{\mathcal{J}} : Y^2 = X^3 + aXZ^4 + bZ^6$$

で表される. この式は, $\mathcal{A}$ の楕円曲線を $(x, y) = (\frac{X}{Z^2}, \frac{Y}{Z^3})$ と変換することで与えられる. また, $\mathcal{J}$ での無限遠点は $(1, 1, 0)$ となり, $P = (X, Y, Z)$ のとき, $-P = (X, -Y, Z)$ となる. ここで, 曲線上の点が $P = (X_1, Y_1, Z_1)$, $Q = (X_2, Y_2, Z_2)$, $P + Q = R = (X_3, Y_3, Z_3)$ のとき,

- 加法公式 $(\mathcal{J} + \mathcal{J})(P \neq \pm Q)$
  $X_3 = -H^3 - 2U_1H^2 + r^2,\ Y_3 = -S_1H^3 + r(U_1H^2 - X_3), Z_3 = Z_1Z_2H$
  $(U_1 = X_1Z_2^2, U_2 = X_2Z_1^2, S_1 = Y_1Z_2^3, S_2 = Y_2Z_1^3, H = U_2 - U_1, r = S_2 - S_1\ )$
- 2 倍算公式 $(2\mathcal{J})(R = 2P)$
  $X_3 = S - T,\ Y_3 = -8Y_1^4 + NT,\ Z_3 = (Y_1 + Z_1)^2 - Y_1^2 - Z_1^2$
  $(S = (X_1 + Y_1^2)^2 - X_1^2 - Y_1^4, N = 3X_1^2 + aZ_1^4, T = 6S - N^2)$
- 3 倍算公式 $(3\mathcal{J})(R = 3P)$
  $X_3 = 16Y_1^2(B - A) + 4X_1T_2^2,\ Y_3 = 8Y_1[(B - A)(A - 2B) - T_2^3], Z_3 = (Z_1 + T_2)^2 - Z_1^2 - T_2^2$
  $(A = (N + T_2)^2 - N^2 - T_2^2, B = 16Y_1^4,\ T_2 = 6[(X_1 + Y_1^2)^2 - X_1^2 - Y_1^4] - N^2, N = 3X_1^2 + aZ_1^4)$

加算は楕円曲線の係数 a に依存しない．しかし，2 倍算や 3 倍算には依存する．加法公式 ($\mathcal{J}$) の計算量は rule(2.1) を使用し，表 4.5 にまとめる．

3. 既 存 研 究

この章では既存の加法公式とスカラー倍算手法を説明する．

3.1 加 法 公 式

特徴的な加法公式の既存研究を以下にまとめ，計算量は表 4.5 でまとめる．

(1) 連続 2 倍算 (wDBL)[7]
- 点 $P \in E(\mathbb{F}_q)$ と $w \geq 1$ から 2^wP を計算
- $2^{(1)}P$ を計算中に出てくる中間変数を $2^{(2)}P$ の計算で利用可能．w 回繰り返し可能.
- 計算アルゴリズムは表 3.1 となる．

(2) 同一 Z 座標を用いた加算 (ZADD)[3),6)]
- 点 $P \in E(\mathbb{F}_q)$ と同一 Z 座標となる $Q \in E(\mathbb{F}_q)$ の加算 $(P + Q)$．計算量は $5M + 2S$.
- 計算アルゴリズムは表 3.1 となる．

(3) 3 倍算 (TPL$^J(P_1)$)[4]
- 点 $P \in E(\mathbb{F}_q)$ と $w \geq 1$ から 3^wP を計算, 計算量は $6wM + 10wS$.
- 3 倍算公式 ($\mathcal{J}$) は表 3.1 から求める事ができる．
- 単純に連続しており，3 倍算の計算量に繰り返し回数を掛けた計算コストがかかる．

(4) 2-and-3 倍算 (DT)[8),9)]
- 点 $P \in E(\mathbb{F}_q)$ から $2P$, $3P$ を計算
- $2P$ を計算中に出てくる中間変数を利用して $3P$ を計算可能.
- DT($\mathcal{J} \to (2\mathcal{J}, 3\mathcal{J})$) の計算量は $6M + 11S$ で，計算アルゴリズムは表 3.1 となる．

Vol.2010-CSEC-51 No.10
2010/12/10

Iterated doublings 2^wP in $\mathcal{J}$
Input $P = (X_1, Y_1, Z_1)$,
Output $2^wP = (X_w, Y_w, Z_w)$
$Y_0' = 2Y_0,\ W_0 = aZ_0^4,\ T_0 = {Y_0'}^4$
$S = ((X_0 + {Y_0'}^2)^2 - X_0^2 - T_0), M = 3X_0^2 + W_0$
$X_1 = M^2 - 2S, Y_1' = 2M(S - X_1) - T_0$
$Z_1 = ((Y_0' + Z_0)^2 - {Y_0'}^2 - {Z_0}^2)/2$
For $i = 1$ to $w - 1$: {
$W_i = W_{i-1}T_{i-1},\ T_i = {Y_i'}^4$
$S = ((X_i + {Y_i'}^2)^2 - X_i^2 - T_i), M = 3X_i^2 + W_i$
$X_{i+1} = M^2 - 2S, Y_{i+1}' = 2M(S - X_{i+1}) - T_i$
$Z_{i+1} = Y_i'Z_i$}
$Y_w = Y_w'/2$.

