数理解析研究所講究録 1420 巻 2005 年 183-192

XTRを用いた暗号とその高速実装

岡山大学・工学部・通信ネットワーク工学科　野上　保之（Yasuyuki Nogami）
Communication Network Engineering,
Okayama University

1　はじめに

近年，情報セキュリティ技術，とりわけ電子認証技術の重要性は高まってきており，これを実現するための技術として公開鍵暗号は必要不可欠である．これまでRSA（Rivest Shamir Adleman）暗号が広く用いられてきたが，楕円曲線暗号，XTR（Efficient and Compact Subgroup Trace Representation;ECSTR→XTR）を用いた暗号などが，次世代の公開鍵暗号技術として注目されている [1],[2]．楕円曲線暗号とXTRを用いた暗号との共通点は，RSA暗号に比べて鍵長が短くて済むこと，また暗号化および復号処理が高速に行えること，定義体として位数の大きな有限体を用いていることなどが挙げられる．本論文では，位数の大きな拡大体の高速実装法を，XTRを用いた暗号の暗号化/復号処理の高速化という観点から紹介する．

拡大体の高速実装に関する従来の研究においては，OEF（Optimal Extension Field）[3]，AOPF（All One Polynomial Field）[4]，TypeII AOPF（TypeII All One Polynomial Field）[5] がよく知られている．位数の大きな拡大体の高速実装においては，その拡大体における乗算に必要となる計算コスト，計算処理時間が鍵となる．これら3つの拡大体においては，その標数，法多項式，乗算アルゴリズムなどを工夫することで，計算コストを少なくし，計算処理を高速化している．例えばOEFでは，標数に擬メルセンヌ素数，法多項式に既約2項式を用い，乗算にKaratsuba法 [6] を採用することで高速実装している．一方AOPFでは，TypeI ONB（Optimal Normal Basis）[7] を利用する乗算アルゴリズムCVMA（Cyclic Vector Multiplication Algorithm）を用いて高速化を実現している．ただしAOPFは，偶数次数の拡大体しか構成できず，これを奇数次数の拡大体も構成できるよう改良したものがTypeII AOPFであり，TypeII ONBおよびCVMAを改良したものを用いていることが特徴である．本論文では，XTRを用いた暗号の高速実装という観点から，とくに乗算の計算コストについて，これら拡大体の性能を比較する．

とくに断らない限り，p を素数，m を正整数として，素体 F_p 上の m 次拡大体を F_{p^m} のように表す．$X \mid Y$ および $X \nmid Y$ は，それぞれ X が Y を割り切ること，および割り切らないことをそれぞれ表す．

2　XTR

本章では，楕円曲線暗号とともに次世代の公開鍵暗号に用いることができるとして注目されているXTR（Efficient and Compact Subgroup Trace Representation;ECSTR）について紹介する．

2.1 XTRで用いる乗法群とトレース表現

XTRでは，単位元1を除くすべての元が$F_{p^{6m}}$の真性元[1]であり，かつ位数が$p^{2m}-p^m+1$を割り切る170ビット以上の素数であるような以下に示す乗法群を準備する．ここで$\cdot$は，乗算を表すものとする．

$$< \{g, g^2, \cdots, g^t = 1\}, \cdot >, \quad t \mid (p^{2m}-p^m+1), \quad t \text{は170ビット以上の素数} \tag{1}$$

gは位数がtの$F_{p^{6m}}$の真性元であり上記の乗法群の生成元である．位数tに対して，上式のような条件を課すことによって，$F_{p^{6m}}$の真性元であることが保障されることとなる．この乗法群の各元を，以下に示すトレース関数を用いて$F_{p^{2m}}$の元に写像し，

$$\mathrm{Tr}(x) = x + x^{p^{2m}} + x^{p^{4m}} \tag{2}$$

次のような群を構成する．

$$< \{\mathrm{Tr}(g), \mathrm{Tr}(g^2), \cdots, \mathrm{Tr}(g^t) = 3\}, \circ > \tag{3}$$

