社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2006－CSEC－34（4）
2006／7／20

Tateペアリングの効率的なアルゴリズム

白勢 政明† 高木 剛† 岡本 栄司††

† 公立はこだて未来大学情報アーキテクチャ学科
〒041-8655 函館市亀田中野町 116-2
†† 筑波大学システム情報工学研究科
〒305-8573 つくば市天王台 1-1-1

あらまし Tateペアリングを高速に計算する方法として，Duursma-Leeアルゴリズムとその改良版であるη_Tペアリングがある．これらのアルゴリズムは，(1) ある有限体での計算，(2)6次拡大体での計算，(3) もとの有限体でのべき乗計算と値の更新，の3ステップからなる．η_Tペアリングのアルゴリズムでは，(3) の部分において3乗根を計算する必要がある．本稿では，(3) の部分を改良しi番目のループでは各値が改良前のアルゴリズムの3^i乗となるようにすることで，3乗根の計算を必要としないη_Tペアリングのアルゴリズムを提案する．また，Diffie-Hellmanペアの検証には2つのペアリングの計算が必要であるが，2つのペアリングを同時に計算し (2) の部分を改良することで，計算量が削減された検証アルゴリズムを提案する．提案方式は従来方式と比較して30〜40%の高速化が達成できる．
キーワード Tateペアリング，η_Tペアリング，Diffie-Hellmanペア

Efficient Algorithm for Tate Pairing

Masaaki SHIRASE†, Tsuyoshi TAKAGI†, and Eiji OKAMOTO††

† Future University-Hakodate School of Systems Information Science, 116-2 Kamedanakano-cho Hakodate Hokkaido, 041-8655, Japan
†† Graduate School of Systems and Information Engineering, University of Tsukuba, 1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, Japan

Abstract Duursma-Lee algorithm and its variant (η_T pairing) can efficiently compute Tate pairing over supersingular curves with small characteristic. The algorithms consist of three main steps: (1) computations in a finite field. (2) computations in the extension field of degree 6, (3) exponentiation calculation in the base field. We have to compute relatively slow cube roots in step (3). This paper proposes some novel algorithms for computing η_T pairing without cube root. The proposed algorithms compute 3^i-th power for each value in the i-th loop in step (3), and thus the cube roots are removed (but we obtain the pairing value powered by 3^k for some k). Moreover we propose some efficient algorithms in which two pairings are computed simultaneously and the part of (2) is improved. The proposed algorithms enable to enhance the speed for verifying a Diffie-Hellman pair. We can achieve 30〜40% faster computation comparing with the previously known methods.
Key words Tate pairing, η_T pairing Diffie-Hellman pair

1. はじめに

楕円曲線上の双線形ペアリングを用いることで，IDベース暗号[4]やshort signature[6]のような新しい暗号システムが可能となる．楕円曲線上の双線形ペアリングとして，Weilペアリングと Tateペアリングがあるが，計算が効率的なため近年では Tateペアリングが用いられている．IDベース暗号では，従来の公開鍵暗号とは異なり，ユーザの IDのような既知の情報を公開鍵とすることができる．また，Tateペアリングを用いる short signatureでは，署名長が従来の楕円曲線を用いる署名のおよそ半分になる．

Tateペアリングの効率的なアルゴリズムは Millerによって最初に提案された[13]．Duursma と Leeにより，用いる楕円曲線を限定することで，より効率的なアルゴリズムがあることが示され[8]，このアルゴリズムも改良されている[12]．更に，ループの回数が約50%となるη_Tペアリングが提案されてい

る[2]. Tateペアリングとη_Tペアリングは，あるべき乗をとることにより値が一致するという関係がある．しかしながらこれらアルゴリズムを用いても，他の公開鍵暗号と比較して計算コストが高いため，Tateペアリングやη_Tペアリングのアルゴリズムを効率化することは重要な研究テーマである．計算の効率化のためTateペアリングのハードウェア実装も研究されている[9],[11].

埋め込み次数を大きくできることや，有限体でのべき乗や楕円曲線の点のスカラー倍算にTriple-and-Add法を用いることができる理由で，標数3の体がTateペアリングに適している．本稿も標数3の体の場合を扱う．

Tateペアリングとη_Tペアリングのアルゴリズムのループ内はともに，(1) ある有限体での計算，(2)6次拡大体での計算，(3) 元の有限体でのべき乗計算と値の更新，に分けられる．η_Tペアリングのアルゴリズムでは，(3) の部分で3乗根を計算する必要がある．本稿では，(3) の部分を改良しi番目のループでは各値が元のアルゴリズムの3^i乗となるようにすることで，3乗根の計算を必要としないη_Tペアリングのアルゴリズムを提案する．この改良により計算量が約15%削減される．また，Diffie–Hellmanペアの検証には2つのペアリングの計算が必要であるが，拡大体の構造により (2) の部分の計算が簡略され，計算量が30〜40%削減された検証アルゴリズムが得られることを示す．

Diffie–Hellmanペアの効率的な検証方法としてバッチ検証がある[7],[10],[15]．これは複数のペアの検証を1回の検証プロセスで行う手法であり，2回のペアリングの計算によりなされる．本稿は高速な1回の検証プロセスを提案する．そのため本稿の提案手法とバッチ検証とを組み合わせることができ，複数のDiffie–Hellmanペアを約1.4回のペアリングの計算量によって行うことができるようになる．

2. Tateペアリング

近年，IDベース暗号[4]，short signature[6]，効率的なブロードキャスト暗号[5]など，楕円曲線のTateペアリングやその類似を用いた暗号システムの研究が盛んである．Tateペアリングを用いる暗号システムには，IDベース暗号[4]や，short signature[6]がある．qを素数pのべき，$\mathbb{F}_q$を元がq個の有限体，Eを$\mathbb{F}_q$上定義された楕円曲線とする．$\mathcal{O}$をEの無限遠点とする．lをqと互いに素な$l\,|\,\#E(\mathbb{F}_q)$を満たす正整数とし，kを$l\,|\,(q^k-1)$となるような最小の正整数とする．すると，Tateペアリングは写像

$$\langle\cdot,\cdot\rangle_l : E(\mathbb{F}_{q^k})[l]\times E(\mathbb{F}_{q^k})/lE(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}^*/(\mathbb{F}_{q^k}^*)^l$$

であり，$P\in E(\mathbb{F}_{q^k})[l]-\{\mathcal{O}\}$に対して，$\langle P,Q\rangle_l \notin (\mathbb{F}_{q^k}^*)^l$となる$Q\in E(\mathbb{F}_{q^k})$が存在し(非退化性)，任意の非負の整数$a$に対して$\langle P,aQ\rangle_l=\langle aP,Q\rangle_l=\langle P,Q\rangle_l^a$が成り立つ(双線形性)．

正確には，$\langle P,Q\rangle_l=g(D)$と定義される．ここで，$g$は因子の等式$l(P)-l(\mathcal{O})=(g)$を満たす関数であり，$D$は$(Q)-(\mathcal{O})$と同値なDのsupportと$g$のsupportとが互いに素な因子である．因子については[14]を参照．

$\langle P,Q\rangle_l\in\mathbb{F}_{q^k}^*/(\mathbb{F}_{q^k}^*)^l$であるから，$\langle P,Q\rangle_l$の値は一意には定まらない．そのため暗号アプリケーションでTateペアリングを用いるときは$\langle P,Q\rangle_l$を$(q^k-1)/l$乗する必要がある．$e(P,Q)=\langle P,Q\rangle_l^{(q^k-1)/l}$と定義されたペアリングをreduced Tate pairingと呼ぶ．

hとlを$(hl)\,|\,\#E(\mathbb{F}_{q^k})$を満たす正整数とする．すると

$$e(P,Q)=\langle P,Q\rangle_l^{(q^k-1)/l}=\langle P,Q\rangle_{(hl)}^{(q^k-1)/(hl)} \qquad (1)$$

が成り立つ[8].

2組の楕円曲線の点のペア(P,Q), (R,S)に対して，これらのペアリングの値$e(P,Q)$と$e(R,S)$が等しいとき，これらはDiffie-Hellmanペアであるという．Diffie–Hellmanペアの検証はshort signatureなどで用いられる．

2.1 拡大次数kとdistortion map

Eを$\mathbb{F}_q$上定義された楕円曲線，lを$l\,|\,\#E(\mathbb{F}_q)$となる素数，kを$l\,|\,(q^k-1)$を満たす最小の正整数とする．すると安全性の理由によりlは約160ビット，$\mathbb{F}_{q^k}$が約1024ビットとなるように選ぶことが標準的である．

Tate pairingを定義するための楕円曲線として，以下のような超特異曲線が研究されてきた[1].

