RA-006

Normal Basisを用いた多項式基底間の基底変換

石井 将大† 猪俣 敦夫† 藤川 和利†

概 要

有限体上の算術とその実装は，暗号，符号理論に基づく様々な応用技術において，最も重要なものである．又，有限体の基底変換を用いることにより，その定義体に適した高速な実装が可能となり，更に，有限体上の算術に関して，特別な基底を用いた多くの効率的なアルゴリズムが研究されている．本研究では，同型な有限体における任意の多項式基底間の基底変換に関して，従来の EDF (Equal Degree Factorization) を用いた手法に対して，normal basis への基底変換を用いることによる，計算コストのより小さい手法を提案する．更に，normal basis を成す元の確率的な探索手法に関しても幾つか考察を与える．提案手法により，多項式基底間の基底変換に関して，拡大体の拡大次数 k について概ね $1/k$ のコスト削減を実現した．又，提案アルゴリズムについて，EDF を用いた手法に対し，計算コストの比較を行った．

1 序論

有限体の基底変換は，超楕円曲線暗号やペアリング暗号等，様々な暗号方式の効率的なハードウェア実装において重要な役割を果たす．即ち，有限体上の算術に関して，基底変換を用いる事により，様々な暗号パラメタに対して，高速な実装をより柔軟に実現出来る．

XTR 暗号 [9] においては，効率的な暗号実装のため，特別な定義体と基底を用い計算を行う．更に，基底変換を用いることにより，サイドチャネル攻撃の対策を施すことも考えられている．又，[15] において，η_T ペアリングのアルゴリズム内で基底変換を適用することにより，高速な実装を実現している．暗号のアプリケーションでは基本的に数百，数千ビット等の比較的大きな数を想定しており，その様なサイズの有限体において，与えられた基底間の基底変換の求解には，第一に計算コストの問題が挙げられる．

†奈良先端科学技術大学院大学
〒 630-0192 奈良県生駒市高山町 8916-5

本論文では，有限体の間の基底変換の求解アルゴリズムに関して，normal basis への変換を用いることによる，より効率的な手法を提案する．本論文では，基底変換に関して，任意の有限体の多項式基底間の変換行列を求めることを想定している．その様な状況における従来手法として，EDF (Equal Degree Factorization) [3] を用いたもの [17] があるが，この手法に比べて，提案手法では変換行列の探索時間を大幅に短縮できることを，シミュレーションにより示す．

本論文では任意の有限体の間の基底変換を対象としており，[19] における optimal normal basis を持つ様な有限体，或いは [11] の GNB (Gauss period Normal Basis) を持つ様な特別な有限体の間の基底変換については考慮していない．[13] では，暗号のパラメタとしては支障はない程度だが，やはり特別なパラメタを持つ有限体の効率的な基底変換手法を提案している．これらに対し，本論文では normal basis への変換により，素朴な方法である EDF を用いた変換手法の改良を行う．加えて，与えられた多項式基底に対し，ある normal basis への変換についても，小標数の有限体に対し，より高速に計算できる提案手法を述べる．

以下に本論文の構成を述べる．第 2 節において，基底変換と normal basis に関する先行研究について述べ，本論文における基底変換のターゲットを具体的に説明する．次に，第 3 節において，多項式基底から normal basis への変換，更に，normal basis を用いた多項式基底間の変換行列求解の提案手法を述べる．第 4 節では，本提案手法のシミュレーション結果を述べ，従来手法に対して優位性を示す．又，提案アルゴリズムの計算コストに関して議論する．最後に第 5 節において本論文の結論を述べる．

2 有限体の基底変換とNormal Basis

本節では，有限体の基底変換について先行研究と，基礎的な性質を述べる．第 2.1 節において基底変換の

先行研究について説明し，本論文で対象とする基底変換と記号等を述べ，従来手法の EDF を用いた変換手法について述べる．更に第 2.2 節において，normal basis の生成元の探索等の関連研究と，与えられた多項式基底から，ある normal basis への変換行列の生成法について述べる．