ここで演算$\circ$は，$\mathrm{Tr}(q^i) \circ \mathrm{Tr}(q^j) = \mathrm{Tr}(g^{i+j})$というように，式(1)に示した乗法群の演算$\cdot$を踏襲するような形で理解すればよい．これを用いて，**2.3**に紹介するXTR-DH鍵交換などを構成することとなる．Lenstraら[2]によって提案されたXTRは，定義体$F_{p^{6m}}$に対して$m=1$としており，これをLimら[8]が$F_{p^{6m}}$に拡張してXTRが実現できることを示した．以降では，表記の簡単化のために$\mathrm{Tr}(g^n) = c_n$とする．XTRでは以下の関係式を駆使して演算$\circ$を実装する．

$$c_0 = 3 \tag{4a}$$

$$c_1 = \mathrm{Tr}(g) \tag{4b}$$

$$c_{n+2} = c_1 \cdot c_{n+1} - c_1^{p^m} \cdot c_n + c_{n-1} \tag{4c}$$

$$c_{2n} = c_n^2 - 2c_n^{p^m} \tag{4d}$$

$$c_{2n-1} = c_{n-1} \cdot c_n - c_1^{p^m} \cdot c_n^{p^m} + c_{n+1}^{p^m} \tag{4e}$$

$$c_{2n+1} = c_{n+1} \cdot c_n - c_1 \cdot c_n^{p^m} + c_{n-1}^{p^m} \tag{4f}$$

$$\tag{4g}$$

以下に，$c_r = \mathrm{Tr}(g^r)$の計算アルゴリズムを示す．

【 $c_r = \mathrm{Tr}(g^r)$ の計算アルゴリズム 】

INPUT： $c_1 = \mathrm{Tr}(g)$，整数r

OUTPUT： $c_r = \mathrm{Tr}(g^r)$

Step1： rが奇数のとき$n=r$，偶数のとき$n=r-1$とする．

Step2： $h=(n-1)/2$とし，$\bar{S}_k = (c_{2k}, c_{2k+1}, c_{2k+2})$とおく．

Step3： 式(4)を用いて，$\bar{S}_1 = (c_2, c_3, c_4)$を計算する．

[1] 真部分体には属さない元のことを真性元と呼ぶこととする．

Step4：h を以下のように2進表現する．ここで，$h_i \in \{0,1\}$, $l-1 \geq i \geq 0$ である．

$$h = h_0 + h_1 2 + h_2 2^2 + h_3 2^3 + \cdots + h_l 2^l, \quad h_l = 1 \tag{5}$$

Step5：h_l を除く $h_{l-1}, \cdots h_0$ に対し，上位ビット h_{l-1} から h_0 の順に以下の操作を繰り返す．ここで，初期値は $k=1$, $\bar{S}_1 = (c_2, c_3, c_4)$ とする．この繰り返しにより $S_n = \bar{S}_k$ となり，S_n が求まる．

- ビットが1のとき，式(4)を用いて $\bar{S}_k$ から $\bar{S}_{2k}$ を求め，$2k$ を k で置き換える．
- ビットが0のとき，式(4)を用いて $\bar{S}_k$ から $\bar{S}_{2k+1}$ を求め，$2k+1$ を k で置き換える．

Step6：r が偶数の場合には，式(4)を用いて S_n から S_{n+1} を計算し，$n+1$ を n で置き換える．

Step7：$S_r = (c_{r-1}, c_r, c_{r+1})$ となり，c_r を得る． ■

アルゴリズムから分かるように，XTR を用いた暗号においては，式(4)の計算を多用することとなる．このことは，すなわち定義体としては，乗算，フロベニアス写像を高速に計算処理できるものが有効であることを意味する．ここで，式(4)の計算は $F_{p^{6m}}$ ではなく，$F_{p^{2m}}$ 上で行われることに注意する．

