推薦論文

Javaを利用した携帯電話上でのTateペアリングの高速実装

川原祐人[†1] 高木剛[†1] 岡本栄司[†2]

近年，新たな暗号システムとしてペアリング暗号システムが注目を集めている．現在，ペアリング暗号システムについて様々な実装が行われているが，それらはまだ従来の公開鍵暗号システムよりも処理速度が遅い状況にある．また，携帯電話上で Java を利用したペアリングを実装した例はまだ発表されていない．本稿ではペアリングの最新の高速クラスである η_T ペアリングを Java で実装し，携帯電話上でペアリングの速度評価を行う．標数 3，次数 $m = \{97, 167, 193, 239, 313\}$ の有限体 $\mathbf{F}_{3^m}$ 上の η_T ペアリングの実装・評価を行った．また，汎用的に利用できるプログラムとの比較のため，次数 $m = 97$ に特化し，有限体 $\mathbf{F}_{3^{97}}$ 上の η_T ペアリングの実装・評価を行った．Java がサポートされている NTT DoCoMo の携帯電話 7 機種で評価を行い，次数 $m = 97$ の有限体 $\mathbf{F}_{3^{97}}$ に特化した実装では 0.5 秒以下の速度を得た．

Efficient Implementation of Tate Pairing on Mobile Phones Using Java

Yuto Kawahara,[†1] Tsuyoshi Takagi[†1] and Eiji Okamoto[†2]

Pairing-based cryptosystems have been attracted by researchers in cryptography. Some implementations show that pairing-based cryptosystems are relatively slower than the standard public key cryptosystems. Then there is no paper is presented implementation of Tate pairing on mobile phones using Java. In this paper, we implemented the η_T pairing for computing Tate pairing on mobile phones using Java, and measured the computation speed of the η_T pairing on mobile phones. We implemented the η_T pairing over some finite fields of characteristic 3 with extension degree $m = \{97, 167, 193, 239, 313\}$, and optimized for extension degree $m = 97$ to compare general purpose program. We measured the computation speed using 7 mobile phones, NTT DoCoMo. Our optimized implementation for $m = 97$ achieved under 0.5 seconds for computing the η_T pairing.

1. はじめに

ペアリング暗号システムでは，ID ベース暗号[5] や short signature[7]，効率的なブロードキャスト暗号[6] などの新たな暗号システムが提唱されている．ID ベース暗号では，従来の公開鍵暗号と異なり，ユーザが既知な任意の情報を公開鍵として用いることが可能である．また，short signature では従来の楕円曲線を用いる署名と比べ署名長が約半分になる．効率的なブロードキャスト暗号では，秘密鍵と送信する暗号文を固定長にすることが可能である．これらのシステムは RSA 暗号や楕円曲線暗号などの従来暗号では構築することができなかった．

ペアリング暗号システムでは有限体上の楕円曲線の Tate ペアリングを用いる．Tate ペアリングの計算アルゴリズムは Miller により初めて提案され[17]，その処理速度は RSA 暗号などの従来暗号と比較すると 5 倍以上遅い[3]．近年，Duursma らにより楕円曲線を標数 3 の有限体 $\mathbf{F}_{3^m}$ 上の超特異曲線に限定したことで効率的なアルゴリズムが示され[8]，Kwon がこのアルゴリズムから 3 乗根を除去したアルゴリズムを提案している[15]．さらに Barreto らによって初めて η_T ペアリングが提案された[2]．η_T ペアリングは Duursma-Lee アルゴリズムと比較すると，アルゴリズム中のループ回数が約半分となり，より高速な計算が可能となる．ま

†1 公立はこだて未来大学大学院システム情報科学研究科
Graduate School of Systems Information Science, Future University-Hakodate
†2 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba

本稿の内容は 2006 年 10 月のコンピュータセキュリティシンポジウム 2006（CSS2006）にて報告され，CSEC 研究会主査により情報処理学会論文誌への掲載が推薦された論文である．

た白勢らによって 3 乗根を用いない η_T ペアリングのアルゴリズムが示されている[20]．一方，ペアリングには最終べきの計算が存在する．Duursma-Lee アルゴリズムの最終べきに対する効率的な計算は Kerins らによって示されている[14]．また η_T ペアリングの最終べきの計算は，白勢らによってトーラス T_2 を用いた高速な最終べきのアルゴリズムが提案されている[19]．現在，最も高速なペアリングは 3 乗根を用いない η_T ペアリングと，白勢らの最終べきの計算の組合せにより計算される．

携帯電話やスマートカードなどのユビキタスデバイスの普及にともない，これらのデバイス上のセキュリティも重要な問題としてあげられる．ユビキタスデバイスはリソースが少ないため，実装する暗号は低リソース環境においても高速に実装できる必要がある．現在，ユビキタスデバイス上の実装は鍵長の短い楕円曲線暗号が適しているが，新たな暗号システムを実装できることから，ユビキタスデバイス上でのペアリング暗号システムの利用を考慮する必要がある．ペアリング暗号は従来の公開鍵暗号に比べ処理が複雑であるため，低リソースのユビキタスデバイス上での速度評価が重要となる．これまで評価では，$\mathbf{F}_{2^m}$ での η_T ペアリングがスマートカード上で 1 秒以下で処理が可能であると発表されている[18]．本稿では，ユビキタスデバイスとして一般的である携帯電話を用いて η_T ペアリングの速度評価を行う．

現在，携帯電話の多くで Java$^{\mathrm{TM}}$ がサポートされ，一般の開発者でも機関の審査などを通さずに容易にソフトウェアを公開することができる．したがって，本稿では携帯電話を用いた Java での実装および評価を行う．Java ではセキュリティのためのコンポーネントが提供され，安全性の高い実装を行うことができる[22]．携帯電話などの組み込み機器への Java の実装の多くは Java 2 Platform, Micro Edition（J2ME）[21] が使用されている．Tillich らは J2ME 環境で携帯電話上の標準的な公開鍵暗号の実装を示している[23]．しかし，現状では Java を利用した携帯電話上でのペアリング暗号システムの実装はまだ報告されていない．