体	曲線の式	k
素体$\mathbb{F}_p$(注1)	$y^2=x^3+(1-b)x+b,\ b\in\{0,1\}$	$k=2$
$\mathbb{F}_{2^n}$	$y^2+y=x^3+x+b,\ b\in\{0,1\}$	$k=4$
$\mathbb{F}_{3^n}$	$y^2=x^3-x\pm b,\ b\in\{-1,1\}$	$k=6$

(注1) pは$p\equiv 2\pmod 3$または$p\equiv 3\pmod 4$を満たす素数．

標数3の体$\mathbb{F}_{3^n}$ではkが大きいことより楕円曲線の定義体を小さくできるため，標数3の体でのTateペアリングの研究が盛んに行われている．本稿も標数3の場合を扱い，楕円曲線

$$E^b: y^2=x^3-x+b,\ b\in\{-1,1\}$$

を用いる．なお，nは6と互いに素でなければならず[8]，$n\equiv\pm1,\ \pm5\pmod{12}$となる．$E^b(\mathbb{F}_{3^n})$の位数は，

$$\#E^b(\mathbb{F}_{3^n})=\begin{cases}3^n+b3^{(n+1)/2}+1, & n\equiv\pm1\pmod{12}\text{のとき}\\ 3^n-b3^{(n+1)/2}+1, & n\equiv\pm5\pmod{12}\text{のとき}\end{cases}$$

となる．また，上のすべての場合で$\#E(\mathbb{F}_{3^{6n}})=(3^{3n}+1)^2$であり，$(3^{3n}+1)\,|\,\#E(\mathbb{F}_{3^{6n}})$となる．これは関係式(1)より$\langle P,Q\rangle_{3^{3n}+1}$を求めることでTateペアリングの値が計算できることを意味する．つまり，最終べきとして$(3^{6n}-1)/(3^{3n}+1)=3^{3n}-1$乗を行うことにより，一意的なTateペアリングの値$e(P,Q)$となる．

標数3の場合，Tateペアリング$\langle P,Q\rangle_l$は$P\in E(\mathbb{F}_{3^{6n}})$, $Q\in E(\mathbb{F}_{3^{6n}})/lE(\mathbb{F}_{3^{6n}})$に対して定義される．しかし実際の暗号アプリケーションでは$P,Q\in E(\mathbb{F}_{3^n})$に対して，ペアリングを定義しなければならない．$P\in E(\mathbb{F}_{3^n})$ならば$P\in E(\mathbb{F}_{3^{6n}})$であるため問題ないが，$Q\in E(\mathbb{F}_{3^n})$に対しては，$E(\mathbb{F}_{3^{6n}})/lE(\mathbb{F}_{3^{6n}})$の元へ写す写像が必要であり，このような写像をdistortion mapと呼ぶ．n mod 6の値に関係なく，distortion map ψは，$(x,y)\in E(\mathbb{F}_{3^n})$に対して，$\sigma^2=-1$を満たす$\sigma\in\mathbb{F}_{3^{2n}}$と$\rho^3=\rho+b$を満たす$\rho\in\mathbb{F}_{3^{3n}}$を用いて$\psi(x,y)=(-x+\rho, y\,\sigma)$

と定義される．distortion map を用いると次のようなペアリング $\hat{e}$ が定義される．

$$\begin{aligned} \hat{e} : E(\mathbb{F}_{3^n}) \times E(\mathbb{F}_{3^n}) &\rightarrow \mathbb{F}^*_{3^{6n}} \\ (P,Q) &\mapsto \hat{e}(P,Q) = e(P,\psi(Q)) \end{aligned}$$

$\hat{e}(P,Q)$ を modified Tate pairing と呼ぶ．$\hat{e}(P,Q)$ の計算には，$l \mid \#E(\mathbb{F}_q)$ となる自然数 l に対して，初めに $\langle P,Q\rangle_l$ を計算し，それからその値を $(q^6-1)/l$ 乗することにより計算される．このべき乗を最終べきと呼ぶ．$\hat{e}$ は非退化性と双線形性を備えている．

3. ペアリングのアルゴリズム

§3 では初めに 2 つの reduced Tate pairing(とその類似) を求めるアルゴリズムを紹介し，2 つの新しいアルゴリズムを提案する．

Algorithm 1: Computation of $\langle P,\psi(Q)\rangle_{3^{3n}+1}$ (The Modified Duursma-Lee Algorithm (char 3) [12])

input: $P=(x_p,y_p)$, $Q=(x_q,y_q) \in E^b(\mathbb{F}_{3^n})[l]$
output: $\langle P,\psi(Q)\rangle_{3^{3n}+1}$

1. $R_0 \leftarrow 1$ (in $\mathbb{F}_{3^{6n}}$), $x_q \leftarrow x_q^3$, $y_q \leftarrow y_q^3$ (in $\mathbb{F}_{3^n}$)
   $d \leftarrow (bn \bmod 3)$
2. **for** $i \leftarrow 0$ **to** $n-1$ **do**
3-a. $x_p \leftarrow x_p^9$, $y_p \leftarrow y_p^9$ (in $\mathbb{F}_{3^n}$)
3-b. $r_0 \leftarrow x_p + x_q + d$ (in $\mathbb{F}_{3^n}$)
4-a. $R_1 \leftarrow -r_0^2 - y_p y_q \sigma - r_0\rho - \rho^2$ (in $\mathbb{F}_{3^{6n}}$)
4-b. $R_0 \leftarrow R_0^3$ (in $\mathbb{F}_{3^{6n}}$)
4-c. $R_0 \leftarrow R_0 R_1$ (in $\mathbb{F}_{3^{6n}}$)
5-a. $y_q \leftarrow -y_q$ (in $\mathbb{F}_{3^n}$)
5-b. $d \leftarrow ((d-b) \bmod 3)$
6. **end for**
7. **return** R_0

$\mathbb{F}_{3^n}/\mathbb{F}_3$ の基底は n の値によって異なり，存在するならば 3 項式基底が選ばれることが多い．ステップ 4-a を効率的に計算するために distortion map の定義で用いた σ,ρ を使って，$\mathbb{F}_{3^{6n}}/\mathbb{F}_{3^n}$ の基底を $\{1,\sigma,\rho,\sigma\rho,\rho^2,\sigma\rho^2\}$ とする．σ と ρ は $\sigma^2=-1$, $\rho^3=\rho+b$ ($b\in\{-1,1\}$) を満たす．$\mathbb{F}_{3^{6n}}$ の元 $a=a_0+a_1\sigma+a_2\rho+a_3\sigma\rho+a_4\rho^2+a_5\sigma\rho^2, a_i\in\mathbb{F}_{3^n}$ を簡単に $(a_0,a_1,a_2,a_3,a_4,a_5)$ と書くことにする．Algorithm 1 に必要な演算は，$\mathbb{F}_{3^n}$ での加減算，乗算，3 乗算，$\mathbb{F}_{3^{6n}}$ での加減算，乗算，3 乗算である．(9 乗算は 3 乗算を 2 回行えば良い．) $\mathbb{F}_{3^n}$ でも $\mathbb{F}_{3^{6n}}$ でも加減算はコストが低いため，Algorithm 1 のコストを $\mathbb{F}_{3^n}$ での乗算と 3 乗算の回数で評価する．M は $\mathbb{F}_{3^n}$ 乗算のコストを，C は $\mathbb{F}_{3^n}$ での 3 乗算のコストを表すとする．なお，$\mathbb{F}_{3^{6n}}$ 乗算は，Karatsuba 法を使うことにより 18 回の $\mathbb{F}_{3^n}$ での乗算 (と加減算) で計算でき [11]，$\mathbb{F}_{3^{6n}}$ での 3 乗算は 6 回の $\mathbb{F}_{3^n}$ での 3 乗算 (と加減算) で計算できる．Algorithm 1 の計算コストは表 1 のようになる．