2.2 トレース表現における離散対数問題

いわゆる有限体上の離散対数問題（Discrete Logarithm Problem;DLP）は，次式のような関係に対して，

$$a = g^x, \quad a, g \in F \tag{6}$$

a, g の情報のみから，指数部 x を求めることが困難であるという問題である．計算機を用いてもなお困難な問題とするために，有限体 F の位数は非常に大きくなければならない[2]．これに対して，XTR ではトレース表現における離散対数問題（XTR-DLP）を次式のように考える．

$$a = \mathrm{Tr}(g^x), \quad g \in F_{p^{6m}}, \ a \in F_{p^{2m}} \tag{7}$$

このような関係を満たす a および $\mathrm{Tr}(g)$ から，その指数部 s を求めることはやはり困難な問題であり，Lenstra ら [2] によって，XTR-DLP と DLP の難しさは同等であることが示されている[3]．それに加え XTR では，トレース関数を用いて部分体 $F_{p^{2m}}$ の元として暗号鍵を構成するため，DLP に基づく暗号方式よりも鍵長が 1/3 程度と短くて済み，さらに先に示した計算アルゴリズムを用いて部分体 $F_{p^{2m}}$ 上で暗号化/復号処理の計算などが行われることから，高速な暗号化/復号処理が実現できることとなる．

2.3 XTR-DH 鍵交換

ここでは，簡単のために XTR を用いた場合の Diffie-Hellman（DH）鍵交換について示す．

【 XTR-DH 鍵交換 】

公開鍵：定義体 $F_{p^{6m}}$ および $\mathrm{Tr}(g)$, $g \in F_{p^{6m}}$

[2] 暗号に用いる場合には，安全性を確保するために，他にも幾つかの条件が課せられる．
[3] XTR-DLP および DLP それぞれの生成元 g が属する有限体が同程度の位数であるとする．

目的：Alice と Bob の間で鍵を共有する．

Step1：Alice は乱数 a を用いて $\mathrm{Tr}(g^a)$ を計算して Bob に送信する．

Step2：Bob は乱数 b を用いて $\mathrm{Tr}(g^b)$ を計算して Alice に送信する．

Step3：Alice，Bob は，それぞれにおいて $\mathrm{Tr}(g^{ab})$ を計算し，これを用いて鍵を共有する． ■

有限体上の DLP を用いた通常の DH 鍵交換と比べて，XTR-DH 鍵交換ではやり取りする鍵データが短くて済み，それぞれの計算処理についても楕円曲線暗号よりも高速であると言われる [2]．

2.4 XTR に用いる拡大体

Lenstra ら [2] が提案した XTR は，定義体を F_{p^6} とし，その部分体 F_{p^2} を用いて鍵の構成，暗号化/復号計算処理を行うというものである．これを Lim ら [8] が $F_{p^{6m}}$ に拡張して XTR を実現できることを示した．拡張したことの最大の効果は，選択できるパラメータが標数 p および拡大次数 m の 2 つになることにある．XTR を用いた暗号の安全性を確保するためには，定義体である $F_{p^{6m}}$ の位数 p^{6m} が 1024 ビット以上でなければならない[4]．すなわち，これを確保するための標数 p のビット数と，拡大次数 m の関係は $6m\log p \geq 1024$ となり，これを満たす組み合わせとして標数と拡大次数を選ぶことができる．

拡大次数 m	1	2	3	4	5	6	7	8	9	10	11	12
標数 p のビット数 $\log p$	170	85	57	43	34	28	24	21	19	17	16	14

表 1: $6m\log p \approx 1024$ となる標数 p と拡大次数 m の組み合わせ

例えば $6m\log p$ が 1024 になるような標数 p と拡大次数 m の組み合わせは，おおよそ表 1 のようになる．**3** において具体的に説明するが，パソコンやマイコンなどのように，ワードサイズが 16 ビットや 32 ビットなど決まっている計算機で XTR を用いた暗号を実装する場合，その有限体の標数 p がワードサイズ未満であるのは，プログラムの高速化やコンパクト実装の面から都合がよい．言い換えれば，用いる計算機のワードサイズに合わせて標数を設定し，その上で十分な安全性を確保できるように拡大次数を決定すればよい．そのような工夫に基づきながら，定義体となる拡大体 $F_{p^{6m}}$ を高速かつコンパクトに実装することが，XTR を用いた暗号の高速実装につながる．