ZADD$((P, P\prime) \to P + Q)$ in $\mathcal{J}$
Input $P = (X_1, Y_1, Z_1), P\prime = (X_1', Y_1', Z_1')$
Output $P + P\prime = (X_3, Y_3, Z_3)$
$Z_3 = Z \times (X_1' - X_1), c = (X_1' - X_1)^2$,
$Pw = X_1' \times c, Yw = X_1 \times c$,
$d = (Y_1' - Y_1)^2, Ya = Y_1 \times (Pw - Yw)$,
$X_3 = d - Pw - Yw, Y_3 = (Y_1' - Yw) \times (Yw - X_3) - Ya$

表 3.1 wDBL, ZADD

TPL$^l(P_1)$
Input $P = (X_1, Y_1, Z_1)$,
Output $3P = (X_3, Y_3, Z_3)$
$X_3 = 16Y_1^2(B - A) + 4X_1T^2$,
$Y_3 = 8Y_1[(B - A)(A - 2B) - T^3]$,
$Z_3 = (Z_1 + T)^2 - Z_1^2 - T^2$,
$A = (N + T)^2 - N^2 - T^2, B = 16Y_1^4$,
$N = 3X_1^2 + aZ_1^4, T = 6S - N^2$,
$S = [(X_1 + Y_1^2)^2 - X_1^2 - Y_1^4]$

DT$(\mathcal{J} \to (2\mathcal{J}, 3\mathcal{J}))$
Input $P = (X_1, Y_1, Z_1)$,
Output $2P = (X_2, Y_2, Z_2),\ 3P = (X_3, Y_3, Z_3)$
$X_2 = 2S - T, Y_2 = (A - B)/2$,
$Z_2 = (Y_1 + Z_1)^2 - Y_1^2 - Z_1^2$,
$X_3 = 16Y_1^2 * (B - A) + 4X_1 * T^2, Z_3 = (Z_1 + T)^2 - Z_1^2 - T^2$,
$Y_3 = 8Y_1 * [(B - A) * (A - 2B) - T^2 * T]$,
$A = (N + T)^2 - N^2 - T^2, B = 16Y_1^4, N = 3X_1^2 + a * Z_1^4$,
$S = (X_1 + Y_1^2)^2 - X_1^2 - Y_1^4, T = 6S - N^2$,

表 3.2 TPL, DT

3.2 スカラー倍算手法

スカラー倍算手法には二つの方法がある．一つ目は最上位から最下位までビットをなめながら計算してく Left-to-Right 手法 (LR) で，もう一つは最下位から最上位までビットをなめる Right-to-Left 手法 (RL) である．一般的な Binary 法と Non-Adjacent Form(NAF), Ternary 法，Ciet らが提案した Ternary/Binary[1] について以下に特長をまとめる．

(1) Binary method
- スカラー k を 2 進展開し，常に 2 倍算を行い，bit が 1 であれば加算を行う．
- Right-to-Left(RL) のアルゴリズムは Algorithm 1 のようになる．

(2) Non Adjacent Form (NAF)
- スカラー k を $\{0, 1\}$ でなく $\{-1, 0, 1\}$ を使って表現する．
- 変換アルゴリズムは Algorithm 3 に示す．
- 連続する 1 の数を減らす事ができ，Binary method より高速である．

(3) Ternary 法
- スカラー k を 3 進展開し，常に 3 倍算を行い，bit が 1 であれば加算，bit が 2 であれば 2 倍点の加算を行う．
- Left to Right(LR) の Ternary アルゴリズムは Algorithm 2 のようになる．

(4) Ternary/ Binary[1]
- スカラー k を 6 で割り，余りによって 3 倍算，2 倍算，2 倍加算を行う．
- Right to Left(RL) の Ternary/Binary アルゴリズムは Algorithm 4 に示す．

Algorithm 1 Binary 法 (RL)
Input: $P \in E(\mathbb{F}_q)$, 正整数 $k = (k_{\ell-1}, \cdots, k_1, k_0)_2$
Output: kP
1. $Y = \mathcal{O},\ R = P$
2. For $i = 0$ to $\ell - 1$
3. $\quad R = 2R$
4. $\quad$ If $k_i = 1$ then $Y = Y + R$
5. Return Y

Algorithm 2 Ternary 法 (LR)
Input: $P \in E(\mathbb{F}_q)$, 正整数 $k = (k_{\ell-1}, \cdots, k_1, k_0)_3$
Output: kP
1. $Y = \mathcal{O},\ R_1 = P, R_2 = 2P$
2. For $i = \ell - 1$ to 0
3. $\quad$ If $k_i = 1$ then $Y = R_1 + Y$
4. $\quad$ Elseif $k_i = 2$ then $Y = R_2 + Y$
5. $\quad R = 3R$
6. Return Y

