コンピュータセキュリティ　14−14
（2001. 7. 25）

# Montgomery 型楕円曲線における高速なスカラー倍同時計算法

秋下 徹

ソニー株式会社 インターネット研究所
〒 141-0001 東京都品川区北品川 6-7-35
E-mail: akishita@pal.arch.sony.co.jp

あらまし　我々は，$GF(p)$ 上の Montgomery 型楕円曲線において $kP+lQ$ の $x$ 座標を同時に計算する新しい方法を提案する．$kP+lQ$ の $x$ 座標の計算は楕円曲線 DSA 署名時に必要となる．提案法は，$kP$, $lQ$ それぞれの Montgomery 型楕円曲線におけるスカラー倍計算法と $Y$ 座標復元を用いた方法より約 25%高速である．また，本手法は，複数の座標系の組合せと NAF を用いた Weierstrass 型楕円曲線におけるスカラー倍同時計算法と同等の計算量を実現している．

キーワード　Montgomery 型楕円曲線，楕円曲線 DSA 署名，スカラー倍同時計算法，素体

# Fast Simultaneous Scalar Multiplication on Elliptic Curve with Montgomery Form

Toru Akishita

Internet Laboratories, Sony Corporation
6-7-35 Kitashinagawa Shinagawa-ku, Tokyo, 141-0001 Japan
E-mail: akishita@pal.arch.sony.co.jp

Abstract　We propose the new method to compute $x$-coordinate of $kP+lQ$ simultaneously on the elliptic curve with Montgomery form over $GF(p)$. To compute $x$-coordinate of $kP+lQ$ is required in ECDSA signature verification. The proposed method is about 25% faster than the method using Montgomery scalar multiplication and the recovery of $Y$-coordinate of $kP$, $lQ$. Also, our method requires about as large computational cost as Weierstrass simultaneous scalar multiplication using mixed coordinates and NAF.

key words　elliptic curve with Montgomery form, ECDSA, simultaneous scalar multiplication, prime field

## 1 はじめに

一般に $GF(p)$ 上の楕円曲線は Weierstrauss 型曲線 $y^2 = x^3 + ax + b$ で表されるが，高速化に適した曲線として Montgomery 型楕円曲線 $By^2 = x^3 + Ax^2 + x$ が提案されている [10]．Montgomery 型楕円曲線では楕円曲線上の点のスカラー倍演算 $kP$ を事前計算された点を必要とせずに高速に計算でき，さらに Timing Attack [6] に耐性が高いことが知られている [11]．この方法は，$GF(2^n)$ 上の楕円曲線 $y^2 + xy = x^3 + ax^2 + b$ にも拡張されている [9]．

楕円曲線 DSA 署名 (ECDSA) [1] の署名検証処理では，楕円曲線上の点 $P$, $Q$ それぞれのスカラー倍点 $kP$, $lQ$ を加算した点 $kP + lQ$ の $x$ 座標を求める必要がある．Weierstrass 型楕円曲線において $kP + lQ$ を同時に計算する方法は Shamir's Trick [4] として知られている [2, 5, 13] が，Montgomery 型楕円曲線では，$kP + lQ$ を同時に計算する方法はこれまで知られていなかった．

そこで，本稿において我々は Motgomery 型楕円曲線において $kP + lQ$ を同時に計算する方法 (スカラー倍同時計算法) を提案する．以下，2 章では Montgomery 型楕円曲線上での演算について，3 章では Weierstrass 型楕円曲線におけるスカラー倍同時計算法について述べる．4 章では提案法である Montgomery 型楕円曲線におけるスカラー倍同時計算法について述べる．5 章では各手法の比較を行ない，6 章でまとめと今後の課題について述べる．