次に [2] の Deffinition 1 で定義される η_T ペアリングを求めるアルゴリズムを紹介する．標数 3 の体では，η_T ペアリングはこれまでと同様に楕円曲線 $E^b: y^2=x^3-x+b$ 上で定義されるペアリングである．Algorithm 1 はループの回数が n 回であったが，η_T ペアリングのアルゴリズムのループの回数は $(n+1)/2$ 回だけである．T と Z,W を

表 1 Algorithm 1 の計算コスト

step	operations	executions
1	x_q^3, y_q^3	2 cubings in $\mathbb{F}_{3^n}$ ($2C$)
3-a	x_p^9, y_p^9	4 cubings in $\mathbb{F}_{3^n}$ ($4C$)
4-a	r_0^2, $y_p y_q$	2 multiplications in $\mathbb{F}_{3^n}$ ($2M$)
4-b	R_0^3	1 cubing in $\mathbb{F}_{3^{6n}}$ ($6C$)
4-c	$R_0 R_1$	1 multiplication in $\mathbb{F}_{3^{6n}}$ ($18M$)
	total	$20nM+(10n+2)C$

$$\begin{cases} T &= -b3^{(n+1)/2}-1,\ Z=-b3^{(n+3)/2}, \\ W &= (3^{3n}+1)(3^n-1)(3^n-b3^{(n+1)/2}+1) \\ &(= (3^{6n}-1)/\#E^b(\mathbb{F}_{3^n})) \end{cases} \quad (2)$$

と定義する．$\eta_T(P,Q)$ は双線形性を満たさないが，W 乗した $\eta_T(P,Q)^W$ は双線形性を満たす．$u_0=\eta_T(P,Q)$ とおくと，$u_1=u_0^{3^{3n}}\cdot u_0$, $u_2=u_1^{3^n}/u_1$, $\eta_T(P,Q)^W=u_2^{3^n}\cdot u_2\cdot(u_2^{3^{(n+1)/2}})^{-b}$ により $\eta_T(P,Q)^W$ を高速に計算できる．また，

$$(\eta_T(P,Q)^W)^{3T^2}=\hat{e}(P,Q)^Z \quad (3)$$

という関係が成り立つ [2]．$\eta_T(P,Q)^W$ から $\hat{e}(P,Q)$ を求める方法が [2] で説明されているが，ここではより効率的な方法を与える．関係式 (3) の両辺を $-b3^{(n-3)/2}$ 乗し，$v=\eta_T(P,Q)^W$ とおくと，関係式 $\hat{e}(P,Q)^{3^n}=v^{3^n(-2-b3^{(n+1)/2})-b3^{(n-1)/2}}$ が得られる．従って，関係式

$$\hat{e}(P,Q)=u^{-2}\cdot\left(u^{3^{(n+1)/2}}\cdot\sqrt[3^n]{u^{3^{(n-1)/2}}}\right)^{-b}$$

が得られる．なお，付録の §1 で示すように，$\mathbb{F}_{3^{6n}}$ での 3^n 乗根は数回の $\mathbb{F}_{3^n}$ の加減算だけで求まる．

$\eta_T(P,Q)$ を求めるアルゴリズムは n の条件によって変わる．ここでは $n\equiv 1 \pmod{12}$ に対するアルゴリズムを与える．

Algorithm 2: Computation of $\eta_T(P,Q)$ [2]

input: $P=(x_p,y_p), Q=(x_q,y_q)\in E^b(\mathbb{F}_{3^n})[l]$
output: $\eta_T(P,Q)$

1-a. **if** $b=1$ **then** $y_p \leftarrow -y_p$
1-b. $R_0 \leftarrow -y_p(x_p+x_q+b)+y_q\sigma+y_p\rho$
2. **for** $i\leftarrow 0$ **to** $(n-1)/2$ **do**
3. $r_0 \leftarrow x_p+x_q+b$ (in $\mathbb{F}_{3^n}$)
4-a. $R_1 \leftarrow -r_0^2+y_p y_q\sigma-r_0\rho-\rho^2$ (in $\mathbb{F}_{3^{6n}}$)
4-b. $R_0 \leftarrow R_0 R_1$ (in $\mathbb{F}_{3^{6n}}$)
5-a. $x_p \leftarrow x_p^{1/3}$, $y_p \leftarrow y_p^{1/3}$ (in $\mathbb{F}_{3^n}$)
5-b. $x_q \leftarrow x_q^3$, $y_q \leftarrow y_q^3$ (in $\mathbb{F}_{3^n}$)
6. **end for**
7. **return** R_0

Algorithm 2 の計算量は表 2 のようになる．R は $\mathbb{F}_{3^n}$ での 3 乗根に必要な計算量を表している．$\mathbb{F}_{3^n}$ での 3 乗根については [3] や付録の §2 を参照．$\mathbb{F}_{3^n}$ での 3 乗根の計算量は $\mathbb{F}_{3^n}$ での乗算の 2 倍以下である．

表 2 Algorithm 2 の計算量

step	operations	executions
1-b	$-y_p(x_p+x_q+b)$	1 multiplication in $\mathbb{F}_{3^n}$ ($1M$)
4-a	$y_p y_q$, r_0^2	2 multiplications in $\mathbb{F}_{3^n}$ ($2M$)
4-b	$R_0 R_1$	1 multiplication in $\mathbb{F}_{3^{6n}}$ ($18M$)
5-a	$x_p^{1/3}$, $y_p^{1/3}$	2 cube-roots in $\mathbb{F}_{3^n}$ ($2R$)
5-b	x_q^3, y_q^3	2 cubings in $\mathbb{F}_{3^n}$ ($2C$)
	total	$(10n+11)M+(n+1)C+(n+1)R$

表 3 §3 のアルゴリズムの比較

Algorithm	出力	計算量	計算量の概算
Alg.2 [2]	$\eta_T(P,Q)$	$(10n+11)M+(n+1)C+(n+1)R$	$(12.05n+13.05)M$
New Alg.3	$\eta_T(P,Q)^{3^{(n+1)/2}}$	$(10n+11)M+(5n+5)C$	$(10.25n+11.25)M$
New Alg.4	$\eta_T(P,Q)^{3^n}$	$(10n+11)M+(10n+4)C$	$(10.5n+11.2)M$
Alg.1 [12]	$\langle P,\psi(Q)\rangle_{3^{3n}+1}$	$20nM+(10n+2)C$	$(20.5n+0.1)M$

M は $\mathbb{F}_3^n$ での乗算の計算コスト，C は $\mathbb{F}_3^n$ での 3 乗算の計算コスト，R は $\mathbb{F}_3^n$ での 3 乗根の計算コスト．
双線形性ペアリングの値を得るには，アルゴリズムの値の最終べきを計算する必要がある．

3.1 提案方式

Algorithm 2 から 3 乗根が取り除かれたアルゴリズムを 2 つ提案する．一つ (Algorithm 3) は $\eta_T(P,Q)^{3^{(n+1)/2}}$ を出力し，もう一つ (Algorithm 4) は $\eta_T(P,Q)^{3^n}$ を出力する．

注意 Algorithm 3 や 4 の出力から $\eta_T(P,Q)$ の値を求めるためには $\mathbb{F}_{3^{6n}}$ の元の $3^{(n+1)/2}$ 乗根や 3^n 乗根を計算しなければならない．付録の §1 で示すように 3^n 乗根は簡単に計算できるため，Algorithm 4 を使って $\eta_T(P,Q)$ を計算できる．また，$\eta_T(P,Q)^W$ の双線形性より $(\eta_T(P,Q)^W)^{3^n} = (\eta_T(3^{(n-1)/2}P,Q)^{3^{(n+1)/2}})^W$ となるため，$3^{(n-1)/2}P$ の計算により Algorithm 3 を使って $\eta_T(P,Q)^W$ を求めることができる．なお $(x,y)\in E^b(\mathbb{F}_{3^n})$ に対して $3(x,y)=(x^9-b,-y^9)$ となり，$3^{(n-1)/2}P$ は $2(n-1)$ 回の 3 乗算で計算できる．

Algorithm 3 : Proposed computation of $\eta_T(P,Q)^{3^{(n+1)/2}}$

input: $P=(x_p,y_p), Q=(x_q,y_q)\in E^b(\mathbb{F}_{3^n})[l]$
output: $\eta_T(P,Q)^{3^{(n+1)/2}}$