Algorithm 3 NAF 変換アルゴリズム
Input: : Positive integer k
Output: : $k[]$
1. $R_1 = 0$
2. While $(IN > 0)$
3. If $(IN$ is odd$)$
4. $\quad k[R_1] = 2 - (IN \bmod 4)$
5. $\quad IN = IN - k[R_1]$
6. Else
7. $\quad k[R_1] = 0$
8. $IN = IN/2; R_1 = R_1 + 1$
9. Return s

Algorithm 4 Ternary/Binary (RL)
Input: $P \in E(\mathbb{F}_q)$, 正整数 k.
Output: kP
1. if $k = 1$ then return P
2. switch $(k \bmod 6)$
3. cases 0 mod 6, 3 mod 6: return $3((\frac{k}{3})P)$
4. cases 2 mod 6, 4 mod 6: return $2((\frac{k}{2})P)$
5. case 1 mod 6, $k = 6m + 1$: return $2((3m)P) + P$
6. case 5 mod 6, $k = 6m - 1$: return $2((3m)P) - P$

Method	ADD	DBL	DA	TPL
Binary	$\frac{\ell-1}{2}$	$\ell-1$	-	-
NAF	$\frac{\ell}{3}$	ℓ	-	-
Ternary(RL)	$\frac{0.63\,\ell}{3}$	-	$\frac{0.63\,\ell}{3}$	$0.63\,\ell$
Ternary(LR)	$\frac{2\times0.63\,(\ell-1)}{3}$	1	-	$0.63\,(\ell-1)$
ternary / binary	-	$0.33\times\frac{\ell}{\log_2 6}$	$0.33\times\frac{\ell}{\log_2 6}$	$0.67\times\frac{\ell}{\log_2 6}$

表 3.3 スカラー倍算手法ごとの理論値

4. 提案手法

この章では rule(2.1) に基づいて新しい 3 倍算公式 (New-TPL) を提案する．また，同一 Z 座標の考えを利用し同一 Z 座標の 3 倍算公式 (Co-TPLUY) など加法公式の改良を 4.1 節にて提案する．また，Co-TPL と Co-wTPL を使用して Co-TPLADD というスカラー倍算手法を提案する．提案手法は 4.2 節で説明する．

4.1 加法公式

New-TPL は $P_{\mathcal{A}}$ から 3 倍点 $3P_{\mathcal{A}}$ を求める計算で，既存の TPL($\mathcal{A}$) の $I+7M+4S$ から $I+6M+5S$ と $1M$ を $1S$ に計算量を削減している．詳細な計算アルゴリズムは表 4.1 に示す．Co-TPLUY は 3 倍点 $3P$ と同一 Z 座標 Y' を出力する．詳細な計算アルゴリズムは表 4.2 に示す．この節で紹介した演算公式の計算量は表 4.5 にて説明する．

New-TPL($\mathcal{A}\to 3\mathcal{A}$)

Input $P=(x_1,y_1)$, $r_1=x_1$, $r_2=y_1$
Output $3P=(x_3,y_3)$

1.	$r_3=r_2^2$	(y_1^2)	10.	$r_3=6\times r_3$	$(12x_1y_1^2)$	19.	$r_4=r_4\times r_6$	$(\frac{(2y_1)^3}{I})$
2.	$r_4=r_3^2$	(y_1^4)	11.	$r_6=r_5^2$	(N^2)	20.	$r_4=r_4-r_3$	(ω)
3.	$r_5=r_1^2$	(x_1^2)	12.	$r_3=r_3-r_6$	(T)	21.	$r_6=r_4-r_3$	$(\omega-\nu)$
4.	$r_3=r_1+r_3$	$(x_1+y_1^2)$	13.	$r_6=2\times r_2$	$(2y_1)$	22.	$r_5=r_4+r_3$	$(\omega+\nu)$
5.	$r_3=r_3^2$	$((x_1+y_1^2)^2)$	14.	$r_6=r_2\times r_3$	$(2y_1T)$	23.	$r_5=r_5\times r_6$	
6.	$r_3=r_3-r_4$	$(x_1^2+2x_1y_1^2)$	15.	$r_6=(r_6)^{-1}$	(I)	24.	$r_5=r_5-r_1$	(x_3)
7.	$r_3=r_3-r_5$	$(2x_1y_1^2)$	16.	$r_3=r_3\times r_6$	$(\frac{1}{2y_1})$	25.	$r_1=r_1-r_5$	
8.	$r_5=3\times r_5$	$(3x_1^2)$	17.	$r_3=r_3\times r_5$	(ν)	26.	$r_6=r_4\times r_1$	
9.	$r_5=r_5+a$	(N)	18.	$r_4=16\times r_4$	$(16y_1^4)$	27.	$r_6=r_6-r_2$	(y_3)

output $(r_5,r_6)=(x_3,\ y_3)$
cost = $1I+6M+5S+3Add$ ($8Sub$), レジスタ数 = 6

表 4.1 New-TPL

後のスカラー倍算で使用する同一 Z 座標 3 倍算公式 (Co-TPLUPY)，連続 3 倍算公式 (Co-wTPLUY) も提案する．Co-TPLUPY は入力点 P と Y から 3 倍点 $3P$ と同一 Z 座標の Y', P' を出力する．Co-wTPLUY は入力点 $Y,\ P$ から点 3^wP と Z 座標が同一となる Y' を出力する．詳細な計算アルゴリズムは表 4.3 に示す．