## 2 Montgomery 型楕円曲線

$GF(p)$ 上の Montgomery 型楕円曲線 $E_M$ は，素因数分解を高速に行なうために導入された以下の形で表される楕円曲線である [10]．

$$E_M : By^2 = x^3 + Ax^2 + x \quad (A, B \in GF(p),\ (A^2 - 4)B \neq 0)$$

Montgomery 型楕円曲線では，曲線上の 2 点の差分の点の $x$ 座標が既知であれば，$y$ 座標を用いないで点の加算公式および 2 倍算公式が導くことができる．Affine 座標 $(x, y)$ は $x = X/Z$, $y = Y/Z$ により Projective 座標 $(X, Y, Z)$ に変換され，$P_0 = (X_0, Y_0, Z_0)$, $P_1 = (X_1, Y_1, Z_1)$ を $E_M$ 上の点とし，$P_2 = (X_2, Y_2, Z_2) = P_1 + P_0$, $P_3 = (X_3, Y_3, Z_3) = P_1 - P_0$ とすると，$E_M$ 上の点の加算公式および 2 倍算公式は以下のようになる．

**加算公式** $P_2 = P_1 + P_0\ (P_1 \neq P_0)$

$$\begin{aligned} X_2 &= Z_3((X_0 - Z_0)(X_1 + Z_1) + (X_0 + Z_0)(X_1 - Z_1))^2 \\ Z_2 &= X_3((X_0 - Z_0)(X_1 + Z_1) - (X_0 + Z_0)(X_1 - Z_1))^2 \end{aligned} \tag{1}$$

**2 倍算公式** $P_2 = 2P_0$

$$\begin{aligned} 4X_0Y_0 &= (X_0 + Z_0)^2 - (X_0 - Z_0)^2 \\ X_2 &= (X_0 + Z_0)^2(X_0 - Z_0)^2 \\ Z_2 &= (4X_0Y_0)((X_0 - Z_0)^2 + ((A + 2)/4)(4X_0Y_0)) \end{aligned}$$

$P_2 = (X_2, Z_2)$ は，加算・2 倍算公式共に $Y$ 座標を用いずに計算することができる．以下，素体上の乗算，2 乗算，逆元算に必要な計算量をそれぞれ $M$, $S$, $I$ とする．$(A + 2)/4$ が事前計算されている場合には，点の加算に必要な計算量は $4M + 2S$ となり，さらに $Z_3 = 1$ の場合には $3M + 2S$ となる．また，点の 2 倍算に必要な計算量は $3M + 2S$ である．

$P = (x, y)$ のスカラー倍点 $kP$ ($k = (k_t \cdots k_0)_2$, $k_t = 1$) を計算するためには，$m_i = (k_t \cdots k_i)_2$ に対して，常に $m_iP$ と $(m_i + 1)P$ の 2 点を計算しながら演算を進めていく．$k_i = 0$ であれば，$m_i = 2m_{i+1}$ より $m_iP = 2m_{i+1}P$, $(m_i + 1)P = (m_{i+1} + 1)P + m_{i+1}P$ となる．$k_i = 1$ であれば，$m_i = 2m_{i+1} + 1$ より $m_iP = (m_{i+1} + 1)P + m_{i+1}P$, $(m_i + 1)P = 2(m_{i+1} + 1)P$ となる．このとき，$\{P, 2P\}$ から $\{kP, (k + 1)P\}$ を計算でき，$k$ のビット長を $|k|$ とすると，点の加算が $|k| - 1$ 回，点の 2 倍算が $|k|$ 回必要となる．$(m_{i+1} + 1)P$ と $m_{i+1}P$ の差分点は $P$ であるから，(1) 式で $(X_3, Z_3) = (x, 1)$ とおける．以上より，Montgomery 型楕円曲線におけるスカラー倍計算法に必要な計算量は，$(6|k| - 3)M + (4|k| - 2)S$ となる．また，Montgomery 型楕円曲線におけるスカラー倍計算法は Timing Attack に対して耐性が高いという特徴をもっているため，ECDSA の署名生成処理や ECDH などでは効果的な計算法である．