本稿では，Java を利用した携帯電話上での η_T ペアリングの速度評価を行う．プログラムは標数 3，次数 m，既約多項式を 3 項式 $f(x) = x^m + x^k + 2$ とし，次数 m と既約多項式の中間項のべき k について可変である汎用的な実装を行う．ペアリングの計算には，3 乗根を用いない η_T ペアリングとトーラス T_2 を用いた最終べきの計算を使用する．速度の評価には，次数 $m = \{97, 167, 193, 239, 313\}$ の有限体 $\mathbf{F}_{3^m}$ を用いる．また次数 $m = 97$ に特化し，有限体 $\mathbf{F}_{3^{97}}$ 上の 3 乗根を用いない η_T ペアリングの実装・評価を行う．これは汎用的な実装との比較として，他の多くの文献で実装報告のある次数 $m = 97$ で次数を固定し，プログラムを最適化したものである．結果，有限体の次数 $m = 97$ に特化した実装を用いた携帯電話上での評価では 0.5 秒以下という結果を得た．

以下に，本稿の構成について記述する．2 章では有限体の元の表現と演算について述べる．3 章ではペアリングの実装方法について述べ，4 章で Java 実装の詳細について説明する．最後に 5 章で本稿をまとめる．

2. 標数 3 の有限体上の演算

本稿で扱う η_T ペアリングの計算では，標数 3，次数 m の有限体 $\mathbf{F}_{3^m}$ と，その 6 次拡大体 $\mathbf{F}_{3^{6m}}$ の演算を使用する．6 次拡大体 $\mathbf{F}_{3^{6m}}$ は 3 次拡大体 $\mathbf{F}_{3^{3m}}$ を 2 次拡大することにより構成する．本章では，有限体 $\mathbf{F}_{3^m}$，3 次拡大体 $\mathbf{F}_{3^{3m}}$，6 次拡大体 $\mathbf{F}_{3^{6m}}$ の演算について説明する．

2.1 $\mathbf{F}_{3^m}$ のビット表現

基礎体 $\mathbf{F}_3 = \{0, 1, 2\}$ が与えられると，基礎体の元 a は $a = (a_{hi}, a_{lo})$ $(a_{hi}, a_{lo} \in \{0, 1\})$ と 2 ビットで表現でき，基礎体の各値を $0 = (0, 0), 1 = (0, 1), 2 = (1, 0)$ とする．また負数 $-a$ は $2a$ と置き換えることにより表現することができる．

基礎体 $\mathbf{F}_3$ 上のすべての多項式を $\mathbf{F}_3[x]$，$\mathbf{F}_3$ 上の m 次の既約多項式を $f(x)$ とすると，有限体 $\mathbf{F}_{3^m}$ は $\mathbf{F}_{3^m} = \mathbf{F}_3[x]/f(x)$ である．この有限体の元を $A(x)$ とすると $A(x)$ は最大 $m-1$ 次の多項式となり，これを $\sum_{i=0}^{m-1} a_i x^i$ $(a_i \in \mathbf{F}_3)$ とする[4]．また，この元の各係数 $a_i = ((a_i)_{hi}, (a_i)_{lo}) \in \mathbf{F}_3$ を hi, lo ビットに分けることにより，

$$A_{hi} = ((a_{m-1})_{hi}, (a_{m-2})_{hi}, \cdots, (a_0)_{hi})$$
$$A_{lo} = ((a_{m-1})_{lo}, (a_{m-2})_{lo}, \cdots, (a_0)_{lo})$$

と表現することができ，これを $A(x) = (A_{hi}, A_{lo})$ と記述する．

この表現を用いると，対象のプロセッサの変数の型のサイズ W に対して，$\mathbf{F}_{3^m}$ の元を保持するためにサイズ $N = \lceil m/W \rceil$ の配列が 2 つ必要となる．また負の元 $-A(x)$ は A_{hi} と A_{lo} を入れ替えることで $2A(x)$ に置き換えることができる．

2.2 $\mathbf{F}_{3^m}$ での加算・乗算

$A(x) = (A_{hi}, A_{lo})$，$B(x) = (B_{hi}, B_{lo})$ を $\mathbf{F}_{3^m}$ の元とすると，有限体 $\mathbf{F}_{3^m}$ 上の加算 $C(x) = (C_{hi}, C_{lo}) = A(x) + B(x)$ は基本論理演算（AND(&), OR(|), XOR(∧)）を利用し **図 1** のとおりに計算す

入力：	$A(x) = (A_{hi}, A_{lo}), B(x) = (B_{hi}, B_{lo})$
出力：	$C(x) = (C_{hi}, C_{lo}) = A(x) + B(x)$
1:	$T = (A_{hi} \mid A_{lo}) \,\&\, (B_{hi} \mid B_{lo})$
2:	$C_{hi} = T \wedge (A_{hi} \mid B_{hi})$
3:	$C_{lo} = T \wedge (A_{lo} \mid B_{lo})$

図 1 $\mathbf{F}_{3^m}$ 上の加算
Fig. 1 Addition in $\mathbf{F}_{3^m}$.

る．また減算 $A(x) - B(x)$ は $B'(x) = 2B(x)$ と置換し $A(x) + B'(x)$ を計算することで求める．

有限体 $\mathbf{F}_{3^m}$ 上の乗算 $C(x) = A(x) \cdot B(x) \bmod f(x)$ は，元 $A(x), B(x) \in \mathbf{F}_{3^m}$ と既約多項式 $f(x)$ を入力とすると，多項式乗算と Reduction の 2 ステップによって求められる．多項式乗算では $A(x)$，$B(x)$ から $C'(x) = \sum_{i=0}^{2m-2} c'_i x^i = A(x) \cdot B(x)$ を求める．この計算は Shift-Addition 法が最もシンプルな方法で，計算途中に必要となるメモリ量が Comb 法[10] などに比べ少ない．また，Shift-Addition 法では Comb 法に比べシフト演算の回数が増加するが，変数への参照などを減少させることができる．したがって，携帯電話などのメモリが制限された環境では Shift-Addition 法を用いるのが適しているといえる．