2.1 基底変換

有限体の基底変換を求めるためにはいくつか方法があるが，まず Lenstra [6] により，有限体の自己同型，即ち基底変換を求めるための多項式時間で計算できるアルゴリズムが提案された．後に詳しく述べるが Sunar は [17] において EDF を用いた任意の多項式基底間の変換行列生成アルゴリズムを提案した．それ以前に，基底変換行列の構成法として Paar [14] による提案手法が挙げられていたが，その手法は原始元に関して，ある拡大体において全数探索を行う必要があり，拡大体のサイズが大きい時は計算時間が現実的でなくなる．又，Paar は合成体において変換行列生成に関する手法について提案しているが，Sunar が提案した手法は任意の有限体に適用できる．

本論文では有限体の基底変換として，拡大体 $\mathbb{F}_{p^k}$ の多項式基底の間の基底変換を考える．ここで，$\mathbb{F}_p$ は素体であり，議論と記述の簡単のため素体上の単拡大について考える．但し，本論文で述べる基底変換求解の手法は，任意の拡大体に適用可能で，一般性を失うものではない．

本論文では数ある基底の内，多項式基底の間の基底変換について考察する．暗号実装において，既約多項式による拡大体の構成と，その拡大体上の多項式基底による算術は最も基本的なものである．$\mathbb{F}_{p^k}$ の二つの基底 $\mathcal{B}, \mathcal{B}'$ の間の基底変換について考える．$\mathbb{F}_{p^k{}_{\mathcal{B}}}$ は，自身を構成する既約多項式の根を，素体に添加して得られ，その根が基底 $\mathcal{B}$ を成している．この時，そのそれぞれの既約多項式の根を $\mathbb{F}_{p^k{}_{\mathcal{B}'}}$ 内で求めること，即ち，その基底 $\mathcal{B}$ を成す根を $\mathcal{B}'$ で表現することにより，基底 $\mathcal{B}$ の元の $\mathcal{B}'$ の表現を得る．

次に EDF を用いた多項式基底間の変換行列の生成法について述べる．特に binary field 上の多項式基底間の基底変換求解に関して，Sunar [17] による EDF を用いた手法がある．この手法は同様にして任意の正標数の場合でも適用できる．EDF は多項式の既約分解を求めるアルゴリズムであり，EDF を用いて，拡大体を構成する定義多項式の一次の既約式，即ち定義多項式の根を直接求めることにより基底変換を行う．EDF により定義多項式の根を求め，既定変換を得る方法を Algorithm 1 に示す．ここで基底変換は

$$\mathbb{F}_{p^k{}_{\mathcal{B}}} \simeq \mathbb{F}_p[x]/(f(x)) \leftrightarrow \mathbb{F}_{p^k{}_{\mathcal{B}'}} \simeq \mathbb{F}_p[x]/(g(x))$$

について考える．

Algorithm 1 EDF による多項式基底間の基底変換

INPUT: 既約多項式 f, g による $\mathbb{F}_{p^k}$ の多項式基底 $\mathcal{B}, \mathcal{B}'$
OUTPUT: 基底変換 $T\colon \mathbb{F}_{p^k{}_{\mathcal{B}}} \to \mathbb{F}_{p^k{}_{\mathcal{B}'}}$

1: $F(x) := f(x)$
2: **while** $\deg(F(x)) \neq 1$ **do**
3: ランダムに $A(x) \in \mathbb{F}_{p^k{}_{\mathcal{B}'}}[x]/(f(x))$ を取る
4:
5: **trace の計算**
6: $T(x) = A(x) + A(x)^{\sigma} + \cdots + A(x)^{\sigma^{k-1}}$
7:
8: $F(x) = (F(x), T(x))$
9: **if** $F(x) = 1$ **then**
10: $F(x) = f(x)$
11: **end if**
12: **end while**
13: $f(x)$ の根 $\theta_{\mathcal{B}'}$ を得る
14: $(1\ \theta_{\mathcal{B}'}\ \cdots\ {\theta_{\mathcal{B}'}}^{k-1}) = \mathcal{B}'T$ なる変換行列 T を計算する

提案手法では，normal basis への変換を用いて，この EDF による基底変換アルゴリズムの改良を行う．

2.2 Normal Basis への変換

まず，有限体論より，拡大体の元の組が基底を成すときの必要十分条件がある．q は標数 p の冪とする．

補題 1 ([10, Corollary 2.38])．$\alpha_1, ..., \alpha_k \in \mathbb{F}_{q^k}$ に対し，$\{\alpha_1, ..., \alpha_k\}$ が $\mathbb{F}_{q^k}$ の $\mathbb{F}_q$ 上の基底であるためには，

$$\begin{vmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_k \\ \alpha_1^q & \alpha_2^q & \cdots & \alpha_k^q \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{q^{k-1}} & \alpha_2^{q^{k-1}} & \cdots & \alpha_k^{q^{k-1}} \end{vmatrix} \neq 0 \qquad (1)$$