ECDSA の署名検証処理においては，異なる 2 点 $P$, $Q$ のスカラー倍点 $kP$, $lQ$ の加算点 $kP + lQ$ の $x$ 座標を求める計算が必要になる．$kP$, $lQ$ はスカラー倍算によって求めることができるが，$kP$, $lQ$ の加算を行なう際には 2 点の差分の点が既知ではないため (1) 式を用いることができない．そこで，別の加算公式を用いて $kP + lQ$ を求めるためには，$Y$ 座標を復元する必要性がある．[12] によると，$Y$ 座標を復元するために必要な計算量は $12M + S$ である．また，$kP$, $lQ$ から $kP + lQ$ を求めるために必要な計算量は $10M + 2S$ であり，$x = X/Z$ より $x$ 座標を求めるために必要な計算量は $M + I$ となる．以上より，$|k| = |l|$ と仮定すると，$kP + lQ$ の $x$ 座標を求めるために必要な計算量の見積りは $(12|k| + 29)M + 8|k|S + I$ となる．

## 3 Weierstrass 型楕円曲線におけるスカラー倍同時計算法

$GF(p)$ 上の Weierstrass 型楕円曲線 $E_W$ は,

$$E_W : y^2 = x^3 + ax + b \ \ (a, b \in GF(p),\ 4a^2 + 27b^3 \neq 0).$$

で表される. Weierstrass 型楕円曲線では, $kP + lQ$ を同時に計算できることが知られている [2, 5, 13]. その中でも最も効率的に $kP + lQ$ を計算する方法は複数の座標系の組合せ [3] と NAF [7] を用いたスカラー倍同時計算法である.

複数の座標系の組合せを用いた方法では, 点の加算の前に行なう点の 2 倍算に対しては Modified Jacobian 座標から Jacobian 座標への 2 倍算 ($J \leftarrow 2J^m$) を用い, その他の 2 倍算に対しては Modified Jacobian 座標から Modified Jacobian 座標への 2 倍算 ($J^m \leftarrow 2J^m$) を用いる. また, 点の加算に対しては Jacobian 座標から Modified Jacobian 座標への加算 ($J^m \leftarrow J + A$) を用いる. ここで, $A$, $J$, $J^m$ はそれぞれ Affine 座標, Jacobian 座標, Modified Jacobian 座標を示しており, $J \leftarrow 2J^m$, $J^m \leftarrow 2J^m$, $J^m \leftarrow J + A$ に必要な計算量はそれぞれ $3M + 4S$, $4M + 4S$, $9M + 5S$ となる. また, NAF は非 0 のビットが約 1/3 となる特性をもっており, 点の加算回数を減らすことができる. ここで, 事前計算に必要な点は $P + Q$, $P - Q$ であり, これらの点は Affine 座標を用いて計算する. 以下に, 複数の座標系の組合せと NAF を用いた Weierstrass 型楕円曲線におけるスカラー倍同時計算法を示す.

---

Algorithm 1: Weierstrass Simultaneous Scalar Multiplication Using Mixed Coordinates and NAF

---

Input: $k = (k_t \cdots k_1 k_0)_2$, $l = (l_t \cdots l_1 l_0)_2$, $P, Q \in E_W$ ($k_t$ **or** $l_t = 1$).
Output: $x$-coordinate of $W = kP + lQ$.

1. Compute $P + Q$, $P - Q$ using affine coordinates.
2. Let $(k'_{t'} \cdots k'_1 k'_0)$ and $(l'_{t'} \cdots l'_1 l'_0)$ be N AF of $k$ and $l$ ($k'_{t'}$ or $l'_{t'} = 1$).
3. $W \leftarrow k'_{t'} P + l'_{t'} Q$.
4. For $i$ from $t' - 1$ downto 0 do:
   4.1 if $(k'_i, l'_i) = (0, 0)$ then
   $W \leftarrow 2W$ $(J^m \leftarrow 2J^m)$;
   4.2 else then
   $W \leftarrow 2W$ $(J \leftarrow 2J^m)$,
   $W \leftarrow W + (k'_i P + l'_i Q)$ $(J^m \leftarrow J + A)$.
5. Compute $x$-coordinate of $W$.