また乗算の高速化の一般的な手法として Window 法[10] がある．Window 法では事前計算を行うことで，走査する係数を複数にし，走査・加算の回数を減少させる手法である．本稿では文献 10) の Alg 2.36 の Window 法を Shift-Addition 法で適応し，窓幅 2 の Window 法を実装した．そして窓幅 2 の Window 法と Window 法を用いない Shift-Addition 法との比較評価を行った．窓幅 2 の Window 法では，事前計算を保持するためのメモリ領域の確保にかかる時間や事前計算結果を保持している配列の各要素へのアクセス速度が低速であり，約 1.8 倍低速となった．そのため本稿では Window 法を利用した実装は行わなかった．

$\mathbf{F}_{3^m}$ の元 $A(x) = \sum_{i=0}^{m-1} a_i x^i$，$B(x) = \sum_{i=0}^{m-1} b_i x^i$ に対し，多項式乗算 $C'(x) = \sum_{i=0}^{2m-2} c'_i x^i = A(x) \cdot B(x)$ の計算は **図 2** によって計算する．

Reduction は $n > m-1$ に対して，入力を多項式 $C'(x) = c'_n x^n + \cdots + c'_0 \in \mathbf{F}_3[x]$ （$c'_i \in \mathbf{F}_3$, $0 \le i \le n$）としたとき，$\mathbf{F}_3$ 上の既約多項式 $f(x) = x^m + x^k + 2$（$m > k \ge 1$）を用い，**図 3** のとおりに処理する．図 3 の Reduction では 1 個の係数 c'_i に対して処理を行っているが，$m - k \ge W$ の場合は W 個の係数を同時に計算することが可能である．入力 $C'(x)$ に対して，配列のある要素に格納されている係数を $C'[j] = (c'_{jW+(W-1)}, \cdots, c'_{jW})$ と表現する．$C'[j]$ の

入力：	$A(x) = \sum_{i=0}^{m-1} a_i x^i, B(x) = \sum_{i=0}^{m-1} b_i x^i$ $\in \mathbf{F}_{3^m}$ $(a_i, b_i \in \mathbf{F}_3)$
出力：	$C'(x) = A(x) \cdot B(x)$
1:	$C'(x) \leftarrow 0$
2:	**for** $i \leftarrow 0$ **to** $m-1$ **then**
3:	$\quad C'(x) \leftarrow C'(x) + A(x) \cdot b_i x^i$
4:	**end for**

図 2 $\mathbf{F}_{3^m}$ 上の Shift-Addition 乗算
Fig. 2 Shift-Addition multiplication in $\mathbf{F}_{3^m}$.

入力：	$C'(x) = c'_n x^n + \cdots + c'_0 \in \mathbf{F}_3[x]$ $(n > m-1), f(x) = x^m + x^k + 2$
出力：	$C(x) = C'(x) \bmod f(x)$
1:	**for** $i \leftarrow n$ **downto** m **then**
2:	$\quad c'_{i-m+k} \leftarrow c'_{i-m+k} - c'_i$
3:	$\quad c'_{i-m} \leftarrow c'_{i-m} - 2c'_i$
4:	$\quad c'_i \leftarrow 0$
5:	**end for**
6:	**return** $C'(x)$

図 3 既約 3 項式 $f(x)$ によるリダクション
Fig. 3 Reduction with irreducible trinomial $f(x)$.

最大次数の係数 $c'_{jW+(W-1)}$ は Reduction を行うと $x^{jW+(W-1)-m+k}$，$x^{jW+(W-1)-m}$ の係数に対して計算を行う．$m - k \ge W$ のとき $x^{jW+(W-1)-m+k}$ は $C'[j]$ の最低の次数 x^{jW} よりも次数が小さいため，$C'[j]$ 中の W 個の係数を同時に計算することができ，Reduction の高速処理が可能である．

2.3 $\mathbf{F}_{3^m}$ 上の他の演算

$\mathbf{F}_{3^m}$ 上の他の演算をそれぞれ 3 乗算（Cubing），逆元算（Inversion），3 乗根（Cube Root）とする．

Cubing： 多項式 $A(x) \in \mathbf{F}_{3^m}$ において，3 乗算は $A(x)^3 = \sum_{i=0}^{m-1} a_i x^{3i}$ と計算される．3 乗算の計算では多項式乗算部分は実質，時間を必要しない．Reduction の方法としては，前述した通常の Reduction を行う方法と，Reduction 結果のテーブルを事前に作成し $A(x) \in \mathbf{F}_{3^m}$ から直接結果を求める方法がある．後者は次数 m に特化しているため高速ではあるが次数によって異なるテーブルを必要とする．したがって，一般的な次数 m に対しては前者を利用するのが適切である．

Inversion： 逆元算は $\mathbf{F}_3$ 上の多項式の拡張ユークリッド互助法を利用する．これは $\mathbf{F}_2$ 上の多項式の拡張ユークリッド互助法[10] を改良したアルゴリズムである．逆元を計算する他の方法として ternary gcd[11] があるが，速度を比較した結果 $\mathbf{F}_3$ 上の多項式の拡張

入力： $A(x) \in \mathbf{F}_3[x]/f(x),\ f(x) = x^m + x^k + 2$
出力： $A(x)^{-1} \bmod f(x)$

1: $u \leftarrow A(x),\ v \leftarrow f(x)$
2: $g \leftarrow 1,\ h \leftarrow 0$
3: **while** $deg(u) \neq 0$ **then**
4: $\quad j \leftarrow deg(u) - deg(v)$
5: $\quad$ **if** $j < 0$ **then**
6: $\quad\quad u \leftrightarrow v,\ g \leftrightarrow h,\ j \leftarrow -j$
7: $\quad$ **end if**
8: $\quad u \leftarrow u - (u_{deg(u)} \cdot v_{deg(v)})\ vx^j$
9: $\quad g \leftarrow g - (u_{deg(u)} \cdot v_{deg(v)})\ hx^j$
10: **end while**
11: **return** $u_0 g$

図 4 $\mathbf{F}_{3^m}$ 上の逆元算
Fig. 4 Inversion in $\mathbf{F}_{3^m}$.