3 XTR を用いた暗号の高速実装

本章では，乗算，フロベニアス写像を高速に処理できる拡大体として，OEF，AOPF，および TypeII AOPF を紹介し，XTR に用いるという観点からこれらを比較する．

3.1 OEF（Optimal Extension Field）

Bailey ら [3] によって提案された OEF は，次のように定義される拡大体 F_{p^m} である．

[4] 2004 年 11 月の時点でそのように言われている．

標数 p : n が計算機のワードサイズ未満である擬メルセンヌ素数 $p = 2^n \pm c$, $\lfloor n/2 \rfloor \geq \log_2 c$

法多項式 : 2 次既約多項式 $x^m - s$, $s \in F_p$

基底 : 法多項式の零点 ω による擬多項式基底 :

$$\{1, \omega, \omega^2, \cdots, \omega^{m-1}\} \tag{8}$$

このような標数 p および法多項式を選ぶことは，それぞれ素体における乗算の後の剰余演算および拡大体における多項式乗算の後の多項式剰余演算を高速に行うための工夫である．とくに，標数 p がメルセンヌ素数である場合を TypeI OEF，法多項式が $x^m - 2$ である場合を TypeII OEF と呼び，さらに高速な実装を実現することができる．OEF の高速実装に関する詳細については文献 [3] を参照していただきたいが，OEF における多項式乗算には Karatsuba 法を応用して高速化を図っている．法多項式に対する条件として，拡大次数 m のすべての素因数は $p-1$ を割り切らなければならない．また，フロベニアス写像には，$m-1$ 回の F_p 上での乗算を必要とする．

3.2 AOPF (All One Polynomial Field)

AOPF は，筆者らがこれまでに提案した拡大体であり，拡大次数は必ず偶数でなければならず，これを $F_{p^{2m}}$ のように表せば，以下のように定義される拡大体である [4].

標数 p : n が計算機のワードサイズ未満である擬メルセンヌ素数 $p = 2^n \pm c$, $\lfloor n/2 \rfloor \geq \log_2 c$

法多項式 : $2m$ 次既約多項式 $(x^{2m+1} - 1)/(x - 1)$

基底 : 法多項式の零点 ω による擬多項式基底 :

$$\{\omega, \omega^2, \cdots, \omega^{2m-1}, \omega^{2m}\} \tag{9a}$$

式 (9a) の擬多項式基底は，式 (9b) に示す正規基底と等価な集合であり，TypeI Optimal Normal Basis (ONB) とも呼ばれ，乗算を高速実装するのに適している．

$$\{\omega, \omega^p, \omega^{p^2}, \cdots, \omega^{p^{2m-1}}\} \tag{9b}$$

また，法多項式 $(x^{2m+1} - 1)/(x - 1)$ が F_p 上で既約となるためには，以下に示す条件が必要十分である．

1 : $2m + 1$ が素数である．

2 : F_{2m+1} において元 p が原始元である．

AOPF における乗算は，以下に紹介する高速乗算アルゴリズム CVMA (Cyclic Vector Multiplication Algorithm) により行う．CVMA では，AOPF の基底である式 (9a) を構成する元 ω が，$(x^{2m+1} - 1)/(x - 1)$ の零点であることから満たす次の 2 つの関係式を駆使し，$F_{p^{2m}}$ の任意元に対する乗算を高速に行う．

$$\omega^{2m+1} = 1, \quad \omega + \omega^2 + \cdots + \omega^{2m} = -1 \tag{10}$$

まず式 (9a) で与えられる擬多項式基底で次のようにベクトル表現される $F_{p^{2m}}$ の 2 元 X, Y に対し,

$$X = (x_1, x_2, \cdots, x_{2m}), \quad Y = (y_1, y_2, \cdots, y_{2m}), \tag{11a}$$
$$x_i, y_i \in F_p, \quad \text{where } 2m \geq i \geq 1$$