---

1. において必要な計算量は $4M + 2S + I$ である. また, $(k'_i, l'_i) = (0, 0)$ となる確率は約 $(1 - 1/3)^2 = 4/9$ であるから, NAF によるビットの桁上がりが起こると仮定すると (実際には $|k| = |l|$ であれば 8/9 の確率で起こる), 4.1. において必要な計算量は $(4|k|/9) \cdot (4M + 4S)$ となる. 同様に, 4.2. において必要な計算量は $(5|k|/9) \cdot ((3M + 4S) + (9M + 5S))$ となる. 最後に, 5. において必要な計算量は $M + S + I$ である. 以上より, 複数の座標系の組合せと NAF を用いた Weierstrass 型楕円曲線上のスカラー倍同時計算法に必要な計算量の見積りは $(76|k| + 45)/9 \cdot M + (61|k| + 27)/9 \cdot S + 2I$ となる.

## 4 Montgomery 型楕円曲線におけるスカラー倍同時計算法

本稿で提案する Montgomery 型楕円曲線上において $kP + lQ$ を同時に計算する方法について説明する.
まず, $m_i = (k_t \cdots k_i)_2$, $n_i = (l_t \cdots l_i)_2$ に対して, 4 点の集合 $G_i$ を

$$G_i = \left\{ \begin{array}{c} m_i P + n_i Q, \\ m_i P + (n_i + 1)Q, \\ (m_i + 1)P + n_i Q, \\ (m_i + 1)P + (n_i + 1)Q \end{array} \right\} \tag{2}$$

と定義する. このとき, $(k_i, l_i)$ のそれぞれの場合について $G_{i+1}$ から $G_i$ を計算する方法を以下に示す.

1. $(k_i, l_i) = (0, 0)$
   $m_i = 2m_{i+1}$, $n_i = 2n_{i+1}$ となるから,

$$\begin{aligned} m_i P + n_i Q &= 2(m_{i+1} P + n_{i+1} Q) \\ m_i P + (n_i + 1)Q &= (m_{i+1} P + (n_{i+1} + 1)Q) + (m_{i+1} P + n_{i+1} Q) \\ (m_i + 1)P + n_i Q &= ((m_{i+1} + 1)P + n_{i+1} Q) + (m_{i+1} P + n_{i+1} Q) \\ (m_i + 1)P + (n_i + 1)Q &= ((m_{i+1} + 1)P + n_{i+1} Q) + (m_{i+1} P + (n_{i+1} + 1)Q). \end{aligned}$$

により，$G_{i+1}$ から $G_i$ の全ての要素を求めることができる．このとき，$G_{i+1}$ の要素 $(m_{i+1}+1)P+(n_{i+1}+1)Q$ は，$G_i$ を求めるためには必要がないことが分かる．

2. $(k_i,l_i)=(0,1)$
$m_i=2m_{i+1}$, $n_i=2n_{i+1}+1$ となるから，

$$\begin{aligned}
m_iP+n_iQ &= (m_{i+1}P+(n_{i+1}+1)Q))+(m_{i+1}P+n_{i+1}Q)\\
m_iP+(n_i+1)Q &= 2(m_{i+1}P+(n_{i+1}+1)Q)\\
(m_i+1)P+n_iQ &= ((m_{i+1}+1)P+(n_{i+1}+1)Q)+(m_{i+1}P+n_{i+1}Q)\\
(m_i+1)P+(n_i+1)Q &= ((m_{i+1}+1)P+(n_{i+1}+1)Q)+(m_{i+1}P+(n_{i+1}+1)Q).
\end{aligned}$$

により，$G_{i+1}$ から $G_i$ の全ての要素を求めることができる．このとき，$G_{i+1}$ の要素 $(m_{i+1}+1)P+n_{i+1}Q$ は，$G_i$ を求めるためには必要がないことが分かる．