ユークリッド互助法を使用した．u, v を $\mathbf{F}_{3^m}$ の元，$deg(u)$ を元 u に対する非零項の最大次数を求める関数で $0 \leq deg(u) \leq m$ とする．また u，v の非零項の最大次数の係数 $u_{deg(u)}, v_{deg(v)} \in \mathbf{F}_3$ に対して，それらの基礎体 $\mathbf{F}_3$ 上の乗算を $u_{deg(u)} \cdot v_{deg(v)}$ とする．入力 $A(x) \in \mathbf{F}_{3^m} = \mathbf{F}_3[x]/f(x)$，$f(x) = x^m + x^k + 2$ に対し，逆元 $A(x)^{-1} \bmod f(x)$ は **図 4** のとおりに計算される．

Cube Root： 3 乗根の効率的な演算は Barreto によって示されている[1]．3 乗根の計算はたかだか $\mathbf{F}_{3^m}$ 上の乗算 2 回の計算で求めることができる．

2.4 3 次拡大体 $\mathbf{F}_{3^{3m}}$ の演算

$\mathbf{F}_{3^m}$ 上の既約多項式を $g(\rho) = \rho^3 - \rho - 1$ とすると，有限体 $\mathbf{F}_{3^m}$ の 3 次拡大体 $\mathbf{F}_{3^{3m}}$ は $\mathbf{F}_{3^m}[\rho]/g(\rho)$ と表現される．$A(\rho)$ を 3 次拡大体の元とすると，この元の係数は $a_0, a_1, a_2 \in \mathbf{F}_{3^m}$ で，$A(\rho) = a_2\rho^2 + a_1\rho + a_0$ と表現できる．

3 次拡大体上の演算は元 $A(\rho) = a_2\rho^2 + a_1\rho + a_0$，$B(\rho) = b_2\rho^2 + b_1\rho + b_0 \in \mathbf{F}_{3^{3m}}$ （$a_i, b_i \in \mathbf{F}_{3^m}$，$i = 0, 1, 2$）に対し，以下のとおりに計算される．

Addition： $A(\rho) + B(\rho) = (a_2 + b_2)\rho^2 + (a_1 + b_1)\rho + (a_0 + b_0)$.

Multiplication： $t_{00} = a_0 b_0$，$t_{11} = a_1 b_1$，$t_{22} = a_2 b_2$，$t_{01} = (a_0 + a_1)(b_0 + b_1)$，$t_{12} = (a_1 + a_2)(b_1 + b_2)$，$t_{02} = (a_0 + a_2)(b_0 + b_2)$ とすると，$A(\rho)B(\rho) = (t_{12} + t_{00} - t_{11} - t_{22}) + (t_{12} + t_{01} + t_{11} - t_{00})\rho + (t_{02} + t_{11} - t_{00})\rho^2$．$\mathbf{F}_{3^{3m}}$ 上の乗算は $\mathbf{F}_{3^m}$ 上の乗算 6 回と加減算で計算される．

Cubing： $A(\rho)^3 = a_2^3\rho^2 + (a_1^3 - a_2^3)\rho + (a_0^3 + a_1^3 + a_2^3)$．$\mathbf{F}_{3^{3m}}$ 上の 3 乗算は $\mathbf{F}_{3^m}$ 上の 3 乗算 3 回と加減算によって計算される．

Inversion： 3 次拡大体上の逆元算は Kerins らによって示されているアルゴリズムを利用し実装した[14]．このアルゴリズムでは，$\mathbf{F}_{3^m}$ 上の逆元算 1 回，乗算 12 回と加減算によって計算される．

2.5 6 次拡大体 $\mathbf{F}_{3^{6m}}$ の演算

$\mathbf{F}_{3^{3m}}$ 上の既約多項式 $h(\sigma) = \sigma^2 + 1$ に対し，6 次拡大体 $\mathbf{F}_{3^{6m}}$ は $\mathbf{F}_{3^{3m}}[\sigma]/h(\sigma)$ である．$A(\sigma)$ を $\mathbf{F}_{3^{6m}}$ の元とすると $A(\sigma) = a_1\sigma + a_0$ と記述することができ，a_0，a_1 は $\mathbf{F}_{3^{3m}}$ の元となる．

6 次拡大体上の演算は元 $A(\sigma) = a_1\sigma + a_0, B(\sigma) = b_1\sigma + b_0 \in \mathbf{F}_{3^{6m}}$（$a_i, b_i \in \mathbf{F}_{3^{3m}}$，$i = 0, 1$）に対し以下のとおりに計算する．

Addition： $A(\sigma) + B(\sigma) = (a_1 + b_1)\sigma + (a_0 + b_0)$.

Multiplication： $t_{00} = a_0 b_0$，$t_{11} = a_1 b_1$，$t_{01} = (a_0 + a_1)(b_0 + b_1)$ とすると，$A(\sigma)B(\sigma) = (t_{00} - t_{11}) + (t_{01} - t_{00} - t_{11})\sigma$．$\mathbf{F}_{3^{6m}}$ 上の乗算は $\mathbf{F}_{3^m}$ 上の乗算 18 回と加減算で計算される．

Cubing： $A(\sigma)^3 = -a_1^3\sigma + a_0^3$．$\mathbf{F}_{3^{6m}}$ 上の 3 乗算は $\mathbf{F}_{3^m}$ 上の 3 乗算 6 回と加減算によって計算される．

Inversion： $A(\sigma)^{-1} = (a_0 - a_1\sigma)(a_1^2 + a_0^2)^{-1}$．$\mathbf{F}_{3^{6m}}$ 上の逆元は $\mathbf{F}_{3^m}$ 上の 1 回の逆元算，36 回の乗算と加減算によって計算される．

3. Tate ペアリング

本稿では Tate ペアリングの実装に有限体 $\mathbf{F}_{3^m}$ 上の超特異曲線を使用する．

$$E(\mathbf{F}_{3^m}) = \{(x, y) \in (\mathbf{F}_{3^m})^2 \mid y^2 = x^3 - x + 1\} \cup \{\mathcal{O}\}$$