1-a. **if** $b=1$ **then** $y_p \leftarrow -y_p$
1-b. $d \leftarrow b$ (in $\mathbb{F}_{3^n}$)
1-c. $R_0 \leftarrow -y_p(x_p+x_q+b)+y_q\sigma+y_p\rho$ (in $\mathbb{F}_{3^{6n}}$)
2. **for** $i \leftarrow 0$ **to** $(n-1)/2$ **do**
3. $r_0 \leftarrow x_p+x_q+d$ (in $\mathbb{F}_{3^n}$)
4-a. $R_1 \leftarrow -r_0^2+y_py_q\sigma-r_0\rho-\rho^2$ (in $\mathbb{F}_{3^{6n}}$)
4-b. $R_0 \leftarrow R_0R_1$ (in $\mathbb{F}_{3^{6n}}$)
5-a. $y_p \leftarrow -y_p$ (in $\mathbb{F}_{3^n}$)
5-b. $x_q \leftarrow x_q^9$, $y_q \leftarrow y_q^9$ (in $\mathbb{F}_{3^n}$)
5-c. $d \leftarrow ((d-b) \bmod 3)$
5-d. $R_0 \leftarrow R_0^3$ (in $\mathbb{F}_{3^{6n}}$)
6. **end for**
7. **return** R_0

Algorithm 3 の i 番目のループの終わりでの各値が Algorithm 2 の各値の 3^i 乗となるように調整している．そうすることで Algorithm 3 から 3 乗根の計算を取り除くことができ，その出力は $\eta_T(P,Q)^{3^{(n+1)/2}}$ となる．詳しくは付録の §3 を参照．Algorithm 3 のコストは表 3 のようになる．なお，双線形ペアリングを得るには W による最終べきも必要である．

表 4 Algorithm 3 の計算量

step	operations	executions
1-c	$-y_p(x_p+x_q+b)$	1 multiplication in $\mathbb{F}_{3^n}$ $(1M)$
4-a	y_py_q, r_0^2	2 multiplications in $\mathbb{F}_{3^n}$ $(2M)$
4-b	R_0R_1	1 multiplication in $\mathbb{F}_{3^{6n}}$ $(18M)$
5-b	x_q^9, y_q^9	4 cubings in $\mathbb{F}_{3^n}$ $(4C)$
5-d	R_0^3	1 cubing in $\mathbb{F}_{3^{6n}}$ $(6C)$
	total	$(10n+11)M+(5n+5)C$

Algorithm 4: Proposed computation of $\eta_T(P,Q)^{3^n}$

input: $P=(x_p,y_p)$, $Q=(x_q,y_q)\in E^b(\mathbb{F}_{3^n})[l]$
output: $\eta_T(P,Q)^{3^n}$

1-a. **if** $b=1$ **then** $y_p \leftarrow -y_p$
1-b. $d \leftarrow b$ in $\mathbb{F}_{3^n}$
1-c. $R_0 \leftarrow -y_p(x_p+x_q+b)+y_q\sigma+y_p\rho$
2. **for** $i \leftarrow 0$ **to** $(n-1)/2$ **do**
3. $r_0 \leftarrow x_p+x_q+d$ (in $\mathbb{F}_{3^n}$)
4-a. $R_1 \leftarrow -r_0^2+y_py_q\sigma-r_0\rho-\rho^2$ (in $\mathbb{F}_{3^n}$)
4-b. $R_0 \leftarrow R_0R_1$ (in $\mathbb{F}_{3^{6n}}$)
5-a. $x_p \leftarrow x_p^3$, $y_p \leftarrow y_p^3$ (in $\mathbb{F}_{3^n}$)
5-b. $x_q \leftarrow x_q^{27}$, $y_q \leftarrow y_q^{27}$ (in $\mathbb{F}_{3^n}$)
5-c. $d \leftarrow ((d+b) \bmod 3)$
5-d. $R_0 \leftarrow R_0^9$ if $i<(n-1)/2$
5-e. $R_0 \leftarrow R_0^3$ if $i=(n-1)/2$ (in $\mathbb{F}_{3^{6n}}$)
6. **end for**
7. **return** R_0

Algorithm 4 の i 番目のループの終わりでの各値は，Algorithm 2 の各値の 3^{2i} 乗 (最後のループでは 3^{2i-1} 乗) となっている．そのため Algorithm 3 は $\eta_T(P,Q)^{3^n}$ を出力する．Algorithm 4 の 5-c は基底の 3^i 乗の計算のために必要である．Algorithm 4 の計算量は表 5 のようになる．

表 5 Algorithm 4 の計算量

step	operations	executions
1-c	$-y_p(x_p+x_q+b)$	1 multiplication in $\mathbb{F}_{3^n}$ $(1M)$
4-a	y_py_q, r_0^2	2 multiplications in $\mathbb{F}_{3^n}$ $(2M)$
4-b	R_0R_1	1 multiplication in $\mathbb{F}_{3^{6n}}$ $(18M)$
5-a	x_p^3, y_p^3	2 cubings in $\mathbb{F}_{3^n}$ $(2C)$
5-b	x_q^{27}, y_q^{27}	6 cubings in $\mathbb{F}_{3^n}$ $(6C)$
5-d	R_0^9	2 cubings in $\mathbb{F}_{3^{6n}}$ $(12C)$
5-e	R_0^3	1 cubing in $\mathbb{F}_{3^{6n}}$ $(6C)$
	total	$(10n+11)M+(10n+4)C$

Algorithm 3 の証明は付録の §3 で行っている．Algorithm 4 の証明は Algorithm 3 とほぼ同じである．

§3 では Tate ペアリングや η_T ペアリングを求めるアルゴリズムを紹介し，3 乗根を必要としない η_T ペアリングのアルゴリズムを提案した．しかしこれらのアルゴリズムは 3 乗算が増えている．$\eta_T(P,Q)$ の値そのものの値は必要とせず非退化双線形ペアリングを必要とするアプリケーションには Algorithm 3 を用いることができる．3^n 乗根は $\mathbb{F}_{3^n}$ での加減算だけで計算できるため，Algorithm 4 は $\eta_T(P,Q)$ の値を必要とするアプリケーション対しても Algorithm 2 の代わりとなる．$C<1/(4R)$ ならば Algorithm 3 は Algorithm 1 より効

率的である．$C < (n+1)R/(9n+3)$ ならば Algorithm 4 は Algorithm 1 より効率的である．

§3 で紹介したアルゴリズムと本稿で提案したアルゴリズムの比較は表 3 のようになる．$\mathbb{F}_{3^n}$ での 3 乗根の計算は $\mathbb{F}_{3^n}$ での乗算の 2 倍以下のコストが必要であり，3 乗計算のコストは乗算に比べてかなり小さい ([3], 付録の §2)．表 3 の計算量の概算では $R = 2M$, $C = 0.05M$ とおいた．この時，Algorithm 3 の計算量は Algorithm 2 の計算量の約 $10.25nM/12.05nM \approx 0.85$ 倍である．但し，基底の取り方などにより $R < M$ となる場合もあり，R が小さいときは従来のアルゴリズムと本稿で提案したアルゴリズムとの計算量の差は小さくなる．

4. 高速検証方式

本節では Diffie–Hellman ペアの効率的な検証法を提案する．§3 と同様に M を $\mathbb{F}_{3^n}$ での乗算の計算量，C を $\mathbb{F}_{3^n}$ での 3 乗算の計算量，R を $\mathbb{F}_{3^n}$ での 3 乗根の計算量を表すとし，更に F_m は最終べきの m 乗の計算量を表すとする．

4.1 Diffie-Hellman ペアの検証

2 組の点のペア $(P,Q), (R,S)$ が $\hat{e}(P,Q) = \hat{e}(R,S)$ を満たすとき，(P,Q) と (R,S) は Diffie–Hellman ペアであるという．例えば，short signature の署名検証で Diffie-Hellman ペアの検証を行う．Diffie–Hellman ペアであるかどうか調べるには 2 つのペアリングの値を求める必要がある．よって Diffie–Hellman ペアの検証に必要な計算量は表 1 より $40nM+(20n+4)+2F_{3^{3n}-1}$ である．