3. $(k_i,l_i)=(1,0)$
$m_i=2m_{i+1}+1$, $n_i=2n_{i+1}$ となるから，

$$\begin{aligned}
m_iP+n_iQ &= ((m_{i+1}+1)P+n_{i+1}Q)+(m_{i+1}P+n_{i+1}Q)\\
m_iP+(n_i+1)Q &= ((m_{i+1}+1)P+(n_{i+1}+1)Q)+(m_{i+1}P+n_{i+1}Q)\\
(m_i+1)P+n_iQ &= 2((m_{i+1}+1)P+n_{i+1}Q)\\
(m_i+1)P+(n_i+1)Q &= ((m_{i+1}+1)P+(n_{i+1}+1)Q)+((m_{i+1}+1)P+n_{i+1}Q)
\end{aligned}$$

により，$G_{i+1}$ から $G_i$ の全ての要素を求めることができる．このとき，$G_{i+1}$ の要素 $m_{i+1}P+(n_{i+1}+1)Q$ は，$G_i$ を求めるためには必要がないことが分かる．

4. $(k_i,l_i)=(1,1)$
$m_i=2m_{i+1}+1$, $n_i=2n_{i+1}+1$ となるから，

$$\begin{aligned}
m_iP+n_iQ &= ((m_{i+1}+1)P+n_{i+1}Q))+(m_{i+1}P+(n_{i+1}+1)Q)\\
m_iP+(n_i+1)Q &= ((m_{i+1}+1)P+(n_{i+1}+1)Q)+(m_{i+1}P+(n_{i+1}+1)Q)\\
(m_i+1)P+n_iQ &= ((m_{i+1}+1)P+(n_{i+1}+1)Q)+((m_{i+1}+1)P+n_{i+1}Q)\\
(m_i+1)P+(n_i+1)Q &= 2((m_{i+1}+1)P+(n_{i+1}+1)Q)
\end{aligned}$$

により，$G_{i+1}$ から $G_i$ の全ての要素を求めることができる．このとき，$G_{i+1}$ の要素 $m_{i+1}P+n_{i+1}Q$ は，$G_i$ を求めるためには必要がないことが分かる．

以上より，$(k_i,l_i)$ の全ての場合において $G_{i+1}$ から $G_i$ の全ての要素を求めることができ，さらに $G_i$ を計算するためには，$G_{i+1}$ の要素 $(m_{i+1}+1-k_i)P+(n_{i+1}+1-l_i)Q$ が必要がないことが分かる．ここで，3 点の集合 $G_i'$ を

$$G_i'=G_i-\{(m_i+1-k_{i-1})P+(n_i+1-l_{i-1})Q\} \tag{3}$$

と定義すると，$G_i$ の全ての要素は $G_{i+1}'$ から計算することができる．つまり，$G_i'$ は $G_{i+1}'$ から計算することができる．ここで，$G_{i+1}'$ から $G_i$ の計算は $(k_i,l_i)$ に依存し，$G_i'$ の定義は $(k_{i-1},l_{i-1})$ に依存することから，$G_{i+1}'$ から $G_i'$ の計算は $(k_i,l_i,k_{i-1},l_{i-1})$ に依存することが分かる．以上より，$k$, $l$ のビットを先読みすることにより $G_i'$ の 3 点を決定することができるので，1 ビットにつき 1 回の楕円曲線上の加算または 2 倍算を減らすことができる．

例えば，$(k_i,l_i,k_{i-1},l_{i-1})=(0,0,0,0)$ の場合には，$m_i=2m_{i+1}$, $n_i=2n_{i+1}$ で，

$$G_{i+1}'=\left\{\begin{array}{c} m_{i+1}P+n_{i+1}Q,\\ m_{i+1}P+(n_{i+1}+1)Q,\\ (m_{i+1}+1)P+n_{i+1}Q \end{array}\right\}, G_i'=\left\{\begin{array}{c} m_iP+n_iQ,\\ (m_i+1)P+n_iQ,\\ m_iP+(n_i+1)Q \end{array}\right\}$$