楕円曲線 $E(\mathbf{F}_{3^m})$ 上の点は $\mathbf{F}_{3^m}$ の元 x，y に対し (x, y) と表現する．無限遠点を含めた $E(\mathbf{F}_{3^m})$ のすべての点は加算に対して群構造を成し，$E(\mathbf{F}_{3^m})$ の位数は $\#E(\mathbf{F}_{3^m}) = 3^m + 3^{(m+1)/2} + 1$ となる．

l を $l | \#E(\mathbf{F}_{3^m})$，$l | (3^{6m} - 1)$ を満たす大きな素数とし，$E(\mathbf{F}_{3^m})[l]$ を位数 l の $E(\mathbf{F}_{3^m})$ の部分群とする．また，$E(\mathbf{F}_{3^m})[l]$ の点 $Q = (x, y)$ に対して，distortion map を $\phi(x, y) = (-x + \rho, y\sigma)$ と定義する．このとき $\phi(Q)$ は拡大体 $\mathbf{F}_{3^{6m}}$ 上の楕円曲線の部分群 $E(\mathbf{F}_{3^{6m}})[l]$ の点となる．ペアリングは以下の写像である．

$$e : E(\mathbf{F}_{3^m})[l] \times E(\mathbf{F}_{3^{6m}})/lE(\mathbf{F}_{3^{6m}}) \to \mathbf{F}_{3^{6m}}^*/(\mathbf{F}_{3^{6m}}^*)^l$$
$$(P, \phi(Q)) \mapsto e(P, Q).$$

ペアリングでは非ゼロの整数 a に対して，双線形性 $e(aP, Q) = e(P, aQ) = e(P, Q)^a$ を満たす．

超特異曲線 $E(\mathbf{F}_{3^m})$ 上のペアリングを利用した暗号システムの安全性は，楕円曲線 $E(\mathbf{F}_{3^m})$ および有限

入力： $P=(x_p,y_p), Q=(x_q,y_q) \in E(\mathbf{F}_{3^m})[l]$
出力： $\eta_T(P,Q)^{3^{(m+1)/2}} \in \mathbf{F}^*_{3^{6m}}$

1: $y_p \leftarrow -y_p,\ d \leftarrow 1$
2: $f \leftarrow y_q\sigma - y_p(x_p+x_q+1)+y_p\rho$
3: **for** $i \leftarrow 0$ **to** $(m-1)/2$ **then**
4: $u \leftarrow x_p+x_q+d$
5: $g \leftarrow y_py_q\sigma - u^2 - u\rho - \rho^2$
6: $f \leftarrow fg$
7: $y_p \leftarrow -y_p$
8: $x_q \leftarrow x_q^9,\ y_q \leftarrow y_q^9$
9: $d \leftarrow (d-1) \bmod 3$
10: $f \leftarrow f^3$
11: **end for**
12: **return** $f^{(3^{3m}-1)(3^m+1)(3^m-3^{(m+1)/2}+1)}$

図 5 3 乗根を用いない η_T ペアリング：
$E: y^2 = x^3 - x + 1,\ m \equiv 1 \pmod{12}$ [20]
Fig. 5 η_T Pairing without Cube Root on $E: y^2 = x^3 - x + 1,\ m \equiv 1 \pmod{12}$.[20]

体 $\mathbf{F}^*_{3^{6m}}$ における離散対数問題の困難性に基づいている．達成したいセキュリティレベルに応じて，部分群 $E(\mathbf{F}_{3^m})[l]$ の位数 l および有限体 $\mathbf{F}^*_{3^{6m}}$ の位数 $3^{6m}-1$ を選択する必要がある．たとえば，次数 $m=97$ を選択すると，$\log_2 l \approx 160$ ビット，$\log_2(3^{6m}-1) \approx 1024$ ビットとなり，現在使用されている RSA 暗号や楕円曲線暗号と同程度のセキュリティが達成できる．また，拡大次数 m は素数として選ばれる[9),11)]．これらの条件を満たす $m > 97$ の拡大次数として，$m = 167$, 193, 239, 313 などが選択できる．

3.1 η_T ペアリング

Tate ペアリングのアルゴリズムは Miller によって提案され，Duursma と Lee によって使用する楕円曲線を超特異曲線に限定し高速化したアルゴリズムが示された[8)]．Barreto らは Duursma-Lee アルゴリズムのループ処理の回数を約半分にした η_T ペアリングを提案した[2)]．また白勢らによって，3 乗根を用いない η_T ペアリングのアルゴリズムが示された[20)]．文献 2), 20) で示されている η_T ペアリングで 3 乗根の有無による計算コストを比較すると，1 回のループ処理に対して $\mathbf{F}_{3^m}$ 上の 3 乗根が 2 回減少し，$\mathbf{F}_{3^m}$ 上の 3 乗算 2 回と $\mathbf{F}_{3^{6m}}$ 上の 3 乗算 1 回増加する．一般的な実装では 3 乗根に比べ 3 乗算が高速であるため，3 乗根を用いないアルゴリズムの方が高速処理が可能となる．3 乗根を用いない η_T ペアリングのアルゴリズムを **図 5** に示す．

この η_T ペアリング $\eta_T(P,Q)$ はあるべき乗をとることによって簡単に Tate ペアリング $e(P,Q)$ と関連付けることができる．

図 5 のステップ 12 では，η_T ペアリングの最終べきの計算を行う．η_T ペアリングの最終べきは Duursma-Lee アルゴリズムに比べ複雑であるが効率的な計算が示されている．$f = f_0 + f_1\sigma \in \mathbf{F}_{3^{6m}}$（$f_0, f_1 \in \mathbf{F}_{3^{3m}}$）とすると，$f^{3^{3m}-1}$ の計算は $f^{3^{3m}-1} = (f_0+f_1\sigma)^{3^{3m}}/(f_0+f_1\sigma) = (f_0-f_1\sigma)/(f_0+f_1\sigma)$ と整理することにより高速に計算可能であるであることが Kerins らにより示された[14)]．また，$f^{3^{3m}-1}$ がトーラス T_2 の元であることから，残りのべきをトーラス T_2 の性質を用いて計算する効率的なアルゴリズムが白勢らによって提案された[19)]．白勢らの見積りによると，これらを使用した最終べきの計算は，次数 $m=97$ のとき全体の 10% 程度の処理時間を必要となる[19)]．

4. Java での実装

Java ではセキュリティのための暗号関連のコンポーネントが提供され，標準的な公開鍵暗号のライブラリは Bouncy Castle [16)] や IAIK [12)] など様々な組織が提供している．しかし現状では携帯電話上で動作可能なペアリング暗号の実装はまだ報告されていない．