関係式 (2) のように定義された T, Z に対して $gcd(3T^2, Z) = 1$ であることと関係式 (3) より

$$\hat{e}(P,Q) = \hat{e}(R,S) \Leftrightarrow \eta_T(P,Q)^W = \eta_T(R,S)^W$$

が成り立つ．ここで，W は関係式 (2) で定義される値である．従って Diffie–Hellman ペアの検証に η_T ペアリングを用いることができ，このときの検証の計算量は，Algorithm 2 を使うと $(20n+22)M + (2n+2)C + (2n+2)R + 2F_W$，Algorithm 3 を使うと $(20n+22)M + (10n+10)C + 2F_W$ となる．

4.2 最終べきの共通化による高速化

ペアリングの計算は，§3 で説明したアルゴリズムを用いて $\langle P, \psi(Q)\rangle_N$ や $\eta_T(P,Q)$ などを計算するステップと，最終べきを計算するステップに分けられる．$\hat{e}(P,Q)$ の双線形性より $\hat{e}(-P,Q) = \hat{e}(P,Q)^{-1}$ が成り立つから，

$$\begin{aligned}
&\hat{e}(P,Q) = \hat{e}(R,S)\\
&\quad\Leftrightarrow \hat{e}(P,Q)\cdot\hat{e}(-R,S) = 1\\
&\quad\Leftrightarrow \langle P,\psi(Q)\rangle_N^{(q^k-1)/N}\cdot\langle -R,\psi(S)\rangle_N^{(q^k-1)/N} = 1\\
&\qquad\qquad\qquad\qquad (\hat{e}\text{の定義より})\\
&\quad\Leftrightarrow (\langle P,Q\rangle_N\cdot\langle -R,S\rangle_N)^{(q^k-1)/N} = 1
\end{aligned}$$

が成り立つ．η_T ペアリングの場合も同様である．従って，Diffie–Hellman ペアの検証は 2 回のペアリングの計算，1 回の最終べき，1 回の $\mathbb{F}_{3^{6n}}$ での乗算で行うことができる．よって，Diffie–Hellman ペアの検証に必要な計算量は，Tate ペアリングを用いるときは $(40n+18)M + (20n+4) + F_{3^{3n}+1}$，$\eta_T$ ペアリングを用いるときは，Algorithm 2 を使うと $(20n+40)M + (2n+2)C + (2n+2)R + F_W$，Algorithm 3 を使うと $(20n+40)M + (10n+10)C + F_W$ となる．

4.3 最終べきの省略

用いるペアリングのアルゴリズムによっては，Diffie–Hellman ペアの検証において指数計算を省略できる場合がある．Algorithm 1 を使うと，$\langle P, \psi(Q)\rangle_{3^{3n}+1}$ $(N = 3^{3n}+1)$ が出力される．最終べきの指数の値は $(3^{6n}-1)/N = (3^{6n}-1)/(3^{3n}+1) = 3^{3n}-1$ である．$3^{3n}-1 = \#\mathbb{F}_{3^{3n}}^*$ であることに注意．有限体の性質より，$x \in \mathbb{F}_{3^{6n}}$ に対して

$$x^{3^{3n}-1} = 1 \Leftrightarrow x \in \mathbb{F}_{3^{3n}}^*$$

が成り立つ．従って，

$$\begin{aligned}
\hat{e}(P,Q) = \hat{e}(R,S) &\Leftrightarrow (\langle P,Q\rangle_{3^{3n}+1}\cdot\langle -R,S\rangle_{3^{3n}+1})^{3^{3n}-1}\\
&\Leftrightarrow (\langle P,Q\rangle_{3^{3n}+1}\cdot\langle -R,S\rangle_{3^{3n}+1}) \in \mathbb{F}_{3^{3n}}^*
\end{aligned}$$

が成り立つ．よって，(P,Q), (R,S) の Diffie-Hellman ペアの検証には，$\langle P,Q\rangle_{3^{3n}+1}\cdot\langle -R,S\rangle_{3^{3n}+1}$ を計算し，この値が $\mathbb{F}_{3^{3n}}^*$ の元であるかどうかを調べればよく，最終べきを必要としない．なお，$\mathbb{F}_{3^{6n}}/\mathbb{F}_{3^n}$ の基底が $\{1, \sigma, \rho, \sigma\rho, \rho^2, \sigma\rho^2\}$ のときは，$x \in \mathbb{F}_{3^{6n}}$ に対して

$$x \in \mathbb{F}_{3^{3n}} \Leftrightarrow \sigma,\ \sigma\rho,\ \sigma\rho^2\text{の係数が } 0$$

となるため，$\mathbb{F}_{3^{3n}}$ の元であるかどうかは簡単に調べられる．この時の Diffie–Hellman ペアの検証の計算量は $(40n+40)M + (10n+5)C$ となる．

4.4 提案方式

Diffie–Hellman ペアの検証において最終べきを 1 回にできることと，場合によっては不要であることを説明した．更に Diffie–Hellman ペアの検証を高速にするには，$\langle P, \psi(Q)\rangle_N \cdot \langle -R, \psi(S)\rangle_N$ や $\eta_T(P,Q)\cdot\eta_T(-R,S)$ の計算の高速化が必要である．ここでは 2 つのペアリングの積を求めるアルゴリズムを考えることで，Diffie–Hellman ペアの検証の高速アルゴリズムを提案する．

Algorithm 5: Proposed DH-Verification based on Alg. 1

input: $P_0 = (x_{p0}, y_{p0}), Q_0 = (x_{q0}, y_{q0})$,
$P_1 = (x_{p1}, y_{p1}), Q_1 = (x_{q1}, y_{q1}) \in E^b(\mathbb{F}_{3^n})[l]$
output: $\langle P_0, \psi(Q_0)\rangle_{3^{3n}+1}\cdot\langle P_1, \psi(Q_1)\rangle_{3^{3n}+1}$

1-a. $R_0 \leftarrow 1$ (in $\mathbb{F}_{3^{6n}}$)
1-b. $x_{q0} \leftarrow x_{q0}^3$, $y_{q0} \leftarrow y_{q0}^3$, $x \leftarrow x_{q0}^3$, $y \leftarrow y_{q0}^3$ (in $\mathbb{F}_{3^n}$)
1-c. $d \leftarrow bn \bmod 3$
2. **for** $i \leftarrow 0$ **to** $n-1$ **do**
3-a. $x_{p0} \leftarrow x_{p0}^9$, $y_{p0} \leftarrow y_{p0}^9$, $x_{p1} \leftarrow x_{p1}^9$, $y_{p1} \leftarrow y_{p1}^9$ (in $\mathbb{F}_{3^n}$)
3-b. $(\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$
$\leftarrow S(x_{p0}, y_{p0}, x_{q0}, y_{q0}, x_{p1}, y_{p1}, x_{q1}, y_{q1}, d)$
3-c. $R_1 \leftarrow (\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$
4-a. $R_0 \leftarrow R_0^3$ (in $\mathbb{F}_{3^{6n}}$)
4-b. $R_0 \leftarrow R_0 R_1$ (in $\mathbb{F}_{3^{6n}}$)
5-a. $y_{q0} \leftarrow -y_{q0}$, $y_{q0} \leftarrow -y_{q1}$ (in $\mathbb{F}_{3^n}$)
5-b. $d \leftarrow ((d-b) \bmod 3)$
6. **end for**
7. **return** R_0

ここで S は次のようなサブアルゴリズムである.