Co-TPLUY($(Y_{\mathcal{J}},P_{\mathcal{J}})\to(Y'_{\mathcal{J}},3P_{\mathcal{J}})$)

Input $P=(X_1,Y_1,Z_1), Y=(X_4,Y_4,Z_1)$, $r_1=X_1$, $r_2=Y_1$, $r_3=Z_1$, $r_4=X_4, r_5=Y_4$
Output $Y'=(X_5,Y_5,Z_3), 3P=(X_3,Y_3,Z_3)$

1.	$r_6=r_2^2$	(Y_1^2)	15.	$r_9=r_9-r_{11}$	(T)	29.	$r_6=r_6\times r_{10}$	$(16Y_1^2(B-A))$
2.	$r_7=r_4^2$	(Y_1^4)	16.	$r_8=r_8+r_9$	$(N+T)$	30.	$r_8=4\times r_1$	$4X_1$
3.	$r_8=r_1^2$	(X_1^2)	17.	$r_8=r_8^2$	$((N+T)^2)$	31.	$r_8=r_8\times r_{11}$	$4X_1T^2$
4.	$r_9=r_1+r_6$	$(X_1+Y_1^2)$	18.	$r_8=r_8-r_{11}$	(T^2+2NT)	32.	$r_6=r_6+r_8$	(X_3)
5.	$r_9=r_9^2$	$((X_1+Y_1^2)^2)$	19.	$r_{11}=r_9^2$	(T^2)	33.	$r_{10}=r_{10}\times r_7$	$((B-A)(A-2B))$
6.	$r_9=r_9-r_7$	$(X_1^2+2X_1Y_1^2)$	20.	$r_8=r_8-r_{11}$	$(2NT=2A)$	34.	$r_7=r_9\times r_{11}$	(T^3)
7.	$r_9=r_9-r_8$	$(2X_1Y_1^2)$	21.	$r_{12}=r_3+r_9$	(Z_1+T)	35.	$r_{10}=r_{10}-r_7$	
8.	$r_{10}=r_3^2$	(Z_1^2)	22.	$r_{12}=r_{12}-r_{11}$	$(Z_1^2+2Y_1Z_1)$	36.	$r_{10}=r_{10}\times r_2$	
9.	$r_{11}=r_{11}^2$	(Z_1^4)	23.	$r_{12}=r_{12}-r_{10}$	$(2TZ_1=Z_3)$	37.	$r_{10}=8\times r_{10}$	(Y_3)
10.	$r_{11}=a\times r_{11}$	(aZ_1^4)	24.	$r_7=16\times r_7$	$(B=16Y_1^4)$	38.	$r_4=4\times r_4$	$(4X_4)$
11.	$r_8=3\times r_8$	$(3X_1^2)$	25.	$r_6=16\times r_6$	$(16Y_1^2)$	39.	$r_4=r_4\times r_{11}$	(X_5)
12.	$r_8=r_8+r_{11}$	(N)	26.	$r_{10}=r_7-r8$	$(B-A)$	40.	$r_5=8\times r_5$	$(8Y_4)$
13.	$r_{11}=r_8^2$	(N^2)	27.	$r_7=2\times r_7$	$(2B)$	41.	$r_5=r_5\times r_7$	(Y_5)
14.	$r_9=6\times r_9$	$(12X_1Y_1^2)$	28.	$r_7=r_8-r_7$	$(A-2B)$	-	-	-

output $(r_4,r_5,r_6,r_{10},r_{12})=(X_5,\ Y_5,\ X_3,\ Y_3,\ Z_3)$
cost = $7M+10S+1D+5Add$ ($9Sub$), レジスタ数 = 12

表 4.2 Co-TPLUPY

さらに，同一 Z 座標 2-and-3 倍算公式 (Co-DTU2P) と同一 Z 座標加算公式 (CoADD3P)，点更新演算 (UpdateP) も提案する．Co-DTU2P は入力点 Y,P から点 $Y',3P,2P'$ を出力する．3 つの出力は同一 Z 座標となる．CoADD3P は入力点 Y,P,P' から $Y',3P'$ を出力する．3 つの入力点は同一 Z 座標点で，二つの出力も同一 Z 座標となる．UpdateP は 2 点の Z 座標を同一にする演算である．詳細な計算アルゴリズムは表 4.1 に示す．この論文中で使用する演算の計算量を既存研究と提案手法を含め，表 4.5 にまとめる．表 4.5 では各演算に必要なレジスタ数も記載している．

4.2 スカラー倍算手法

提案するスカラー倍算手法のアルゴリズムは Algorithm 5, 6 となる．Algorithm 5 は整数 k を 3 進展開し，最下位ビットから最上位ビットまでなめながら kP を計算する．ある i 番目のビットが 0 であれば，$w=w+1$ とし，0 でなければ w 回の Co-wTPLUY を行う．もし $k_i=1$ なら CoADDU3P を使って 3 倍点 ($3P$) と同一 Z 座標となる Y' と P' を出力し，CoADDU3P で同一 Z 座標点同士を加算する．さらに，CoADDU3P は 3 倍点 ($3P$) を加算