本稿では，Java を利用した携帯電話上での η_T ペアリングの速度評価を行う．プログラムは標数 3，次数 m，既約多項式を 3 項式 $f(x) = x^m + x^k + 2$ とし，次数 m と既約多項式の中間項のべき k について可変である実装を行う．評価は次数 $m = \{97, 167, 193, 239, 313\}$ の有限体 $\mathbf{F}_{3^m}$ 上の演算と η_T ペアリングについて行う．また次数 $m=97$ について次数を特化した実装を行い，有限体 $\mathbf{F}_{3^{97}}$ 上の演算と η_T ペアリングの評価を行う．複数の次数，既約多項式で使用可能なプログラムの実装はコンポーネントを汎用的に利用すること目的とした．また次数の特化は汎用的な実装との比較として，他の多くの文献で実装報告のある次数 $m=97$ で次数を固定し，プログラムを最適化したものである．

4.1 実装詳細

今回の実装はスクラッチから行った．プログラムはペアリングの計算を効率的に行うことが目的であるため，既存のコンポーネントの使用を少なくし，またクラス構成をシンプルにすることにより余分なオーバヘッドを減少させた．今回は基本論理演算を利用するために，有限体 $\mathbf{F}_{3^m}$ の元をビットにより表現した．クラス構成は Finite Field Parameters，Finite Field，Extension Field of Degree 3，Extension Field of Degree 6，Elliptic Curve Point，Tate Pairing の 6 個からなる．Finite Field Parameters では 2 章で示した標数

3，次数 m，既約多項式の中間項のべき k，配列のサイズ N といった有限体 $\mathbf{F}_{3^m}$ の変数を保持するため，様々な次数や既約多項式に対応できる．

一方，$\mathbf{F}_{3^{97}}$ に対する特化では，有限体のパラメータを保持するクラスが不要で，プログラム中で即値の記述ができる．また乗算では，次数 $m = 97$，変数のサイズ $W = 32$，配列のサイズ $N = \lceil m/W \rceil = 4$ であるため，$m \bmod W \equiv 1$ より，配列の 4 つ目の要素には 1 つの係数のみが格納される．多項式乗算時の入力配列の左シフトは最大 31 ビットであるので，シフト後の配列のサイズ $\lceil (m+31)/W \rceil = 4$ となり元の配列のサイズ N から増加しない．したがって，シフト後の加算においてつねに一定のサイズの配列に対して加算を行うため処理を高速化できる．また 3 乗算では 2.3 節で述べた，次数を特化し事前計算テーブルを利用するアルゴリズムを適応することで処理を高速化できる．

4.2 実測結果

本節では，速度評価の結果を記述する．使用した携帯電話は NTT DoCoMo の携帯電話 FOMA 902iS シリーズの 5 機種と FOMA SH903i，FOMA SH901iS の 7 機種で，携帯電話上で Java を動作させるために i αppli を使用した[⋆1]．

実験した次数は $m = \{97, 167, 193, 239, 313\}$ で，既約多項式はそれぞれ $x^{97}+x^{12}+2$，$x^{167}+x^{96}+2$，$x^{193}+x^{12}+2$，$x^{239}+x^{24}+2$，$x^{313}+x^{126}+2$ とした．汎用プログラムは次数 $m = \{97, 167, 193, 239, 313\}$ に対して行い，また次数 $m = 97$ で特化したプログラムは “97(*opt*)” と記述する．各演算の演算回数は，有限体 $\mathbf{F}_{3^m}$ 上の加減算 4,000,000 回，乗算 500,000 回，3 乗算 1,000,000 回，逆元算 100,000 回，3 乗根 500,000 回，η_T ペアリングを 5,000 回行い速度の平均をとった．入力は毎回，有限体 $\mathbf{F}_{3^m}$ 上のランダムな元を選択し計算を行った．異なる機種でのテストの際の入力の元は，ランダムな元を選択しているため必ずしも同一の元を入力としていない．7 機種中最も高速な SH903i についての演算速度の結果を **表 1** に示す．

有限体 $\mathbf{F}_{3^m}$ での汎用プログラムの加算，減算，3 乗算の処理速度は，配列のサイズと関連する．そのため次数 m の上昇に従って，処理速度は $O(m)$ となる．一方，乗算や逆元算，3 乗根では次数 m がループ処理と加算の回数と関連する．したがって，次数 m の上昇により処理速度は $O(m^2)$ となる．η_T ペアリングのアルゴリズムでは次数 m がループ処理の回数に影響する．またループ中の処理の大部分は $\mathbf{F}_{3^m}$ と $\mathbf{F}_{3^{6m}}$ 上の乗算と 3 乗算で，処理の時間的なウェイトは特に乗算に影響される．したがって，η_T ペアリングの処理速度は次数の上昇にともない $O(m^3)$ となる．

次数 $m = 97$ での汎用プログラムと特化したプログラムとの比較では，次数を特化することで汎用プログラムよりも全体的に処理速度が向上している．これは特化したプログラムでは，即値が記述や乗算，3 乗算の最適化といった，4.1 節で述べた最適化によるものである．

一方，各機種の η_T ペアリングの処理速度の結果を **表 2** に示す．次数 $m = 97$ で特化した場合の速度を例に比較すると，同世代の携帯電話 FOMA 902iS では，SH902iS の処理速度が最も速く，最も遅い D902iS に比べ 1.8 倍程度高速である．携帯電話の異なる世代での比較では，2 世代差のある SH902iS と SH901iS では約 1.4 倍処理速度が向上している．また 1 世代差のある SH903i と SH902iS を比較すると約 1.2 倍処理速度が向上している．以上より，同世代の異なる携帯電話でも処理性能にばらつきがあり，最大で 1.8 倍程度の速度差があるが，今回の次数 $m = 97$ で特化した場合の実装では FOMA 902iS のすべての機種において 0.5 秒以下の速度を得ることができ，機種の違いによる影響は少ないと考えられる．また，異なる世代の携帯電話の処理能力が向上していることから，今後も処理能力の向上を望むことができる．