を満たすことが必要十分である．

拡大体 $\mathbb{F}_{q^k}/\mathbb{F}_q$ の基底 $\mathcal{B}$ が，ある元 $a \in \mathbb{F}_{q^k}$ と Frobenius map $\sigma \in \mathrm{Aut}_{\mathbb{F}_q}(\mathbb{F}_{q^k})$ により

$$\mathcal{N} = \{a, \sigma(a), ..., \sigma^{k-1}(a)\}$$

と書けるとき，$\mathcal{N}$ を normal basis と呼ぶ．ここで Frobenius map は $\sigma(a) = a^p$（p は標数）なる写像（或いはこの σ を有限回合成したもの）であり，数学上重要な概念で，上記の通り有限体において自己同型

群を成す．任意の有限体の有限次拡大体に対し normal basis は存在する．

式 (1) より，

$$\begin{vmatrix} a_{\mathcal{B}} & \sigma(a_{\mathcal{B}}) & \cdots & \sigma^{k-1}(a_{\mathcal{B}}) \\ \sigma(a_{\mathcal{B}}) & \sigma^2(a_{\mathcal{B}}) & \cdots & a_{\mathcal{B}} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma^{k-1}(a_{\mathcal{B}}) & a_{\mathcal{B}} & \cdots & \sigma^{k-2}(a_{\mathcal{B}}) \end{vmatrix} \neq 0 \quad (2)$$

を満たす元 $a_{\mathcal{B}}$ を見つければ

$$\mathcal{N} = \{a_{\mathcal{B}}, \sigma(a_{\mathcal{B}}), ..., \sigma^{k-1}(a_{\mathcal{B}})\}$$

は $\mathbb{F}_{q^k\mathcal{B}}$ において normal basis を成す．式 (2) の行列式の値は，$\mathbb{F}_{q^k\mathcal{B}}$ 上の多項式

$$f(x) = x^k - 1 \quad (3)$$
$$g(x) = a_{\mathcal{B}}x^{k-1} + a_{\mathcal{B}}^q x^{k-2} + \cdots + a_{\mathcal{B}}^{q^{k-2}} x + a_{\mathcal{B}}^{q^{k-1}}$$

における終結式 $\mathrm{Res}(f(x), g(x))$ と一致するから ([10, Theorem 2.39]), $\mathcal{N}$ が normal であることと，$f(x)$ と $g(x)$ が互いに素である（共通根を持たない）ことは同値である．

以上をまとめると，拡大体 $\mathbb{F}_{q^k\mathcal{B}}$ に対し，式 (3) の $f(x), g(x)$ に対し $(f(x), g(x)) = 1$ なる $a_{\mathcal{B}}$ は normal basis $\mathcal{N}$ を成し，

$$\mathcal{N} = (a_{\mathcal{B}}, \sigma(a_{\mathcal{B}}) \cdots \sigma^{k-1}(a_{\mathcal{B}})) = \mathcal{B}P$$

として基底変換 $P\colon \mathcal{N} \to \mathcal{B}$ が求まる．

Normal basis の生成元の探索に関して，ONB 等の特別な有限体に存在する normal basis を用いて行う手法があり，体の元の位数による特徴付け [19] により，上記の GCD を計算する手法に比べて，生成元の効率的な探索が可能となる．又，根が normal basis を成す既約多項式を N-polynomial と呼ぶが，normal basis と N-polynomial は一対一に対応しているので N-polynomial に関する研究も行われている．N-polynomial の特徴付けは [4] に詳しくあるが，[18] により normality の判定は式 (3) の $f(x)$ の既約因子が重要な役割を果たす．従って，上記の様に $x^n - 1$ の根を調べることは意義がある．[7, 8] ではある N-polynomial から始め，新たに N-polynomial を生成する手順を与えている．又，[2] において，既約多項式に対して，normal 性，即ち N-polynomial であるかどうかをテストするアルゴリズムを提案している．本論文では，与えられた多項式基底に対して normal basis への変換を考慮しており，N-polynomial の探索による変換は適用できない．