Co-TPLUPY$((Y_{\mathcal{J}}, P_{\mathcal{J}}) \to (Y'_{\mathcal{J}}, 3P_{\mathcal{J}}, \tilde{P_{\mathcal{J}}}))$

Input $P = (X_1, Y_1, Z_1), Y = (X_4, Y_4, Z_1)$
Output $Y' = (X_5, Y_5, Z_3), 3P = (X_3, Y_3, Z_3)$, $\tilde{P} = (\tilde{X_1}, \tilde{Y_1}, Z_3)$
$B = 16Y_1^4, N = 3X_1^2 + aZ_1^4$
$S = [(X_1 + Y_1^2)^2 - X_1^2 - Y_1^4]$
$A = (N+T)^2 - N^2 - T^2, T = 6S - N^2$
$X_3 = 16Y_1^2(B-A) + 4X_1T^2$,
$Y_3 = 8Y_1[(B-A)(A-2B) - T^3]$,
$Z_3 = (Z_1+T)^2 - Z_1^2 - T^2$,
$\tilde{X_1} = 4T^2 * X_1$,
$\tilde{Y_1} = 8T^3 * Y_1$,
$X_5 = 4T^2 * X_4$,
$Y_5 = 8T^3 * Y_4$,
cost = $8M + 10S + 1D + 5Add$ ($9Sub$),
レジスタ数 = 12

Co-wTPLUY$((Y_{\mathcal{J}}, P_{\mathcal{J}}) \to (Y'_{\mathcal{J}}, 3^wP_{\mathcal{J}}))$

Input $P = (X_1, Y_1, Z_1)$, $Y = (X_4, Y_4, Z_1)$, w : 連続回数
Output $Y' = (X_5, Y_5, Z_3), 3^wP = (X_3, Y_3, Z_3)$
$B = Y_1^4, W_1 = aZ_1^4, N_1 = 3X_1^2 + W_1$
$S = [(X_1 + Y_1^2)^2 - X_1^2 - B_1], B = 16Y_1^4$
$T_1 = 6S - N_1^2, A = (N_1+T_1)^2 - N_1^2 - T_1^2$
$X_3 = 16Y_1^2(B-A) + 4X_1T_1^2$,
$Y_3 = 8Y_1[(B-A)(A-2B) - T_1^3]$,
$Z_3 = (Z_1+T_1)^2 - Z_1^2 - T_1^2$,
For $i = 2$ to w {
$W_{3(i-1)} = 16W_{3(i-2)} * T^4_{3(i-2)}, B = Y^4_{3(i-1)}$,
$S = [(X_{3(i-1)} + Y^2_{3(i-1)})^2 - X^2_{3(i-1)} - B$,
$B = 16B, N_{3(i-1)} = 3X^2_{3(i-1)} + W_{3(i-1)}$,
$T_{3(i-1)} = 6S - N^2_{3(i-1)}$,
$A = (N_{3(i-1)} + T_{3(i-1)})^2 - N^2_{3(i-1)} - T^2_{3(i-1)}$,
$X_{3^i} = 4Y^2_{3(i-1)}(B-A) + X_{3(i-1)}T^2_{3(i-1)}$,
$Y_{3^i} = Y_{3(i-1)}[(B-A)(A-2B) - T^3_{3(i-1)}]$,
$Z_{3^i} = Z_{3(i-1)}T_{3(i-1)}$
}
$X_5 = 4T^2_{3(i-1)} * X_4$,
$Y_5 = 8T^3_{3(i-1)} * Y_4$,
cost = $(6w+1)M + (8w+2)S + wD + 5wAdd$ ($9wSub$),
レジスタ数 = 12

表 4.3 Co-TPLUY, Co-wTPLUY

Co-DTU2P$[(Y, P) \to (Y',\ 3P,\ 2P')]$

Input $P = (X_1, Y_1, Z_1), Y = (X_4, Y_4, Z_1)$
Output $Y' = (X_5, Y_5, Z_2), 3P = (X_3, Y_3, Z_2), 2P' = (X_2, Y_2, Z_3)$
$B = 16Y_1^4, N = 3X_1^2 + aZ_1^4$,
$S = [(X_1 + Y_1^2)^2 - X_1^2 - Y_1^4]$
$A = (N+T)^2 - N^2 - T^2, T = 6S - N^2$
$X_3 = 16Y_1^2(B-A) + 4X_1T^2$,
$Y_3 = 8Y_1[(B-A)(A-2B) - T^3]$,
$Z_3 = (Z_1+T)^2 - Z_1^2 - T^2$,
$(X_2 = 2S - T, Y_2 = (A-B)/2, Z_2 = (Y_1+Z_1)^2 - Y_1^2 - Z_1^2)$
$X_2 = 4 * T^2 * X_2$
$Y_2 = 8 * T^3 * Y_2$
$Z_2 = 2 * T * Z_2$
$X_3 = 4 * Y_1^2 * X_3$
$Y_3 = 8 * Y_1^3 * Y_3$
$X_5 = 16 * Y_1^2 * T^2 * X_4$,
$Y_5 = 64 * Y_1^3 * T^3 * Y_4$,
cost = $15M + 11S + 1D + 21Add$ ($15Sub$),
レジスタ数 = 11