となるから，

$$\begin{aligned}
m_iP+n_iQ &= 2(m_{i+1}P+n_{i+1}Q)\\
m_iP+(n_i+1)Q &= (m_{i+1}P+(n_{i+1}+1)Q)+(m_{i+1}P+n_{i+1}Q)\\
(m_i+1)P+n_iQ &= ((m_{i+1}+1)P+n_{i+1}Q)+(m_{i+1}P+n_{i+1}Q)
\end{aligned} \tag{4}$$

により，$G'_i$ を計算することができる．ここで，$G'_i = \{T_0[i], T_1[i], T_2[i]\}$ と定義すると，式(6)は，

$$T_0[i] = 2T_0[i+1]$$
$$T_1[i] = T_1[i+1] + T_0[i+1]\ (T_1[i+1] - T_0[i+1] = Q)$$
$$T_2[i] = T_2[i+1] + T_0[i+1]\ (T_2[i+1] - T_0[i+1] = P).$$

と表せる．

また，式(3),(4)より $G'_i$ の初期値 $G'_{t+1}$ を

$$G_{t+1} = \{\mathcal{O}, Q, P, P+Q\}$$
$$G'_{t+1} = G_{t+1} - \{(1-k_t)P + (1-l_t)Q\},$$

と定義する．ここで，$\mathcal{O}$ は無限遠点を表す．$G'_{i+1}$ から $G'_i$ の計算を繰り返すことにより，$G'_{t+1}$ から $G'_1$ を求め，最後に $G'_1$ から $kP+lQ$ を求めることができる．以下に詳細なアルゴリズムを示す．ただし，$T_i + T_j\ (P)$ は $T_i$ と $T_j$ の差分点が $P$ であることを示すものとする．

---

Algorithm 2: Montgomery Simultaneous Scalar Multiplication

---

Input: $k = (k_t \cdots k_1 k_0)_2,\ l = (l_t \cdots l_1 l_0)_2,\ P, Q \in E_M\ (k_t \textbf{ or } l_t = 1)$.
Output: $x$-coordinate of $W = kP + lQ$.

1. Compute $P+Q,\ P-Q$.
2. If $(k_t, l_t) = (0,1)$ then $T_0 \leftarrow \mathcal{O},\ T_1 \leftarrow Q,\ T_2 \leftarrow P+Q$;
   else if $(k_t, l_t) = (1,0)$ then $T_0 \leftarrow \mathcal{O},\ T_1 \leftarrow P,\ T_2 \leftarrow P+Q$;
   else then $T_0 \leftarrow Q,\ T_1 \leftarrow P,\ T_2 \leftarrow P+Q$.
3. For $i$ from $t$ downto 1 do:
   - 3.1. if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,0,0,0)$ then
     $T_2 \leftarrow T_2 + T_0\ (P),\ T_1 \leftarrow T_1 + T_0\ (Q),\ T_0 \leftarrow 2T_0$;
   - 3.2. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,0,0,1)$ then
     $T_2 \leftarrow T_2 + T_1\ (P-Q),\ T_1 \leftarrow T_1 + T_0\ (Q),\ T_0 \leftarrow 2T_0$;
   - 3.3. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,0,1,0)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P),\ T_0 \leftarrow 2T_0,\ T_2 \leftarrow T_2 + T\ (P-Q)$;
   - 3.4. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,0,1,1)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P),\ T_0 \leftarrow T + T_0\ (Q),\ T_2 \leftarrow T_2 + T\ (P-Q)$;
   - 3.5. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,1,0,0)$ then
     $T_2 \leftarrow T_2 + T_0\ (P+Q),\ T_0 \leftarrow T_1 + T_0\ (Q),\ T_1 \leftarrow 2T_1$;
   - 3.6. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,1,0,1)$ then
     $T_2 \leftarrow T_2 + T_1\ (P),\ T_0 \leftarrow T_1 + T_0\ (Q),\ T_1 \leftarrow 2T_1$;
   - 3.7. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,1,1,0)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P+Q),\ T_0 \leftarrow T + T_0\ (Q),\ T_2 \leftarrow T_2 + T\ (P)$;
   - 3.8. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (0,1,1,1)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P+Q),\ T_0 \leftarrow 2T,\ T_2 \leftarrow T_2 + T\ (P)$;
   - 3.9. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (1,0,0,0)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P+Q),\ T_0 \leftarrow T + T_0\ (P),\ T_2 \leftarrow 2T$;
   - 3.10. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (1,0,0,1)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P+Q),\ T_0 \leftarrow T + T_0\ (P),\ T_2 \leftarrow T_2 + T\ (Q)$;
   - 3.11. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (1,0,1,0)$ then
     $T_0 \leftarrow T_1 + T_0\ (P),\ T_2 \leftarrow T_2 + T_1\ (Q),\ T_1 \leftarrow 2T_1$;
   - 3.12. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (1,0,1,1)$ then
     $T_0 \leftarrow T_2 + T_0\ (P+Q),\ T_2 \leftarrow T_2 + T_1\ (Q),\ T_1 \leftarrow 2T_1$;
   - 3.13. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (1,1,0,0)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P),\ T_0 \leftarrow T + T_0\ (P-Q),\ T_2 \leftarrow T_2 + T\ (Q)$;
   - 3.14. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (1,1,0,1)$ then
     $T \leftarrow T_1,\ T_1 \leftarrow T_2 + T_0\ (P),\ T_0 \leftarrow T + T_0\ (P-Q),\ T_2 \leftarrow 2T_2$;
   - 3.15. else if $(k_i, l_i, k_{i-1}, l_{i-1}) = (1,1,1,0)$ then