野上らは [11] において，GNB (Gauss period normal basis) という normal basis の中でも特別なものを用いて，この基底を経由して与えられた基底の間の変換を求める手法を提案した．一般に，二つの基底間の基底変換をそれぞれの基底と，normal basis との変換を求め，それらの変換を合成して求める方法が考えらる．但し，normal basis を生成する拡大体の元は多く存在し，それぞれの基底に対して求めた normal basis が基底として一致していなければ，基底変換を合成できない．野上らは，より特別な normal basis を用いることにより，この問題を解決している．GNB については，一度その生成元を見つけると，他の生成元を見つけたとしてもそれぞれの生成元が互いに共役（Frobenius map で写り合う）であるから，同じ normal basis を生成する．従って，GNB の生成元を一つ見つければ，GNB を経由して基底変換を求めることが可能である．加えて，GNB の生成元は拡大体上の元として，その位数によって特徴付けられ，探索をより効率的に行うことが出来る．又，ONB や AOP (All-One Polynomial) [12] が持つ同様の利点から，難波らは [19] において EDF を用いた手法と比較して，変換行列の高速な生成法を提案している．更に [13] では，GNB が存在しない場合を考慮し，ある位数の小さい元と多項式基底を用いることから，より一般の拡大体に対し，加えてより高速に変換行列を求める手法を提案している．但し，暗号のパラメタとしては支障はない程度だが，特別な有限体に限定される．

上記の手法 [13] や，GNB を持つ拡大体には，標数，拡大次数に制限があるため，本論文では GNB の様な特別な基底を用いることは考えず，先に述べた EDF を用いたアルゴリズムの改良を行う．

3 Normal Basis を用いた多項式基底間の基底変換

第 2.1 節で述べた様に，本論文では任意の多項式基底間の基底変換行列を求めるため，EDF による手法を採用し，それを normal basis への変換を用いることにより改良する．まず第 3.1 節で多項式基底から normal basis への変換行列の生成法について述べ，第 3.2 節において，多項式基底間の基底変換について提案手法を述べる．

3.1 多項式基底と Normal Basis との間の基底変換

初めに，与えられた多項式に対して，その基底とある normal basis との変換を求めるアルゴリズムについて考察する．ここでは，補題 1 を用いた，normal basis の生成元を判定する標準的な手法としての GCD 計算の代わりに，式 (3) の $f(x), g(x)$ が共通根を持つかどうか直接確かめる方法について考察する．まず，$f(x) = x^k - 1$ の根の探索について考える．位数 $(k, p^k - 1)$ の元が成す巡回群を直接求める方法により，$f(x)$ の根を求めることが出来る．

$f(x) = x^k - 1$ の根について次のことが言える．

- $k \nmid (p^k - 1)$ のとき．
  - $(k, p^k - 1) = 1$ のとき，位数が k の約数である元は 1 のみ．よって，$x^k - 1$ の根は 1 のみ．
  - $(k, p^k - 1) = d > 1$ のとき，拡大体の元 $a \in \mathbb{F}_{p^k}$ で $(a^{(p^k-1)/d})^m \neq 1$, $(1 \leq m \leq d-1)$ なるものを確率的に探す．この様な a は位数 d の巡回群をなし，$x^k - 1$ の $\mathbb{F}_{p^k}$ における根をすべて含む．
- $k \mid (p^k - 1)$ のとき，$c = (p^k - 1)/k$ として，$(a^c)^m \neq 1$ $(1 \leq m \leq k-1)$ なる元 $a \in \mathbb{F}_{p^k}$ は位数 k の巡回群を生成する．このとき，その巡回群は，$x^k - 1$ の $\mathbb{F}_{p^k}$ における根をすべて含む．

これにより，$f(x)$ の根の候補を調べあげ，$g(x)$ と共通根を持たないような $a_{\mathcal{B}}$ の探索を試みる．ここで，$f(x)$ の根は拡大体 $\mathbb{F}_{p^k}$ において，完全に分離するわけではないことに注意しておく．$k \mid p^k - 1$ のとき，$f(x)$ は $\mathbb{F}_{p^k}$ に根を k 個持つので，$f(x)$ と $g(x)$ が共通根を持たないことと，$(f(x), g(x)) = 1$ を満たすことは同値である．しかし，$k \nmid p^k - 1$ のとき，$f(x)$ は全ての根を $\mathbb{F}_{p^k}$ に持つわけではないため，$f(x)$ が $\mathbb{F}_{p^k}$ において，$g(x)$ と共通根を持たなくても $f(x)$ と $g(x)$ がと共通因子を持たないとは限らない．従って，この手法で $a_{\mathcal{B}}$ を求めたとき，$k \nmid p^k - 1$ の場合ではいつも変換行列が生成できるとは限らない．本論文では，変換行列として正則行列が得られない場合を考慮し，どの程度の失敗が含まれるかシミュレーションにより確かめた．