これら 2 元 X, Y の積 Z を次式で考えれば,

$$Z = XY = (z_1, z_2, \cdots, z_{2m}), \quad z_i \in F_p, \quad \text{where } 2m \geq i \geq 1 \tag{12}$$

CVMA により, $2m \geq j \geq 0$ として次式の q_j を求め,

$$q_j = \sum_{k=1}^{m} \left\{ \left(x_{\langle 2^{-1}j+k \rangle} - x_{\langle 2^{-1}j-k \rangle} \right) \cdot \left(y_{\langle 2^{-1}j+k \rangle} - y_{\langle 2^{-1}j-k \rangle} \right) \right\}, \tag{13a}$$

Z の各係数 z_i は次式により求められる.

$$z_i = q_0 - q_i, \quad 2m \geq i \geq 1. \tag{13b}$$

Katatsuba 法と異なり, CVMA は式 (13a) のように具体的な計算式に基づくため, 計算コストを拡大次数を用いた定式により与えることができる [4]. なお, 式 (13a) にみられる記号 $\langle \cdot \rangle$ は, $\cdot$ を F_{2m+1} において計算した結果を示す. CVMA の例として, $m = 2$ とした場合について以下に示す.

$$z_1 = q_0 - \{(x_2 - x_4)(y_2 - y_4) + x_1 y_1\} \tag{14a}$$
$$z_2 = q_0 - \{(x_3 - x_4)(y_3 - y_4) + x_2 y_2\} \tag{14b}$$
$$z_3 = q_0 - \{(x_1 - x_2)(y_1 - y_2) + x_3 y_3\} \tag{14c}$$
$$z_4 = q_0 - \{(x_1 - x_3)(y_1 - y_3) + x_4 y_4\} \tag{14d}$$
$$q_0 = (x_1 - x_4)(y_1 - y_4) + (x_2 - x_3)(y_2 - y_3) \tag{14e}$$

CVMA は, 拡大次数が小さい場合にとくに有効であることに加え, 式 (14) から分かるように次に定義される自己相反元に対してとくに有効となる [4]. 後者の有効性については **3.3** で詳しく述べる.

定義 1 $F_{p^{2m}}$ の元 $A = (a_1, a_2, \cdots, a_{2m})$ に対し, $A^* = (a_{2m}, \cdots, a_2, a_1)$ を A の相反元と呼び, $A = A^*$ が成り立つとき A を自己相反元と呼ぶ. ■

なお, CVMA の計算式である式 (13a) と式 (13b) は, $2m$ 次 AOP の零点 ω に対して成り立つ式 (10) のみから導かれ, これが既約であるか否か, すなわち式 (9a) が基底を成しているか否かには依存しないことに注意する [4]. この事実は **3.3** で重要となる. なお, フロベニアス写像には, 正規基底を用いていることからも分かるように, 加算や乗算といった計算は必要としない.

3.3 TypeII AOPF (TypeII All One Polynomial Field)

まず TypeII AOPF が基底として用いる TypeII ONB を紹介して, つづいて TypeII AOPF の定義を与える. そして, **3.2** で紹介した自己相反元と TypeII AOPF の元との関係を示す.

3.3.1 TypeII Optimal Normal Basis

TypeII ONB は，以下のように定義される [7].

定義 2 F_p 上の $2m$ 次 AOP $(x^{2m+1}-1)/(x-1)$ の零点を ω とし，次に示す m 個の元の集合を考える．

$$\{\omega+\omega^{-1}, \omega^2+\omega^{-2}, \cdots, \omega^m+\omega^{-m}\} \tag{15a}$$