$T_0 \leftarrow T_1 + T_0\ (P-Q),\ T_1 \leftarrow T_2 + T_1\ (Q),\ T_2 \leftarrow 2T_2;$

3.16. else then

$T_0 \leftarrow T_2 + T_0\ (P),\ T_1 \leftarrow T_2 + T_1\ (Q),\ T_2 \leftarrow 2T_2.$

4. If $(k_0, l_0) = (0,0)$ then $W \leftarrow 2T_0$;
   else if $(k_0, l_0) = (0,1)$ then $W \leftarrow T_1 + T_0\ (Q)$;
   else if $(k_0, l_0) = (1,0)$ then $W \leftarrow T_1 + T_0\ (P)$;
   else then $W \leftarrow T_1 + T_0\ (P-Q)$.
5. Compute $x$-coordinate of $W$ by $x = X/Z$.

---

提案法の計算量について考察する．1. において必要な計算量は $4M + 2S + I$ となる．2, 3, 4. では Projective 座標を用いる．3. では $k$ の 1 ビットにつき，2 回の加算公式と 1 回の 2 倍算公式，または 3 回の加算公式を用いている．どちらの場合でも $k$ の 1 ビット当たりの計算量は $9M + 6S$ となる．よって，3. において必要な計算量は $9(|k|-1)M + 6(|k|-1)S$ である．4. において必要な計算量は $3M + 2S$ で，5. において必要な計算量は $M + I$ となる．以上より，提案法において必要な計算量は $(9|k|-1)M + (6|k|-2)S + 2I$ となる．

## 5 各手法の比較

4 章で提案した手法を 2 章，3 章で示した方法と比較する．まず，各手法に必要な計算量を示す．

| | $M$ | $S$ | $I$ |
|---|---|---|---|
| Montgomery + $Y$ 座標復元 | $12\|k\| + 29$ | $8\|k\|$ | 1 |
| Weierstrass 同時計算法 (Algorithm 1) | $(76\|k\| + 45)/9$ | $(61\|k\| + 27)/9$ | 2 |
| Montgomery 同時計算法 (Algorithm 2) | $9\|k\| - 1$ | $6\|k\| - 2$ | 2 |

$|k| = 160$ として，さらに $S/M = 0.8$, $I/M = 30$ [8] と仮定した場合にはそれぞれの計算量は以下のようになる．

| | $M$ | $S$ | $I$ | $M\ (S/M = 0.8, I/M = 30)$ |
|---|---|---|---|---|
| Montgomery + $Y$ 座標復元 | 1949 | 1280 | 1 | 3003 |
| Weierstrass 同時計算法 (Algorithm 1) | 1356 | 1087 | 2 | 2286 |
| Montgomery 同時計算法 (Algorithm 2) | 1439 | 958 | 2 | 2265 |