3.2 多項式基底間の基底変換に関する提案手法

次に，任意の多項式基底間の基底変換について，提案手法を述べる．本論文では，第 2.1 節で述べた様に，任意の拡大体のパラメタを許容出来るように，特別な GNB 等の基底を用いず，EDF を用いたアルゴリズムを使用する．

従来手法の EDF を用いた基底変換の求解アルゴリズムは，有限体として binary field を想定し，効率的なハードウェア実装が実現出来るものだった．本稿では任意の有限体を想定しており，この時，Algorithm 1 において，EDF による基底変換は step 6 の trace の計算コストが拡大体のサイズに対し急激に増加する．そこで，本提案手法では，normal basis への基底変換用いて trace の計算を効率的に行い，変換行列の探索の高速化を実現した．Trace 計算は Frobenius map の作用が何度も行われるため，normal basis で有限体の元を表すことにより Frobenius map の作用の計算量を削減することで，高速化が見込める．具体的な手法を Algorithm 2 に示す．Trace 計算以外のフェーズについては，比較的計算量も小さく，改善すべき点はアルゴリズム自体を変更しない限り見当たらない．

Algorithm 2 Normal basis を用いた EDF による多項式基底間の基底変換

INPUT: 既約多項式 f, g による $\mathbb{F}_{p^k}$ の多項式基底 $\mathcal{B}, \mathcal{B}'$
OUTPUT: 基底変換 $T\colon \mathbb{F}_{p^k\mathcal{B}} \to \mathbb{F}_{p^k\mathcal{B}'}$

1: **normal basis への基底変換 $Q\colon \mathcal{B}' \to \mathcal{N}$ を求める**
2: $F(x) = f(x)$
3: **while** $\deg(F(x)) \neq 1$ **do**
4: ランダムに $A(x) \in \mathbb{F}_{p^k\mathcal{B}'}[x]/(f(x))$ を取る
5:
6: **$A(x)_{\mathcal{N}}$ を計算する（基底変換）**
7: trace の計算
8: $T(x) = A(x)_{\mathcal{N}} + A(x)^{\sigma}_{\mathcal{N}} + \cdots + A(x)^{\sigma^{k-1}}_{\mathcal{N}}$
9: **$T(x)_{\mathcal{B}'}$ を計算する（基底変換）**
10:
11: $F(x) = (F(x), T(x))$
12: **if** $F(x) = 1$ **then**
13: $F(x) = f(x)$
14: **end if**
15: **end while**
16: $f(x)$ の根 $\theta_{\mathcal{B}'}$ を得る
17: $(1, \theta_{\mathcal{B}'} \cdots {\theta_{\mathcal{B}'}}^{k-1}) = \mathcal{B}'T$ なる変換行列 T を計算する

提案手法では，step 1 にある様に，$\mathcal{B}'$ と，ある normal basis $\mathcal{N}$ との間の基底変換を求めておき，$A(x)_{\mathcal{N}}$ として $A(x)$ の各係数を $\mathcal{N}$ を用いて表現することにより (step 6), Frobenius map による作用がシフト演算で行えることから，step 8 の拡大体上での trace 計

算に関して，高速化が可能である．ここで，$A(x)^{\sigma}$ 等の計算は，多項式 $A(x)$ の係数に σ を作用させることで行えることに注意しておく．Trace 計算の後に，$T(x)_{\mathcal{B}'}$ として trace の各係数を基底 $\mathcal{B}'$ で表し，この基底変換を適用した trace 計算を繰り返し続ける．後にシミュレーション結果として示すが，全体として基底変換に掛かる時間より，normal basis に変換するコストを加えても，効率的に trace 計算を行う方が高速である．