このとき，拡大次数 m と標数 p が以下に示す条件 **1** を満たし，これに加えて条件 **2a** あるいは **2b** のいずれかを満たすとき，

1： $2m+1$ が素数

2a： F_{2m+1} において p が原始元

2b： F_{2m+1} において p の位数が m，かつ $2m+1 \equiv 3 \bmod 4$

式 (15a) は F_{p^m} の基底を成すと共に，次に示す正規基底と等価となることから TypeII ONB と呼ばれる．

$$\{\beta, \beta^p, \cdots, \beta^{p^{m-1}}\} \tag{15b}$$

ここで，$\beta = \omega+\omega^{-1}$ である． ■

TypeII ONB に関連して，本論文では次に述べる性質に注目する．まず $2m+1$ が素数であることより，ω は位数 $2m+1$ の元であり，式 (9a) のような相異なる $2m$ 個の元の集合を考えることができる．**3.2** で紹介したように，上述の条件 **1** および **2a** が満たされるときは，この集合は $F_{p^{2m}}$ の擬多項式基底であり，すなわち TypeI ONB を成す．そして，ω に対して成り立つ式 (10) の前者の関係を踏まえれば，

$$\omega^{-1} = \omega^{2m} \tag{16}$$

が成り立ち，条件 **2a** あるいは **2b** のいずれの場合に対しても，式 (9a) に示される元の集合は，その順序も含めて以下の集合と等価となる．

$$\{\omega, \omega^2, \cdots, \omega^m, \omega^{-m}, \cdots, \omega^{-2}, \omega^{-1}\} \tag{17}$$

加えて TypeII ONB は，式 (15b) に示されるように正規基底であり，フロベニアス写像には，AOPF と同様に加算や乗算といった計算は必要としない

3.3.2 TypeII AOPF の定義

TypeII AOPF は，以下のように定義される拡大体 F_{p^m} である．

標数 p： n が計算機のワードサイズ未満である擬メルセンヌ素数 $p = 2^n \pm c$, $\lfloor n/2 \rfloor \geq \log_2 c$

法多項式： $2m$ 次多項式 $(x^{2m+1}-1)/(x-1)$

基底： 法多項式の零点 ω による TypeII ONB：

$$\{\omega+\omega^{-1},\omega^2+\omega^{-2},\cdots,\omega^m+\omega^{-m}\} \tag{18}$$

TypeII AOPF において重要な点は，TypeII ONB を用いていることであり，それに伴って定義 2 の条件が満たされなければならない．また，上記の法多項式は既約である必要はなく，TypeII ONB を構成する元 ω を零点にもつという意味で記している．加えて m が素数ならば，素数次数の拡大体の構成法となる．

3.3.3 自己相反元と TypeII AOPF の元

3.2 でも触れたように，CVMA に対しては必ずしも $2m$ 次 AOP が既約である必要はない．ここで重要なことは，$2m$ 次 AOP の零点 ω により式 (9a) のように与えられる元の集合を用いて，$2m$ 次のベクトル表現を考えることである．ここでは，**3.2** で紹介した自己相反元と TypeII ONB によりベクトル表現される TypeII AOPF の元との関係をみる．まず TypeII AOPF F_{p^m} の任意元 X は式 (18) の TypeII ONB を用いて次式のように表すことができる．

$$X=\sum_{i=1}^{m}x_i(\omega^i+\omega^{-i}),\quad x_i\in F_p,\quad \text{where } m\geq i\geq 1. \tag{19a}$$

上式のように表されることに加え，式 (9a) と式 (17) は順序も含めて等価であり，式 (16) を踏まえて，TypeII AOPF F_{p^m} の任意元が **3.2** で紹介した自己相反元として次式のように取り扱えることが分かる．

$$X=x_1\omega+x_2\omega^2+\cdots+x_m\omega^m+x_m\omega^{m+1}+\cdots+x_2\omega^{2m-1}+x_1\omega^{2m} \tag{19b}$$

なお，定義 2 の条件 **1** および **2a** を満たす場合は，**3.3.1** にも述べたように式 (9a) は TypeI ONB となるということに加え，**3.2** で紹介した AOPF $F_{p^{2m}}$ の基底を成すことから，AOPF $F_{p^{2m}}$ における自己相反元とはその部分体である F_{p^m} の元である [4].