CoADD3P$((Y, P, \tilde{P}) \to (Y + \tilde{P}, P'))$

Input $P = (X_1, Y_1, Z_1), Y = (X_4, Y_4, Z_1), \tilde{P} = (\tilde{X}, \tilde{Y}, Z_1)$
Output $Y + \tilde{P} = (X_5, Y_5, Z_3), P' = (X_3, Y_3, Z_3)$
$Z_3 = Z_1(\tilde{X} - X_4), c = (\tilde{X} - X_4)^2$
$Pw = \tilde{X} * c, Yw = X_4 * c, X_3 = X_1 * c$,
$d = (\tilde{Y} - Y_4)^2, Ya = Y_4 * (Pw - Yw)$,
$Y_3 = Y_1 * (Pw - Yw), X_5 = d - Pw - Yw$,
$Y_5 = (\tilde{Y} - Yw)(Yw - X_5) - Ya$.
cost = $7M + 2S$ ($8Sub$), レジスタ数 = 8

UpdateP$((Y, P) \to (Y', P'))$

Input $P = (X_1, Y_1, Z_1), Y = (X_4, Y_4, Z_4)$
Output $Y' = (X_5, Y_5, Z_3), P' = (X_3, Y_3, Z_3)$
$Z_5 = Z_4 * Z_1$,
$X_5 = X_4 * Z_1^2, X_3 = X_1 * Z_4^2$,
$Y_5 = Y_4 * Z_1^3, Y_3 = X_1 * Z_4^3$,
cost = $7M + 2S$, レジスタ数 = 7

表 4.4 Co-DTU2P, CoADD3P, UpdateP

した点 $(Y + P')$ と同一 Z 座標となるよう更新する．もし $k_i = 2$ なら，CoDTU2P を使って 3 倍点と Y', $2P'$ を計算する．この時出力する 3 つの点は同一 Z 座標を保持し，CoADDU3P の演算を行う．また，Algorithm 6 は入力の整数 k を $\{0, 1, 2\}$ から $\{-1, 0, 1\}$ に変換した表現であり，最上位ビットは必ず 1 となるので $\ell' - 2$digit から最下位ビットまでなめながら kP を計算していく．もし i 番目の $k_i = 1, -1$ ならそれまでの Y の値と加減算される P を同一 Z 座標にするため UpdateP 演算を使用する．同一 Z 座標になれば ZADD 演算を行う．

提案手法の理論値を表 4.2 に示す．160 ビットの整数 k を 3 進展開したときのビット長の長さは $\ell' = \log_3 2^{160}$ となる．Co-TPLADD(LR) では整数変換しており，高々1 ビットの増長が見られるため，ビット長の長さは $\ell'' = \ell' + 1$ となる．ゼロが連続する時の期待値は式 (4.1) で求める事ができる．ゼロが連続する事で 1 回に対して $1M + 2S$ 分の計算量削減が見られるため，$S = 0.8M$ とした $2.6M$ をかけている．

$$\text{期待値} = \frac{(2.6 \times \sum_{w=2}^{33}(w-1) \text{ 場合の数})}{\frac{\prod_{i=1}^{101} i}{\prod_{i=1}^{34} i \times \prod_{i=1}^{34} i \times \prod_{i=1}^{33} i}} \tag{4.1}$$

5. 性 能 比 較

既存研究と提案手法の計算量を比較する．$k = 160$bits の場合，具体的に $I = 7M, 20M$ に換算した計算量を表に示す．理論値は表 5.2 となる．表 5.2 からどの手法も $\mathcal{A}$ より $\mathcal{J}$ の方が計算コストが低いので，Ternary($\mathcal{J}$)，Ternary/Binary$\mathcal{J}$)，EX-CoDT($\mathcal{J}$) と Co-TPLADD(LR)($\mathcal{J}$) について全体にかかるレジスタ数を記載する．また，Ternary($\mathcal{J}$)，Ternary/Binary$\mathcal{J}$) と EX-CoDT($\mathcal{J}$) においては 10^4 回の実験による平均値を表 5.1 に示す．

6. 結　　論

我々は効率的な 3 倍算公式を提案した．その計算量は $1I + 6M + 5S$ である．さらに同一 Z 座標を用いた 3 倍算公式 (Co-TPL) と整数 k の i 番目が 0 となり，連続して 0 が得られる場合に使用する連続 3 倍算公式 (Co-wTPLUY) 等の演算を提案した．提案した演算を用いてスカラー倍算を Right-to-Left と Left-to-Right で行う EX-CoDT と Co-TPLADD を提案