4 シミュレーションによる評価

本節では，第 3.1, 3.2 節で述べた基底変換に関する提案手法に対して，シミュレーション結果を示し，幾つか考察を与える．シミュレーションは NTL [16] を用いた C++によって実装した．以下の表 1 に，基底変換に関するシミュレーション環境を示す．

OS	Fedora 17
CPU	Intel Core i7-3960X, 3.30GHz, 6 Cores
Memory	DDR3-1333, 64GB
Compiler	GCC-4.5.3
Language	C++ with NTL-5.5.2

表 1: 実装環境

4.1 Normal Basis への基底変換

第 3.1 節では，normal basis を成す元の探索について，GCD 計算と，具体的に多項式の共通根を計算する方法を述べた．ここではそれらの手法を用いた normal basis への基底変換求解のシミュレーション結果を示す．

シミュレーションでは比較のため次の 3 つの方法による，多項式基底と normal basis との基底変換について実装を行った．

拡大体 $\mathbb{F}_{p^k}{}_{\mathcal{B}}$ において，$\mathcal{B}$ から，ある normal basis への基底変換について，以下の 1 から 3 の手法で normal basis の生成元の探索を行う．

1. **GCD:** $(f(x), g(x)) = 1$ を満たす $a_{\mathcal{B}}$ を確率的に探索する方法．
2. **no common root:** $\mathbb{F}_{p^k}$ における $f(x)$ の根を求めて，$g(x)$ と共通根を持たないような $a_{\mathcal{B}}$ を確率的に探索する方法．求めた変換行列が正則であるとは限らない．
3. **GCD + no common root:** 上の no common root の方法において，$k \nmid p^k - 1$ の時は GCD 計算によるもので normal basis の生成元を求めるもの．これにより変換行列の生成に失敗することはない．

図 1 において，標数 3 の時の，多項式基底と normal basis との基底変換求解に掛かる実行時間に関するシミュレーション結果を示す．横軸は拡大次数を取り，縦軸は変換行列求解の計算時間を表す．小標数においては，提案手法である $f(x)$ と $g(x)$ の共通根をチェックする手法が GCD に比べ高速である．ここで，計算時間が 0 を取っている拡大次数が 90 と 105 付近に二つあるが，これは変換行列として計算した行列が正則ではなかった場合である．しかし，小標数の時でさえも，変換行列生成の失敗が起こる場合は少なく，ほとんどの拡大次数において，$k \nmid p^{k-1}$ の時であっても変換行列を求めることが出来ている．

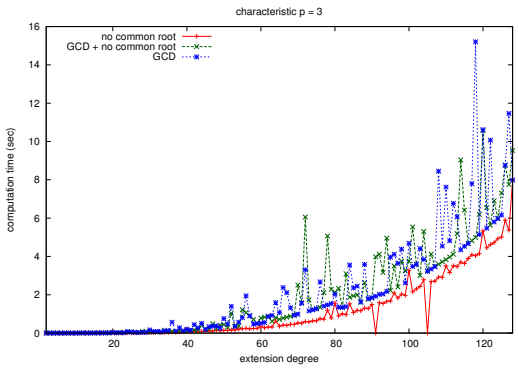

図 1: $\mathbb{F}_{p^k}$ $(p = 3)$ の多項式基底と normal basis の間の基底変換求解に掛かる時間

次に，図 2 において，標数が 128-bits prime の時の，多項式基底と normal basis との基底変換求解に掛かる実行時間に関するシミュレーション結果を示す．同様に横軸に拡大次数を取り，縦軸は基底変換求解の計算時間を取る．この時は，安定して GCD による求解が高速であり，提案手法はそれぞれの拡大次数において，約 10 倍程度 GCD に比べ解の探索に時間が掛かっている．$(k, p^k - 1) = 1$ のときは，$f(x)$ の根は 1 しか無いため，GCD による手法より高速である．又，標数が少しでも大きいと，変換行列生成が失敗することはまず起こらなくなる．