3.3.4 CVMA を改良して用いた TypeII AOPF における乗算

3.3.3 に述べた自己相反元と TypeII AOPF の元との関係を踏まえた上で，式 (10) の関係を用いれば，TypeII ONB を用いた CVMA を考えることができる．式 (19) と同様に TypeII AOPF F_{p^m} の元 Y を考え，

$$Y=\sum_{i=1}^{m}y_i(\omega^i+\omega^{-i}),\quad y_i\in F_p,\quad \text{where } m\geq i\geq 1. \tag{20}$$

2 元 X,Y の積 Z を式 (12) および式 (13) に示す CVMA を用いて考える．まず式 (13a) で q_0 を求めれば，X,Y が式 (19b)，(20) に示すように共に自己相反元であることより $q_0=0$ のようになる．すなわち，q_0 は計算する必要はない．続いて，X,Y が自己相反元であることにより，$m\geq j\geq 1$ として，式 (11) に対して次式が成り立つことから，

$$x_j=x_{\langle -j\rangle}=x_{2m+1-j},\quad y_j=y_{\langle -j\rangle}=y_{2m+1-j} \tag{21}$$

式 (13a) に対し $q_j=q_{\langle -j\rangle}=q_{2m+1-j}$ が成り立つ．これに $q_0=0$ を踏まえ，式 (12) に対して次式を得る．

$$z_j=z_{\langle -j\rangle}=z_{2m+1-j}, \tag{22}$$

すなわち，自明な性質ではあるが，自己相反元である TypeII AOPF F_{p^m} の 2 元 X, Y の積 Z もまた自己相反元となる．この結果，TypeII AOPF F_{p^m} における任意の 2 元 X, Y の積 Z を，CVMA を用いて求める計算式として次式を得る．

$$z_j = \sum_{k=1}^{m} \left\{ \left(x_{\langle 2^{-1}j-k \rangle} - x_{\langle 2^{-1}j+k \rangle} \right) \cdot \left(y_{\langle 2^{-1}j+k \rangle} - y_{\langle 2^{-1}j-k \rangle} \right) \right\}, \quad \text{where } m \geq j \geq 1 \tag{23}$$

ここで $\langle \cdot \rangle$ は，$\cdot$ を F_{2m+1} で計算をした結果であり，式 (23) の計算は，式 (21) の関係を用いて行うことに注意する．TypeII AOPF における CVMA を用いた乗算の一例として，拡大次数を $m = 2$ の場合を考える．2 元 X, Y を以下のように表せば，

$$X = x_1(\omega + \omega^{-1}) + x_2(\omega^2 + \omega^{-2}), \quad Y = y_1(\omega + \omega^{-1}) + y_2(\omega^2 + \omega^{-2}) \tag{24}$$

式 (23) を用いて，その積 Z が次式のように求まる．

$$Z = z_1(\omega + \omega^{-1}) + z_2(\omega^2 + \omega^{-2}) \tag{25a}$$

$$z_1 = (x_1 - x_2)(y_2 - y_1) - x_1 y_1 \tag{25b}$$

$$z_2 = (x_1 - x_2)(y_2 - y_1) - x_2 y_2 \tag{25c}$$

細かいことではあるが，式 (25b) および式 (25c) にみられる $(x_1 - x_2)(y_2 - y_1)$ の計算結果は共用し，それぞれで別途に計算するようなことはしない．

3.4 乗算，フロベニアス写像に要する計算コストの比較

本章では，OEF，AOPF，および TypeII AOPF における乗算，およびフロベニアス写像の計算コストについて，それぞれに必要となる素体上の加算（減算）および乗算の回数という観点から比較する．具体的には 6 次拡大体について比較する．なお，表 2 において () 内の数字は，左から素体上の加算（減算）および乗算の回数である．逐次拡大法については，文献 [9] を参照していただきたい．