演算	計算量	レジスタ数
$ADD((\mathcal{A},\mathcal{A}) \to \mathcal{A}+\mathcal{A})$	$I+2M+1S$	-
$\mathrm{DBL}(\mathcal{A} \to 2\mathcal{A})$	$I+2M+2S$	-
$\mathrm{TPL}(\mathcal{A} \to 3\mathcal{A})$[1)]	$I+7M+4S$	6
$\mathrm{New\text{-}TPL}(\mathcal{A} \to 3\mathcal{A})$	$I+6M+5S$	6
$\mathrm{DA}((\mathcal{A}_1,\mathcal{A}_2) \to 2\mathcal{A}_1+\mathcal{A}_2(\mathcal{A}))$[1)]	$I+9M+2S$	-
$\mathrm{TA}((\mathcal{A}_1,\mathcal{A}_2) \to 3\mathcal{A}_1+\mathcal{A}_2(\mathcal{A}))$[1)]	$2I+9M+4S$	-
$\mathrm{ADD}((\mathcal{J},\mathcal{J}) \to \mathcal{J}+\mathcal{J}(\mathcal{J}))$[2)]	$11M+5S$	8
$\mathrm{mADD}((\mathcal{J},\mathcal{A}) \to \mathcal{J}+\mathcal{A}(\mathcal{J}))$[7)]	$7M+4S$	6
$\mathrm{UpdateP}((Y,P) \to (Y',P'))$	$7M+2S$	7
$\mathrm{ZADD}((\mathcal{J},\mathcal{J}) \to \mathcal{J}+\mathcal{J})$[3),6)]	$5M+2S$	6
$\mathrm{CoADD3P}((Y,P,\tilde{P}) \to (Y+\tilde{P},P'))$	$7M+2S$	8
$\mathrm{DBL}(\mathcal{J} \to 2\mathcal{J})$[4)]	$1M+8S+1D$	6
$\mathrm{wDBL}(\mathcal{J} \to 2^w\mathcal{J})$[7)]	$(2w-1)M+(5w+3)S+wD$	-
$\mathrm{DA}(\mathcal{J})$[5)]	$14M+9S$	8
$\mathrm{TPL}(\mathcal{J} \to 3\mathcal{J})$[4)]	$5M+10S+1D$	8
$\mathrm{Co\text{-}TPLUPY}((Y_{\mathcal{J}},P_{\mathcal{J}},\tilde{P_{\mathcal{J}}}) \to (Y'_{\mathcal{J}},3P_{\mathcal{J}},\tilde{P_{\mathcal{J}}}))$	$8M+10S+1D$	12
$\mathrm{Co\text{-}TPLUY}((Y_{\mathcal{J}},P_{\mathcal{J}}) \to (Y'_{\mathcal{J}},3P_{\mathcal{J}}))$	$7M+10S+1D$	12
$\mathrm{wTPL}(\mathcal{J} \to 3^w\mathcal{J})$[4)]	$5wM+10wS+wD$	-
$\mathrm{Co\text{-}wTPLUY}((Y_{\mathcal{J}},P_{\mathcal{J}},\tilde{P_{\mathcal{J}}}) \to (Y'_{\mathcal{J}},3^wP_{\mathcal{J}}))$	$(6w+1)M+(8w+2)S+wD$	-
$\mathrm{DT}(\mathcal{J} \to (2\mathcal{J},3\mathcal{J}))$[8),9)]	$5M+11S+1D$	11
$\mathrm{DT}(\mathcal{A} \to (2\mathcal{J},3\mathcal{J}))$[8),9)]	$5M+7S$	8
$\mathrm{Co\text{-}DTU2P}((Y,P) \to (Y',\ 3P,\ 2P'))$	$15M+11S+1D$	11

ここで D とは曲線パラメータである a の掛け算を表している．

表 4.5 加法公式の計算量

Algorithm 5 EX-CoDT (Right-to-Left)

Input: $P \in E(\mathbb{F}_q), k=(k_{\ell-1},\cdots,k_0)_3,\ w$
Output: kP

$Y=\mathcal{O}, R_1=P, \tilde{R_1}=P, R_2=P, w=0;$
For $i=0$ to $\ell'-1$
  If $k_i=0$ then $w=w+1;$
  ElseIf
    $\{Y,R_1\}$ =Co-wTPLUY$[w,Y,R_1]$
    If $k_i=1$ then $\{Y,R_1,\tilde{R_1}\}$ =CoTPLUP$[Y,R_1]$
      $\{Y,R_1\}$ =CoADDU3P$[Y,R_1,\tilde{R_1}]$
    If $k_i=2$ then $\{Y,R_1,R_2\}$ =CoDTU2P$[Y,R_1]$
      $\{Y,R_1\}$ =CoADDU3P$[Y,R_1,R_2]$
  $w=0;$
Return Y

Algorithm 6 Co-TPLADD (Left-to-Right)

Input: $P \in E(\mathbb{F}_q), k=(k_{\ell'-1},\cdots,k_0), k_i=\{-1,0,1\}$
Output: kP

$Y=P, R_1=P, w=0;$
For $i=\ell''-2$ to 0
  $Y=$ TPL$[Y]$
  If $k_i=1$ then $\{Y,R_1\}=$ UpdateP$[Y,R_1]$
    $Y=$ ZADD$[Y,R_1]$
  If $k_i=-1$ then $\{Y,R_1\}=$ UpdateP$[Y,R_1]$
    $Y=$ ZADD$[Y,-R_1]$
Return Y