次に FOMA SH903i を用いて従来の公開鍵暗号の速度評価を行った．評価には Bouncy Castle[16] が公開している Java ライブラリを使用した．このライブラリでは多くの面で汎用性を持ち，例外処理なども多いためあまり高速ではない．次数 $m = 97$ の η_T ペアリングと同程度のセキュリティレベルである 1,024 ビットの RSA 暗号のべき剰余および $\mathbf{F}_{2^{163}}$ 上の楕円曲線暗号のスカラー倍算の速度は，それぞれ 3,049.50 ミリ秒，6,990.76 ミリ秒であった．

最後に，同様のコンポーネントを用いて PC 上での速度評価との比較を示す．PC 上での評価に使用した PC のスペックは CPU: Intel Pentium D (3.20 GHz)，RAM: 2 GByte で，PC 上での各演算の演算回数は有限体 $\mathbf{F}_{3^m}$ 上の加減算 4,000,000 回，乗算 500,000 回，3 乗算 2,000,000 回，逆元算 250,000 回，3 乗根 500,000 回，η_T ペアリングを 20,000 回行い速度の平均をとった．PC での JVM は Java HotSpot$^{\text{TM}}$ Server VM を使用した．入力は毎回，有限体 $\mathbf{F}_{3^m}$ 上のランダムな元を選択し計算を行った．**表 3** は PC 上の各次数

⋆1 FOMA, i αppli, DoJa は株式会社エヌ・ティ・ティ・ドコモの登録商標である．

表 1 携帯電話 SH903i 上での有限体 $\mathbf{F}_{3^m}$ の演算と η_T ペアリングの計算速度 (msec)
Table 1 Timing of arithmetic of finite field $\mathbf{F}_{3^m}$ and the η_T pairing on SH903i (msec).

Degree(m)	97(opt)	97	167	193	239	313
Addition	0.0089	0.0094	0.0107	0.0111	0.0116	0.0131
Subtraction	0.0093	0.0094	0.0107	0.0111	0.0116	0.0134
Multiplication	0.1786	0.2180	0.4977	0.6548	0.9031	1.4358
Cubing	0.0177	0.0380	0.0481	0.0555	0.0603	0.0722
Inversion	1.0307	1.0973	2.6325	3.4890	4.8382	7.8790
Cube Root	0.1655	0.2109	0.2791	0.3733	0.5189	0.3939
η_T Pairing	**214.50**	272.13	850.54	1,241.24	2,015.57	3,943.17

表 2 各携帯電話の有限体 $\mathbf{F}_{3^m}$ 上の η_T ペアリングの計算速度の比較 (msec)
Table 2 Comparison of timing of the η_T pairing on several mobile phones (msec).

Degree(m)	97(opt)	97	167	193	239	313
SH903i	214.50	272.13	850.54	1,241.24	2,015.57	3,943.17
SH902iS	254.05	318.97	962.17	1,338.08	2,135.12	4,102.79
N902iS	305.27	382.24	1,121.11	1,579.10	2,416.62	4,635.49
P902iS	350.04	414.14	1,165.25	1,646.50	2,542.11	4,929.68
F902iS	408.63	516.74	1,527.19	2,069.24	3,244.14	6,282.27
D902iS	449.21	544.11	1,585.26	2,097.00	3,312.14	6,260.61
SH901iS	356.96	436.49	1,313.90	1,838.05	2,888.96	5,516.07

表 3 PC 上での有限体 $\mathbf{F}_{3^m}$ の演算と η_T ペアリングの計算速度 (μsec)
Table 3 Timing of arithmetic of finite field $\mathbf{F}_{3^m}$ and the η_T pairing on PC (μsec).

Degree(m)	97(opt)	97	167	193	239	313
Addition	0.0778	0.1253	0.1555	0.1645	0.1680	0.1990
Subtraction	0.0820	0.1255	0.1598	0.1755	0.1833	0.2108
Multiplication	3.9780	5.3180	9.9960	15.7160	20.7440	26.9360
Cubing	0.1325	0.5550	0.7820	0.8680	0.9685	1.1170
Inversion	19.7680	22.2200	52.2520	68.9240	90.7760	122.8080
Cube Root	3.2280	4.3800	6.0260	7.9960	11.2540	8.8140
η_T Pairing	3,772.55	5,442.15	17,371.55	26,707.45	43,270.75	71,897.55

での有限体 $\mathbf{F}_{3^m}$ の演算と η_T ペアリングの処理速度の結果である．

PC 上での η_T ペアリングの速度評価では次数 $m = 97$ で特化したプログラムで 5 ミリ秒以下という結果を得た．これは携帯電話上で最も高速な SH903i での速度に比べ，50 倍以上高速であることが分かる．

汎用プログラムにおいて，1 度の η_T ペアリングの計算に使用するメモリ量は，DoJa-5.0 Emulator 上で MemoryManager クラスを使用し測定した[13]．各次数 $m = \{97, 167, 193, 239, 313\}$ において，それぞれ 0.849 MByte，1.724 MByte，2.156 MByte，2.865 MByte，4.282 MByte という結果を得た．今回使用した携帯電話のヒープ領域は平均で約 8 MByte であるため，複数回のペアリングを行った際にヒープ領域が足りなくなり Garbage Collection が行われる．

4.3 暗号プロトコルへの適応

本節では，1 章のはじめに述べた 3 種類の暗号プロトコルでの実装例を考察する．文献 5) での ID ベース暗号では，encryption，decryption ともにペアリングを 1 回計算する．また文献 7) の short signature の署名検証ではペアリングを 2 回，文献 6) の効率的なブロードキャスト暗号の decryption においてもペアリングを 2 回計算する．本稿の結果より，次数 $m = 97$ で特化したプログラムにおいて，ペアリング 2 回の計算は 1 秒以内で計算可能である．

5. ま と め

本稿は Java でペアリング暗号を実装し，携帯電話上で評価した最初の論文である．標数 3 の有限体 $\mathbf{F}_{3^m}$ 上の η_T ペアリングで実装し，次数 $m = 97$ で特化した η_T ペアリングでは NTT DoCoMo の携帯電話 FOMA SH903i で 0.5 秒以下の結果を得た．したがって，ペアリング暗号システムの基となるペアリング暗号は携帯電話上でも十分効率的に実装可能といえる．