表 2: 6 次拡大体 F_{p^6} における乗算およびフロベニアス写像の計算コスト

構成法	法多項式	乗算	フロベニアス写像
TypeII OEF[3]	$x^6 - 2$	(69, 18)	(0, 5)
TypeII AOPF	$(x^{13} - 1)/(x - 1)$	(60, 21)	(0, 0)
AOPF[4]	$(x^7 - 1)/(x - 1)$	(50, 21)	(0, 0)
逐次拡大 OEF[9]	$x^3 - 2 \rightarrow x^2 + x + \omega^{\dagger}$	(64, 18)	(0, 5)
逐次拡大 TypeII AOPF	$(x^7 - 1)/(x - 1) \rightarrow (x^5 - 1)/(x - 1)$	(48, 18)	(0, 0)

$^{\dagger}\omega$ is a zero of $x^3 - 2$ that is the modular polynomial of F_{p^3}.

表 3 は，具体的に標数 p を $2^{24} - 3$ に設定し，それぞれの 6 次拡大体において乗算 1 回にかかる計算処理時間を計測した結果である．CPU には PentiumIII (800MHz) を用いた．これらデータから，XTR を用いた暗号を高速実装する場合には，AOPF や TypeII AOPF が適しているということができる．OEF と AOPF の大きな違いは，その乗算方法が Karatsuba 法か CVMA であるかにあり，それぞれの特徴がある．例えば CVMA であれば，拡大次数が小さい方が性能がよく，また並列実装に適している．詳細については文献 [5] を参照していただきたい．

表 3: F_{p^6} における乗算の計算処理時間の比較

unit:μs

標数	構成法	法多項式	処理時間
$2^{24}-3$	TypeII OEF	x^6-2	1.47
	TypeII AOPF	$(x^{13}-1)/(x-1)$	1.46
	逐次拡大 TypeII OEF	$x^3-2 \rightarrow x^2-\omega^{\dagger}$	1.46
	逐次拡大 TypeII AOPF	$(x^7-1)/(x-1) \rightarrow (x^5-1)/(x-1)$	1.29

$^{\dagger}\omega$ is a zero of x^3-2 that is the modular polynomial of F_{p^3}.

*CPU:PentiumIII, 800MHz

4 まとめ

本論文では，XTR を用いた暗号の高速実装という観点から，従来より知られている拡大体の高速な実装手法である OEF（Optimal Extension Field），AOPF（All One Polynomial Field），TypeII AOPF（TypeII All One Polynomial Field）の性能を，とくに乗算に必要な計算コストについて比較した．

参考文献

[1] A.Menezes, Elliptic Curve Public Key Cryptosystems, Kluwer Academic Publishers, 1993.

[2] A.Lenstra and E.Verheul, "The XTR Public Key System," *Crypto 2000*, LNCS 1880, pp. 1-20, 2000.

[3] D.B.Bailey and C.Paar, "Optimal Extension Fields for Fast Arithmetic in Public-Key Algorithms," Proc. Asiacrypt2000, LNCS 1976, pp.248-258, 2000.

[4] Y.Nogami,A.Saito, and Y.Morikawa, "Finite Extension Field with Modulus of All-One Polynomial and Expression of Its Elements for Fast Arithmetic Operations" The International Conference on Fundamentals of Electronics, Communications and Computer Sciences, pp.10-15(R18), 2002.

[5] S.Shinonaga, Y.Fujii, Y.Nogami, and Y.Morikawa,"TYPE-II All-One Polynomial Field," Symposium on Cryptography and Information Security 2004, pp.377 to 382, 2004.

[6] D.Knuth, The Art of Computer Programming, vol.2, Addison-Wesley, 1981.

[7] I.Blake et al, Elliptic Curves in Cryptography, LNS 265, Cambridge Univ. Press, 1999.

[8] S.Lim et al, "XTR Extended to $GF(p^{6m})$," SAC2001, LNCS2259, pp.301-312, 2001.

[9] T.Kobayashi, K.Aoki, and F.Hoshino, "OEF Using a Successive Extension," Proc. The 2000 Symposium on Cryptography and Information Security, no.B02, 2000, *in Japanese.*