提案法である Montgomery 型楕円曲線におけるスカラー倍同時計算法は，$kP$, $lQ$ それぞれの Montgomery 型楕円曲線におけるスカラー倍計算法と $Y$ 座標復元を用いた方法と比較して約 25%高速であることが分かる．さらに，提案法の計算量は，複数の座標系の組合せと NAF を用いた Weierstrass 型楕円曲線におけるスカラー倍同時計算法とほぼ同等の計算量である．

## 6 まとめと今後の課題

我々は Montgomery 型楕円曲線におけるスカラー倍同時計算法を提案した．提案法は，$kP$, $lQ$ それぞれの Montgomery 型楕円曲線におけるスカラー倍計算法と $Y$ 座標復元を用いた方法と比較して約 25%高速であり，複数の座標系の組合せと NAF を用いた Weierstrass 型楕円曲線におけるスカラー倍同時計算法とほぼ同等の計算量を実現している．今後の課題としては，Window Method を用いた Weierstrass 型楕円曲線におけるスカラー倍同時計算法との比較や提案法の $GF(2^n)$ 上の楕円曲線への適用が考えられる．

## 参考文献


[1] ANSI X9.62-1998, *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*, Working Draft, September 1998.

[2] M. Brown, D. Hankerson, J. Lopez, and A. Menezes, "Software Implementation of the NIST Elliptic Curves Over Prime Fields", *Topics in Cryptology - CT-RSA 2001*, LNCS 2020, pp. 250-265, 2001.

[3] H. Cohen, A. Miyaji, and T. Ono, "Efficient Elliptic Curve Exponentiation Using Mixed Coordinates", *Advances in Cryptography - ASIACRYPT '98*, LNCS 1514, pp. 51-65, 1998.

[4] T. ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", *IEEE Transaction on Information Theory*, Vol. 31, pp. 469-472, 1985.

[5] D. Hankerson, J. Lopéz, and A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields", *Cryptographic Hardware and Embedded Systems - CHES 2000*, LNCS 1965, pp. 3-24, 2000.

[6] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems", *Advances in Cryptology - CRYPTO '96*, LNCS 1109, pp. 104-113, 1996.

[7] IEEE P1363, *Standard Specifications for Public Key Cryptography*, Draft Version 13, November 1999. http://grouper.ieee.org/groups/1363/

[8] C. H. Lim and H. S. Hwang, "Fast Implementation of Elliptic Arithmetic in $GF(p^n)$", *Public Key Cryptography - PKC 2000*, LNCS 1751, pp. 405-421, 2000.

[9] J. Lopéz and R. Dahab, "Fast Multiplication on Elliptic Curves over $GF(2^m)$ without precomputation", *Cryptographic Hardware and Embedded Systems – CHES '99*, LNCS 1717, pp. 316-327, 1999.

[10] P. L. Montgomery, "Speeding the Pollard and Elliptic Curve Methods of Factorization", *Mathematics of Computation*, vol. 48, pp. 243-264, 1987.

[11] K. Okeya, H. Kurumatani, and K. Sakurai, "Elliptic Curves with the Montgomery-Form and Their Cryptographic Applications", *Public Key Cryptography – PKC 2000*, LNCS 1751, pp. 238-257, 2000.

[12] K. Okeya and K. Sakurai, "A Scalar Multiplication Algorithm on a Montgomery-form Elliptic Curve", *Proceedings of the 2001 Symposium on Cryptography and Information Security – SCIS 2001*, pp. 311-316, 2001.

[13] E. D. Win, S. Mister, B. Preneel, and M. Wiener, "On the Performance of Signature Schemes Based on Elliptic Curves", *Algorithm Number Theory – ANTS III*, LNCS 1423, pp. 252-266, 1998.