DH-Verification Subalgorithm S:

input: $x_{p0}, y_{p0}, x_{q0}, y_{q0}, x_{p1}, y_{p1}, x_{q1}, y_{q1} \in \mathbf{F}_{3^n}$, $d \in \mathbf{F}_3$
output: $(\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5) \in \mathbf{F}_{3^{6n}}$

01. $r_0 \leftarrow x_{p0} + x_{q0} + d$, $r_1 \leftarrow x_{p1} + x_{q1} + d$ (in $\mathbf{F}_{3^n}$)
02. $r_2 \leftarrow y_{p0}y_{q0}$, $r_3 \leftarrow y_{p1}y_{q1}$ (in $\mathbf{F}_{3^n}$)
03. $r_4 \leftarrow r_0 r_1$ (in $\mathbf{F}_{3^n}$)
04. $r_5 \leftarrow r_2 r_3$ (in $\mathbf{F}_{3^n}$, $y_{p0}y_{p1}y_{q0}y_{q1}$)
05. $r_6 \leftarrow r_2 + r_0$, $r_7 = r_3 + r_1$
(in $\mathbf{F}_{3^n}$, $r_0 + y_{p0}y_{q0}$, $r_1 + y_{p1}y_{q1}$)
06. $r_8 \leftarrow r_0 + r_1$ (in $\mathbf{F}_{3^n}$)
07. $\lambda_3 \leftarrow r_6 r_7 - r_4 - r_5$ (in $\mathbf{F}_{3^n}$, $r_0 y_{p1}y_{q1} + r_1 y_{p0}y_{q0}$)
08. $\lambda_5 \leftarrow r_2 + r_3$ (in $\mathbf{F}_{3^n}$, $y_{p0}y_{q0} + y_{p1}y_{q1}$)
09. $\lambda_0 \leftarrow r_4^2 - r_5 + br_8$
(in $\mathbf{F}_{3^n}$, $r_0^2 r_1^2 - y_{p0}y_{p1}y_{q0}y_{q1} + r_0 + r_1$)
10. $\lambda_1 \leftarrow r_8\lambda_3 - r_4\lambda_5$ (in $\mathbf{F}_{3^n}$, $r_0^2 y_{p1}y_{q1} + r_1^2 y_{p0}y_{q0}$)
11. $\lambda_2 \leftarrow (r_4 + 1)r_8 + b$
(in $\mathbf{F}_{3^n}$, $r_0^2 r_1 + r_0 r_1^2 + r_0 + r_1 + 1$)
12. $\lambda_4 \leftarrow r_8^2 - r_4 + 1$ (in $\mathbf{F}_{3^n}$, $r_0^2 + r_0 r_1 + r_1^2 + 1$)
13. **return** $(\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$

$b \in \{-1, 1\}$ は楕円曲線の定義式 $y^2 = x^3 - x + b$ によって定まる. Subalgorithm S の出力 $(\lambda_0, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)$ は

$$\left.\begin{array}{rcl} \lambda_0 &=& r_0^2 r_1^2 - y_{p0}y_{p1}y_{q0}y_{q1} + br_0 + br_1 \\ \lambda_1 &=& r_0^2 y_{p1}y_{q1} + r_1^2 y_{p0}y_{q0} \\ \lambda_2 &=& r_0^2 r_1 + r_0 r_1^2 + r_0 + r_1 + b \\ \lambda_3 &=& r_0 y_{p1}y_{q1} + r_1 y_{p0}y_{q0} \\ \lambda_4 &=& r_0^2 + r_0 r_1 + r_1^2 + 1 \\ \lambda_5 &=& y_{p0}y_{q0} + y_{p1}y_{q1} \end{array}\right\} \quad (4)$$

となっている. Algorithm 5 は, 拡大体の構造を利用して関係式 (4) を効率的に計算することにとり, 高速性を達成している.

次に Algorithm 5 の正当性を示す. Algorithm 1 に P_m, Q_m $(m = 0, 1)$ を入力したときの各値を, 右に (m) を付けて表すことにする. 例えば, Algorithm 1 に P_0, Q_0 を入力したときの R_0 は $R_0(0)$ と表される. 各 i に対して Algorithm 1 と Algorithm 5 のループの最後で

$$R_0 = R_0(0)R_0(1) \quad (5)$$

が成り立っていれば, Algorithm 5 は正しく動作する. 関係式 (5) は以下のようにして示される.

$R_1(m) = (-r_0(m)^2, -y_p(m)y_q(m), r_0(m), 0, -1, 0)$, $m = 0, 1$ であるから, 計算により

$$\begin{aligned} &R_1(0)R_1(1) = \\ &(r_0(0)^2 r_0(1)^2 + y_p(0)y_p(1)y_q(0)y_q(1) + br_0(0) + br_0(1), \\ &\ r_0(0)^2 y_p(1)y_q(1) + r_0(1)^2 y_p(0)y_q(0), \\ &\ r_0(0)^2 r_0(1) + r_0(0)r_0(1)^2 + r_0(0) + r_0(1) + b, \\ &\ r_0(0)y_p(0)r_0(1) + r_0(1)y_p(1)y_q(1), \\ &\ r_0(0)^2 + r_0(0)r_0(1) + r_0(1)^2 + 1, \\ &\ y_p(0)y_q(0) + y_p(1)y_q(1)) \end{aligned}$$

となることが分かる. Algorithm 5 では r_0 が $r_0(0)$ に, r_1 が $r_0(1)$ に, y_{pm} が $y_p(m)$ に, y_{qm} が $y_q(m)$ に対応している. 従って Algorithm 5 の 3-b より

$$R_1 = R_1(0)R_1(1) \quad (6)$$

となる.

Algorithm 1 の 4-c が行われる前の $R_0(m)$ を $R_0(m)'$, 4-c が行われた後の $R_0(m)$ を $R_0(m)''$ とする. 同様に Algorithm 5 の 4-b が行われる前の R_0 を R_0', 4-b が行われた後の R_0 を R_0'' とする. 関係式 (5) を証明するには, すべての i に対して

$$R_0'' = R_0(0)''R_0(1)'' \quad (7)$$

となることを示せば十分である.

$i = 0$ のときは $R_0(0)' = R_0(1)' = R_0' = 1$ と関係式 (6) より関係式 (7) は成り立っている.

$i = j$ で関係式 (7) が成り立つと仮定する. すると, $i = j + 1$ のときでは

$$R_0' = R_0(0)'R_0(1)' \quad (8)$$

が成り立っている. Algorithm 5 の 4-b では

$$R_0'' \leftarrow R_1'R_0' \quad (9)$$

が計算され, Algorithm 1 の 4-c では

$$R_0(m)'' \leftarrow R_1(m)'R_0(m)' \quad (10)$$

が行われる. よって

$$\begin{aligned} &R_0(0)''R_0(1)'' \\ &= R_1(0)'R_1(1)'R_0(0)'R_0(1)' \quad (10) \text{ より} \\ &= R_1'R_0' \quad (6), (8) \text{ より} \\ &= R_0'' \quad (9) \text{ より} \end{aligned}$$

となり, 関係式 (7) が示された. よって Algorithm 5 の正当性が示された.

Algorithm 5 の計算量は表 6 のようになる.

表 6 Algorithm 5 の計算量

step	operations	executions
1-b	x_{q0}^3, y_{q0}^3, x_{q1}^3, y_{q1}^3	4 cubings in $\mathbf{F}_{3^n}$ $(4C)$
3-a	x_{p0}^9, y_{p0}^9, x_{p1}^9, y_{p1}^9	8 cubings in $\mathbf{F}_{3^n}$ $(8C)$
3-b	Subalgorithm $S1$	10 multiplications in $\mathbf{F}_{3^n}$ $(10M)$
4-b	R_0^3	1 cubing in $\mathbf{F}_{3^{6n}}$ $(6C)$
4-c	R_0R_1	1 multiplication in $\mathbf{F}_{3^{6n}}$ $(18M)$
	total	$28nM + (14n + 4)C$

Tate ペアリングでの Diffie-Hellman ペアでは, Algorithm 5 を一度用いるだけで検証を行うことができる. よって検証の計算量は $28nM + (14n + 4)C$ となる.

同様に Algorithm 2,3,4 に対しても Algorithm 5 のような Diffie-Hellman ペアの検証アルゴリズムを導くことができる. ここでは Algorithm 3 の改良による Diffie-Hellman ペアの検証アルゴリズムを与える.