Method	CoADD3P	CoTPLUY	Co-TPLUP	Co-wTPLUY	CoDTU2P
EX-CoDT(RL)	$\frac{2\times\ell'}{3}$	ℓ'	$\frac{\ell'}{3}$	$-\ell'\times$期待値	$\frac{\ell'}{3}$
Method	ZADD	TPL	UpdateP	-	-
Co-TPLADD(LR)	$\frac{2\times\ell''}{3}$	ℓ''	$\frac{2\times\ell''}{3}$	-	-

表 4.6 提案手法の理論値

手法 ($\mathcal{J}$)	全体の計算量 160bits ($S=0.8M$)	全体にかかる乗算数 160bits	
		$I=7M$	$I=20M$
Ternary(LR)($\mathcal{J}$)	$2I+1079M+1277S(2101M)$	2115	2141
Ternary/Binary(RL)($\mathcal{J}$)	$I+906M+1160S(1834M)$	1841	1854
EX-CoDT(RL)($\mathcal{J}$)	$I+1414M+976S(2195M)$	2202	2215

表 5.1 実験上の計算量の比較 (試行 10^4 回平均)

した．提案手法は同一 Z 座標を用いているため既存研究よりスカラー倍算時に使用するレジスタの数を高々12 個持っておくことで計算できる．今後，連続 3 倍算の期待値の算出方法を再度検討していく．

参 考 文 献

1) Mathieu Ciet, Marc Joye, Kristin Lauter, and PeterL. Montgomery. Trading inversions for multiplications in elliptic curve cryptography. *Des. Codes Cryptography*, 39(2):189–206, 2006.
2) D.J.Bernstein and T.Lange. Fast addition and doubling on elliptic curves. In *ASIACRYPT*, pages 29–50, 2007.
3) RaveenR. Goundar, Marc Joye, and Atsuko Miyaji. Co- addition formulæ and binary ladders on elliptic curves - (extended abstract). In *CHES*, pages 65–79, 2010.
4) Patrick Longa and Ali Miri. Fast and flexible elliptic curve point arithmetic over prime fields. *IEEE Trans. Computers*, 57(3):289–302, 2008.
5) Patrick Longa and Ali Miri. New multibase non-adjacent form scalar multiplication and its application to elliptic curve cryptosystems(extended version), 2008.
6) Nicolas Meloni. New point addition formulae for ecc applications. In *WAIFI*, pages 189–201, 2007.
7) Atsuko Miyaji. Generalized scalar multiplication secure against spa, dpa, and rpa. *IEICE Transactions*, 91-A(10):2833–2842, 2008.
8) 横川広幸, 宮地充子, and 笹原大揮. 新しい加法公式に基づく効率的なスカラー倍算について, SCIS 2010, 2D4-3.
9) 笹原大揮, 宮地充子, and 横川広幸. 新しい加法公式に基づく効率的な予備計算手法について, SCIS 2010, 2D4-4.

手法	全体の計算量		全体の乗算数				レジスタ数
	160bits	256bits	160bits		256bits		
	($S = 0.8M$)	($S = 0.8M$)	$I = 7M$	$I = 20M$	$I = 7M$	$I = 20M$	
Binary($\mathcal{A}$)	$239I + 478M + 398S(796M)$	$383I + 766M + 638S(1276M)$	2469	5576	4468	9447	-
Binary($\mathcal{J}$)	$1I + 881M + 1593S(2155M)$	$1I + 1409M + 2553S(3451M)$	2162	2175	3458	3471	-
NAF($\mathcal{A}$)	$213I + 427M + 374S(726M)$	$341I + 683M + 597S(1161M)$	2222	5000	3550	7987	-
NAF($\mathcal{J}$)	$I + 694M + 1486S(1883M)$	$I + 1110M + 2382S(3016M)$	1890	1903	3023	3036	-
ternary(RL)($\mathcal{A}$)	$168I + 1077M + 505S(1481M)$	$269I + 1723M + 808S(2369M)$	2658	4846	4253	7753	-
ternary(RL)($\mathcal{J}$)	$1I + 1450M + 1185S(2635M)$	$1I + 2318M + 2370S(4214M)$	2642	2655	4221	4234	-
ternary(LR)($\mathcal{A}$)	$169I + 839M + 471S(1215M)$	$270I + 1443M + 753S(1946M)$	2396	4588	3833	7337	-
ternary(LR)($\mathcal{J}$)	$2I + 1078M + 1275S(2098M)$	$2I + 1724M + 2043S(3358M)$	2112	2138	3372	3398	17
ternary/binary($\mathcal{J}$)	$1I + 945M + 1246S(1942M)$	$1I + 1511M + 1993S(3105M)$	1949	1962	3112	3125	14
EX-CoDT(RL)($\mathcal{J}$)	$1I + 1613M + 982S(2398M)$	$1I + 2579M + 1571S(3836M)$	2405	2418	3843	3856	12
Co-TPLADD(LR)($\mathcal{J}$)	$1I + 1430M + 1292S(2464M)$	$1I + 2278M + 2060S(3926M)$	2471	2484	3933	3946	12

表 5.2 理論上の計算量の比較