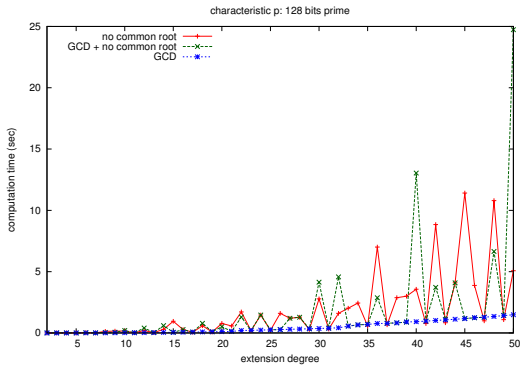

図 2: $\mathbb{F}_{p^k}$ p: 128-bits prime の多項式基底と normal basis の間の基底変換求解に掛かる時間

4.2 多項式基底間の基底変換

次に，多項式基底間の基底変換の求解についてシミュレーション結果を示す．以下のそれぞれの図は，横軸に拡大次数，縦軸に変換行列の生成時間を取っている．図 3 において，小標数 $p = 3$ の時のシミュレーション結果を示した．シミュレーション結果によると，提案手法と既存手法に差異はない．小標数の場合では，素体の元同士の演算コストが小さいため，trace 計算において提案手法における normal basis を用いた Frobenius map の作用の計算量削減効果が大きくないためである．

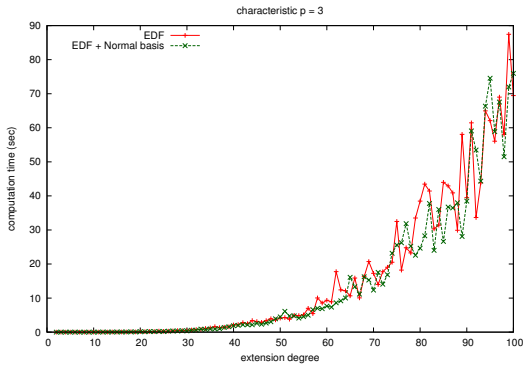

図 3: $\mathbb{F}_{p^k}$ $(p = 3)$ の多項式基底間の基底変換求解に掛かる時間

最後に，図 4 において，標数が 16-bits prime の時の多項式基底間の基底変換求解に関するシミュレーション結果を示す．標数が大きい程，既存手法では拡大次数の増加により急激に求解に掛かる計算時間が増加するが，提案手法においては，増加度合いも既存手法に比べ緩やかで，大きなサイズの $\mathbb{F}_{p^k}$ においても現実的な計算コストで基底変換を求めることが出来る．小標数の場合とは違い，trace 計算のコスト削減が達成されていることが分かる．

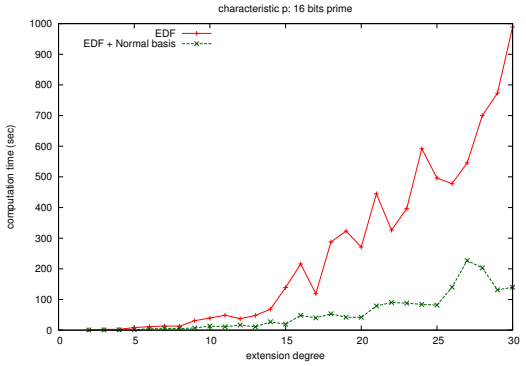

図 4: $\mathbb{F}_{p^k}$ p: 16-bits prime の多項式基底間の基底変換求解に掛かる時間

4.3 考察

ここでは，第 3.1, 3.2 節で述べたそれぞれの基底変換手法に関して，アルゴリズムの計算量の比較と考察を行う．計算量の基礎単位として，素体 $\mathbb{F}_p$ 上の乗算コストを用い，各アルゴリズムの計算量を $\mathbb{F}_p$ 上の乗算回数として表す．更に，本論文では拡大体 $\mathbb{F}_{p^k}$ 上の乗算は $O(k^2)$ で行うものとする．

4.3.1 Normal Basis を成す元の探索

まず，式 (3) の多項式 $f(x), g(x)$ に対して，GCD: $(f(x), g(x))$ は $O(k^2 \log k)$ で計算出来る．これに対し f と g の共通根を直接確かめる方法について比較する．ここでは，f の根 θ を見つけるコストは考慮せず，$g(\theta^i)$ の計算コストについて比較する．

$k \mid (p^k - 1)$ のとき，f の根は $\{\theta^i \mid 1 \leq i \leq k\}$ で k 個あり，計算量は最大になる．本実装では，まず根の集合を得るために，θ の冪を求め，その計算量は $O(k^3)$ である．更に，$g(\theta^i)$ $(\leq i \leq k)$ の計算では，それぞれの i に対し計算しておいた根を用いて，$O(k^3)$ で計算でき，全ての根に対して $g(\theta^i)$ の値を計算するとき計算量は $O(k^4)$ で表される．従って，全体として計算量は $O(k^4)$ となり，GCD 計算に比べ，特に p