表 7 Tate ペアリングを用いる Diffie–Hellman ペア検証の計算量

Alg.1 [12], 従来の方法 (§4.1)	Alg.1 [12], 指数計算削除 (§4.3)	New Alg.5
$(40n+18)M+(20n+4)C+2F_{3^{3n}-1}$	$(40n+18)M+(20n+4)C$	$28nM+(14n+4C)$
$((41n+18.2)M+2F_{3^{3n}-1})$	$((41n+18.2)M)$	$(28.7n+0.5M)$

表 8 η_T ペアリングを用いる Diffie–Hellman ペア検証の計算量

アルゴリズム	Alg.2 [2]	New Alg.3	New Alg.6
従来の方法	$(20n+22)M+(2n+2)C+(2n+2)R+2F_W$	$(20n+22)M+(10n+10)C+2F_W$	–
(§4.1)	$((24.1n+26.1)M+2F_W)$	$((20.5n+22.5)M+2F_W)$	–
指数計算の共通化	$(20n+40)M+(2n+2)C+(2n+2)R+F_W$	$(20n+40)M+(10n+10)C+F_W$	$(14n+34)M+(7n+7)C+F_W$
(§4.2)	$((24.1n+44.1)M+F_W)$	$((20.5n+40.5)M+F_W)$	$(14.35n+34.35)M+F_W)$

(下段) は計算量の概算，M は $\mathbf{F}_{3^n}$ での乗算の計算量，C は $\mathbf{F}_{3^n}$ での 3 乗算の計算量，F_m は最終べき (m 乗) の計算量．

Algorithm 6: Proposed DH-Verification based on Alg. 3

input: $P_0=(x_{p0},y_{p0}), Q_0=(x_{q0},y_{q0})$,
$P_1=(x_{p1},y_{p1}), Q_1=(x_{q1},y_{q1})\in E^b(\mathbf{F}_{3^n})[l]$
output: $(\eta_T(P_0,Q_0)\cdot\eta_T(P_1,Q_1))^{3^{(n+1)/2}}$

1-a. **if** $b=1$ **then** $y_{p0}\leftarrow -y_{p0}$, $y_{p1}\leftarrow -y_{p1}$
1-b. $d\leftarrow b$ (in $\mathbf{F}_{3^n}$)
1-c. $R_0\leftarrow y_{p0}(x_{p0}+x_{q0}+b)+y_{q0}\sigma+y_{p0}\rho$,
$R_1\leftarrow y_{p1}(x_{p1}+x_{q1}+b)+y_{q1}\sigma+y_{p1}\rho$ (in $\mathbf{F}_{3^n}$)
1-d. $R_0\leftarrow R_0R_1$
2. **for** $i\leftarrow 0$ **to** $(n-1)/2$ **do**
3-a. $(\lambda_0,\lambda_1,\lambda_2,\lambda_3,\lambda_4,\lambda_5)$
$\leftarrow S(x_{p0},y_{p0},x_{q0},y_{q0},x_{p1},y_{p1},x_{q1},y_{q1},d)$
3-b. $R_1\leftarrow(\lambda_0,\lambda_1,\lambda_2,-\lambda_3,\lambda_4,-\lambda_5)$
4. $R_0\leftarrow R_0R_1$ (in $\mathbf{F}_{3^{6n}}$)
5-a. $x_{q0}\leftarrow x_{q0}^9$, $y_{q0}\leftarrow y_{q0}^9$, $x_{q1}\leftarrow x_{q1}^9$, $y_{q1}\leftarrow y_{q1}^9$,
5-b. $d\leftarrow((d-b)\bmod 3)$
5-c. $R_0\leftarrow R_0^3$ (in $\mathbf{F}_{3^{6n}}$)
6. **end for**
7. **return** R_0

Algorithm 6 の証明は Algorithm 5 と同様にできる．Algorithm 6 の計算量は表 6 のようになる．

表 9 Algorithm 6 の計算量

step	operations	executions
1-c	$y_{p0}(x_{p0}+x_{q0}+b)$, $y_{p0}(x_{p0}+x_{q0}+b)$	2 multiplications in $\mathbf{F}_{3^n}$ $(2M)$
1-d	R_0R_1	1 multiplication in $\mathbf{F}_{3^{6n}}$ $(18M)$
3	Subalgorithm S	10 multiplications in $\mathbf{F}_{3^n}$ $(10M)$
4	R_0R_1	1 multiplication in $\mathbf{F}_{3^{6n}}$ $(18M)$
5-a	x_{q0}^9, y_{q0}^9, x_{q1}^9, y_{q1}^9	8 cubings in $\mathbf{F}_{3^n}$ $(8C)$
5-c	R_0^3	1 cubing in $\mathbf{F}_{3^{6n}}$ $(6C)$
	total	$(14n+34)M+(7n+7)C$

Algorithm 6 を使うとき，Diffie–Hellman ペアの検証には Alorithm 6 を 1 回と $W=(3^{6n}-1)/(3^n+b3^{(n+1)/2}+1)$ 乗計算を 1 回行う必要がある．よって，このときの Diffie–Hellman ペアの検証の計算量は $(14n+34)M+(7n+7)C+F_W$ となる．

Diffie-Hellman ペアの検証の従来の方法と §3 で提案したアルゴリズムによる方法とを比較すると，表 7 や表 8 のようになる．表 2 と同様に計算量の概算では $C=0.05M$, $R=2M$ とおいた．Tate ペアリングを用いる場合は，提案したアルゴリズムを用いると署名検証の計算量が約 $28.7nM/41nM=0.70$ 倍になり，η_T ペアリングを用いる場合は，提案したアルゴリズムを用いると署名検証の最終べきを除く部分の計算量が約 $14.35nM/24.1nM=0.60$ 倍になる．場合によっては Diffie-Hellman ペアの検証において最終べきが不要になることを示した．

注意 Diffie-Hellman ペアのバッチ検証 [7], [10], [15] は複数のペアの検証を 1 回の検証プロセスで行う方法であり，従来は 2 回のペアリングの計算を必要とした．§4 の手法は 1 回の検証プロセスを更に高速化でき，計算量が 30〜40%削減される．また，ペアリングを用いたブロードキャスト暗号 [5] の復号化には 2 つのペアリングの積 $\hat{e}(P_1,P_2)\hat{e}(P_3,P_4)$ の計算を用いるが，これにも §4 の手法を用いて効率的に行うことができる．(ここで P_1,P_2,P_3,P_4 は任意の点とする．)

謝辞 本論文は，独立行政法人新エネルギー・産業技術総合開発機構の受託研究「Pairing Lite の研究開発」の一環として行われた．

文　献

[1] P. S. L. M. Barreto, "Efficient algorithms for pairing-based cryptosystems," *CRYPTO 2002*, LNCS 3027, pp.354-368, 2002.
[2] P. S. L. M. Barreto, S. Galbraith, C. Ó. hÉigeartaigh and M. Scott, "Efficient pairing computation on supersingular abelian varieties," Cryptology ePrint Archive, Report 2004/375, 2004.
[3] P. S. L. M. Barreto, H. Y. Kim, B. Lynn and M. Scott, "A note on efficient computation of cube roots in characteristic 3," Cryptology ePrint Archive, Report 2004/305, 2004.
[4] D. Boneh and M. Franklin, "Identity based encryption from the Weil pairing," *CRYPTO 2001*, LNCS 2139, pp.213-229, 2001.
[5] D. Boneh, C. Gentry and B. Waters, "Collusion resistant broadcast encryption with short ciphertexts and private keys," *CRYPTO 2005*, LNCS 3621, pp.258-275, 2005.
[6] D. Boneh, B. Lynn and H. Shacham, "Short signature from the Weil pairing," *ASIACRYPT 2001*, LNCS 2248, pp.514-532, 2001.
[7] J. H. Cheon, Y. Kim and H. J. Yoon, "A new ID-based signature with batch verification," Cryptology ePrint Archive, Report 2004/131, 2004.
[8] I. Duursma and H. S. Lee, "Tate pairing implementation for hyperelliptic curves $y^2=x^p-x+d$," *ASIACRYPT 2003*, LNCS 2894, pp.111-123, 2003.
[9] P. Grabher and D. Page, "Hardware acceleration of the Tate pairing in characteristic three," *CHES 2005*, LNCS 3659,

pp.398-411, 2005.
[10] F. Hoshino, K. Abe and T. Kobayashi, "Lenient/strict batch verification in several groups," *ISC 2001*, LNCS 2200, pp.81-94, 2001.
[11] T. Kerins, W. P. Marnane, E. M. Popovici and P. S. L. M. Barreto, "Efficient hardware for the Tate pairing calculation in characteristic three," *CHES 2005*, LNCS 3659, Springer-Verlag, pp.412-426, 2005.
[12] S. Kown, "Efficient Tate pairing computation for supersingular elliptic curves over binary fields," Cryptology ePrint Archive, Report 2004/303, 2004.
[13] V. Miller, "Short programs for functions on curves," Unpublished manuscript, 1986. available at `http://crypto.stanford.edu/miller/miller.pdf` .
[14] J. Silverman, *The arithmetic of elliptic curves*, Springer-Verlag, 1986.
[15] H. Yoon, J. H. Cheon and Y. Kim, "Batch verification with ID-based signatures," *ICISC 2004*, LNCS 3506, pp.233-248, 2005.