謝辞 本研究は新エネルギー・産業技術総合開発機構（NEDO）による受託研究「Pairing Lite の研究開発」の助成を受けて行われた．

参 考 文 献

1) Barreto, P.S.L.M.: A note on efficient computation of cube roots in characteristic 3, IACR Cryptology ePrint Archive, Report 2004/305 (2004).
2) Barreto, P.S.L.M., Galbraith, S., O'hEigeartaigh, C. and Scott, M.: Efficient pairing computation on supersingular abelian varieties (2005). To appear in Designs, Codes, and Cryptography.
3) Barreto, P.S.L.M., Kim, H., Lynn, B. and Scott, M.: Efficient algorithms for pairing-based cryptosystems, *CRYPTO 2002*, LNCS, Vol.2442, pp.354–368 (2002).
4) Bertoni, G., Guajardo, J., Kumar, S., Orland, G., Paar, C. and Wollinger, T.: Efficient GF(p^m) arithmetic architectures for cryptographic application, *CT-RSA 2003*, LNCS, Vol.2612, pp.158–175 (2003).
5) Boneh, D. and Franklin, M.: Identity based encryption from the Weil pairing, *SIAM J. Comput.*, Vol.32, No.3, pp.586–615 (2001).
6) Boneh, D., Gentry, C. and Waters, B.: Collusion resistant broadcast encryption with short ciphertexts and private keys, *CRYPTO 2005*, LNCS, Vol.3621, pp.258–275 (2005).
7) Boneh, D., Lynn, B. and Shacham, H.: Short signatures from the Weil pairing, *ASIACRYPT 2001*, LNCS, Vol.2248, pp.514–532 (2001).
8) Duursma, I. and Lee, H.: Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$, *ASIACRYPT 2003*, LNCS, Vol.2894, pp.111–123 (2003).
9) Galbraith, S., Harrison, K. and Soldera, D.: Implementing the Tate pairing, *ANTS V*, LNCS, Vol.2894, pp.324–337 (2002).
10) Hankerson, D., Menezes, A. and Vanstone, S.: *Guide to elliptic curve cryptography*, Springer-Verlag (2004).
11) Harrison, K., Page, D. and Smart, N.: Software implementation of finite fields of characteristic three, for use in pairing-based cryptosystems, *LMS Journal of Computation and Mathematics*, Vol.5, pp.181–193 (2002).
12) Institute for Applied Information Processing and Communication: Stiftung Secure Information and Communication Technologies. http://www.iaik.tugraz.at/
13) 株式会社 NTT ドコモ：i アプリコンテンツ開発ガイド for DoJa-5.x/5.x LE〜詳細編〜第 1.00 版 (2006).
14) Kerins, T., Marnane, W., Popovici, E. and Barreto, P.S.L.M.: Efficient hardware for the Tate pairing calculation in characteristic three, *CHES 2005*, LNCS, Vol.3659, pp.412–426 (2005).
15) Kwon, S.: Efficient Tate pairing computation for supersingular elliptic curves over binary fields, IACR Cryptology ePrint Archive, Report 2004/303 (2004).
16) Legion of the Bouncy Castle: Bouncy Castle Crypto APIs. http://www.bouncycastle.org/
17) Miller, V.: Short program for functions on curves (1986).
18) Scott, M., Costigan, N. and Abdulwahab, W.: Implementing cryptographic pairings on smartcards, IACR ePrint Archive, Report 2006/144 (2006).
19) 白勢政明，高木　剛，岡本栄司：η_T ペアリングの最終べきについて，電子情報通信学会，情報セキュリティ研究会，ISEC2006-98, pp.19–26 (2006).
20) 白勢政明，高木　剛，岡本栄司：Tate ペアリングの効率的な計算方法，電子情報通信学会，情報セキュリティ研究会，ISEC2006-12, pp.19–26 (2006).
21) Sun Microsystems, Inc.: Java 2 Platform, Micro Edition (J2ME). http://java.sun.com/javame/
22) Sun Microsystems, Inc.: Java Cryptography Extension (JCE). http://java.sun.com/products/jce/
23) Tillich, S. and Großchadl, J.: A survey of public-key cryptography on J2ME-enabled mobile devices, *ISCIS 2004*, LNCS, Vol.3280, pp.935–944 (2004).

(平成 19 年 3 月 18 日受付)
(平成 19 年 10 月 2 日採録)

推 薦 文

ペアリングは，近年，暗号プロトコルの効率化などへの応用が期待されている暗号技術であるが，従来の公開鍵暗号より処理速度が遅いという状況にある．本稿は Java を用いてペアリングを実装し，携帯電話上で評価した初めての報告である．標数 3，次数 m の有限体 $\mathbf{F}_{3^m}$ 上の η_T ペアリングの実装が行われ，NTT DoCoMo の携帯電話 7 機種上で評価されている．実際の携帯電話機上で十分効率的に実装可能であることを示した点が評価できる．

（コンピュータセキュリティ研究会主査　寺田真敏）

川原 祐人（学生会員）

2007 年公立はこだて未来大学システム情報科学部卒業．現在，同大学大学院システム情報科学研究科博士（前期）課程在学中．暗号および情報セキュリティに関する研究に従事．電子情報通信学会会員．

高木 剛（正会員）

1993 年名古屋大学理学部数学科卒業．1995 年同大学大学院理学研究科修士課程修了．同年 NTT 情報流通プラットフォーム研究所入社．2001 年理学博士（ダルムシュタット工科大学）．その後，ダルムシュタット工科大学情報科学部助教授を経て，2005 年より公立はこだて未来大学システム情報科学部准教授，現在に至る．暗号および情報セキュリティに関する研究に従事．電子情報通信学会，IACR 各会員．

岡本 栄司（フェロー）

1978 年東京工業大学大学院電子専攻博士課程修了．工学博士．同年 NEC 中央研究所入社．その後，北陸先端科学技術大学院大学，東邦大学を経て，2002 年より筑波大学大学院システム情報工学研究科教授，現在に至る．1990 年電子情報通信学会論文賞，1993 年本会ベストオーサー賞受賞．2003 年電子情報通信学会フェロー，2004 年本会フェロー．著書に『暗号理論入門』（共立出版），『電子マネー』（岩波書店）等．