付　　録

1. 拡大体での 3^n 乗根の計算

$n \equiv 1 \pmod 6$ (つまり $\sigma^{3^n} = -\sigma$, $\rho^{3^n} = \rho + 1$, $(\rho^2)^{3^n} = \rho^2 - \rho + 1$) の場合の $\mathbf{F}_{3^{6n}}$ の元の 3^n 乗根の求め方を考える．$(a_0, a_1, a_2, a_3, a_4, a_5) \in \mathbf{F}_{3^{6n}}$ の 3^n 乗根を $(b_0, b_1, b_2, b_3, b_4, b_5) \in \mathbf{F}_{3^{6n}}$ とすると

$$
\begin{aligned}
&(b_0, b_1, b_2, b_3, b_4, b_5)^{3^n} \\
&= (b_0 + b_1\sigma + b_2\rho + b_3\sigma\rho + b_4\rho^2 + b_5\sigma\rho^2)^{3^n} \\
&= b_0^{3^n} + b_1^{3^n}\sigma^{3^n} + b_2^{3^n}\rho^{3^n} + b_3^{3^n}(\sigma\rho)^{3^n} + b_4^{3^n}(\rho^2)^{3^n} + b_5^{3^n}(\sigma\rho^2)^{3^n} \\
&= b_0 + b_1(-\sigma) + b_2(\rho+1) + b_3(-\sigma)(\rho+1) \\
&\qquad + b_4(\rho^2 - \rho + 1) + b_5(-\sigma)(\rho^2 - \rho + 1) \\
&= (b_0 + b_2 + b_4) + (-b_1 - b_3 - b_5)\sigma + (b_2 - b_4)\rho \\
&\qquad + (-b_3 + b_5)\sigma\rho + b_4\rho^2 - b_4\sigma\rho^2 \\
&= (b_0 + b_2 + b_4, -b_1 - b_3 - b_5, b_2 - b_4, -b_3 + b_5, b_4, -b_5)
\end{aligned}
$$

となる．これが $(a_0, a_1, a_2, a_3, a_4, a_5)$ と等しいため，連立一次方程式を解くことにより

$$
\begin{cases}
b_0 &= a_0 - a_2 + a_4 \\
b_1 &= -a_1 + a_3 - a_5 \\
b_2 &= a_2 + a_4 \\
b_3 &= -a_3 - a_5 \\
b_4 &= a_4 \\
b_5 &= -a_5
\end{cases}
$$

が得られる．つまり，$\mathbf{F}_{3^{6n}}$ の元の 3^n 乗根の計算には，$\mathbf{F}_{3^n}$ 乗算を必要としない．$n \equiv 0, 2, 3, 4, 5 \pmod 6$ の場合も同様である．

2. 3 乗根について

[3] にあるように，$n \not\equiv 0 \bmod 3$，$\mathbf{F}_{3^n}/\mathbf{F}_3$ が 3 項式基底を持つときは，$r = \sum_{i=0}^{n-1} r_i x_i \in \mathbf{F}_{3^{6n}}$ に対して

$$
\sqrt[3]{r} = \begin{cases}
\sum_{i=0}^{u} r_{3i}x^i + x^{1/3}\cdot\sum_{i=0}^{u-1} r_{3i+1}x^i + x^{2/3}\cdot\sum_{i=0}^{u-1} r_{3i+2}x^i & \text{if } n = 3u+1 \\
\sum_{i=0}^{u} r_{3i}x^i + x^{1/3}\cdot\sum_{i=0}^{u} r_{3i+1}x^i + x^{2/3}\cdot\sum_{i=0}^{u} r_{3i+2}x^i & \text{if } n = 3u+2
\end{cases}
$$

となる．よって，あらかじめ $x^{1/3}$ と $x^{2/3}$ を計算しておけば，$\sqrt[3]{r}$ は 2 回の乗算，$x^{1/3}\cdot\sum_{i=0}^{u-1} r_{3i+1}x^i$ と $x^{2/3}\cdot\sum_{i=0}^{u-1} r_{3i+2}x^i$ で計算できる．$x^{1/3}$ と $x^{2/3}$ は固定されているので，3 乗根の計算量は更に削減できる場合がある．

3. Algorithm 3 の証明

初めに ρ^{3^n} と σ^{3^n} を考える．$\mu^2 = -1, \rho^3 = \rho + 1$ の場合では，

$$
\sigma^{3^n} = \begin{cases} \sigma & n \equiv 0 \pmod 2 \\ -\sigma & n \equiv 1 \pmod 2 \end{cases} \tag{A·1}
$$

$$
\rho^{3^n} = \begin{cases} \rho & n \equiv 0 \pmod 3 \\ \rho + 1 & n \equiv 1 \pmod 3 \\ \rho - 1 & n \equiv 2 \pmod 3 \end{cases} \tag{A·2}
$$

$$
(\rho^2)^{3^n} = \begin{cases} \rho^2 & n \equiv 0 \pmod 3 \\ \rho^2 - \rho + 1 & n \equiv 1 \pmod 3 \\ \rho^2 + \rho + 1 & n \equiv 2 \pmod 3 \end{cases} \tag{A·3}
$$

となる．

Algorithm m の R_0 を $R_0[m]$ などと書くことにする．$i = j$ のとき Algorithm 2 の各ループの終わりでの $R_0[2]$ と Algorithm 3 の各ループの終わりでの $R_0[3]$ が

$$
R_0[3] = R_0[2]^{3^{j+1}} \tag{A·4}
$$

が成り立っていることを示せば十分である．$i = j$ のとき，両アルゴリズムの各ループの終わりでは

$$
\begin{aligned}
x_p[3] &= x_p[2]^{3^j},\ y_p[3] = y_p[2]^{3^j}, \\
x_q[3] &= x_q[2]^{3^j},\ y_q[3] = (-1)^j y_q[2]^{3^j},
\end{aligned} \tag{A·5}
$$

が成り立っていることは簡単に分かる．最初に

$$
R_1[3] = R_1[2]^{3^j} \tag{A·6}
$$

を示し，それからこの結果を使って関係式 (A·4) を示す．

$j \equiv 0 \pmod 6$ のとき，関係式 (A·1), (A·2), (A·3) より

$$
\sigma^{3^j} = \sigma,\ \rho^{3^j} = \rho,\ (\rho^2)^{3^j} = \rho^2,\ d = b \tag{A·7}
$$

である．Algorithm 2 と Algorithm 3 のステップ 3 より

$$
\begin{aligned}
r_0[2]^{3^j} &= (x_p[2] + x_q[2] + b)^{3^j} \\
&= x_p[2]^{3^j} + x_q[2]^{3^j} + b \\
&= x_p[3] + x_q[3] + b \qquad \text{(A·5) より} \\
&= r_0[3]
\end{aligned} \tag{A·8}
$$

となる．よって Algorithm 2 と Algorithm 3 のステップ 4-a より

$$
\begin{aligned}
R_1[2]^{3^j} &= (y_p[2]y_q[2]\sigma - r_0[2]^2 - r_0[2]\rho - \rho^2)^{3^j} \\
&= y_p[2]^{3^j} y_q[2]^{3^j} \sigma^{3^j} - (r_0[2]^2)^{3^j} - r_0[2]^{3^j}\rho^{3^j} - (\rho^2)^{3^j} \\
&= y_p[3]y_q[3]\sigma - r_0[3]^2 - r_0[3]\rho - (\rho^2) \quad \text{(A·7),(A·8) より} \\
&= R_1[3]
\end{aligned}
$$

となる．よって，$j \equiv 0 \pmod 6$ の時は，関係式 (A·6) が成り立っている．

$j \equiv 1, 2, 3, 4, 5 \pmod 6$ の) ときも同様に関係式 (A·6) が成り立つことを示すことができる．

この結果を使って，関係式 (A·4) を数学的帰納法によって示す．$i = 0$ のときは関係式 (A·4) は明らかに成り立つ．

$i = j$ で関係式 (A·4) が成り立っていると仮定する．すると更新前の $R_0[2]$, $R_0[3]$ に対して，更新後の $R_0[3]$ の値は

$$
(R_0[3]R_1[3])^3 = (R_0[2]^{3^{j+1}} R_1[2]^{3^{j+1}})^3 = (R_0[2]R_1[2])^{3^{j+2}}
$$

となる．$R_0[2]$ は更新後 $R_0[2]R_1[2]$ となるから，更新後では $R_0[3] = R_0[2]^{3^{j+2}}$ となる． □