が大きい時大幅に計算時間が増加する．但し，シミュレーション結果で示したように，$(k, p^k - 1) = 1$ の時は，$f(x)$ の根は 1 しか無いため，確率的ではあるが normal basis を成す元の探索も多くの場合で成功し，更に，共通根のチェックは

$$g(1) = a_{\mathcal{B}} + a_{\mathcal{B}}^{q} + \cdots + a_{\mathcal{B}}^{q^{k-2}} + a_{\mathcal{B}}^{q^{k-1}} = \mathrm{trace}(a)$$

として一度の trace 計算で済むため，計算量はより小さい．

4.3.2 EDF による多項式基底間の基底変換

最後に，Algorithm 1,2 の計算量を比較する．特に trace の計算と normal basis への基底変換の計算コストについて考える．ここでは，従来の EDF による手法の計算量に対して，normal basis への変換を用いることによる本提案手法の計算量が，どの程度削減出来ているか議論するため，計算量をオーダ記法でなく具体的に書く．

Algorithm 1 の step 6 の trace 計算に関して，

$$A(x) = \sum_{i=0}^{k-1} a_i x^i \quad (a_i \in \mathbb{F}_{p^k}[x])$$

とし，$A(x)^\sigma$ について考える．まず，全ての a_i^σ の計算にコスト $k^3 \log p$ 掛かる．更に，本実装では σ による作用の変換行列 M を

$$(1 \ x \ \cdots \ x^{k-1})^\sigma = (1 \ x \ \cdots \ x^{k-1})M$$

として計算しておき，

$$A(x)^\sigma = (1 \ x \ \cdots \ x^{k-1})M \ {}^t(a_0^\sigma \ a_1^\sigma \ \cdots \ a_{k-1}^\sigma)$$

を計算している．よって，行列 M の積がコスト k^3 で計算され，従って $A(x)^{\sigma^i}$ $(1 \le i \le k-1)$ がコスト $(k-1)(k^3 + k^3 \log p)$ で計算できる．

次に Algorithm 2 について，normal basis へ変換した後の trace 計算については各 a_i^σ の計算量を無視出来るため，上の議論からコスト $k^3(k-1)$ で計算できる．ここで，normal basis を成す元の探索については，$g(x)$ の係数の計算コストは $(k-1)k^2 \log p$ で，GCD 計算のコストは $k^2 \log k$ である．但し，normal basis への基底変換は，アルゴリズムの始めに一度求めておけば良い．更に，Algorithm 2 の step 6 において，$A(x)$ の各係数に基底変換行列を掛けなければならず，コスト k^3 で計算される．

従って，Algorithm 2 については，始めの normal basis への変換コストが

$$(k-1)k^2 \log p + k^2 \log k$$

であり，探索フェーズの step 3-14 に関しては拡大体の元に対する基底変換の計算コスト k^3 が余計に加算される．しかし，それらは単純な EDF による手法の計算量について，最大の因子である trace 計算のコスト

$$(k-1)(k^3 \log p)$$

と比べ，オーダの面からも計算コストがより小さいことが分かる．又，$\log p$ の影響と素体上の乗算コストにより，標数 p がより大きい方が，提案手法においてコストの削減量がより大きい．シミュレーション結果でも示されている様に，概ね $1/k$ のコスト削減が達成出来ている．

5 結論

本論文では任意の多項式基底間の基底変換に関して，従来の EDF を用いた手法について，normal basis への基底変換を用いることによる，より計算コストの小さい手法を提案した．又，計算量について議論し，拡大次数 k に対して，従来手法に比べ提案手法では概ね $1/k$ のコスト削減が見込まれることを示し，実際にシミュレーション結果からその正当性を示した．

本論文では，拡大体が optimal normal basis を持つ等の特別な場合は考えなかった．これらの有限体では，より高速に基底変換を求めるアルゴリズムが発見されている．今後，任意の有限体において，より効率的な基底変換求解アルゴリズムを研究していく必要がある．

参考文献

[1] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman: *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

[2] M. Alizadeh: Some Algorithms for Normality Testing Irreducible Polynomials and Computing Complexity of the Normal Polynomials over Finite Fields, *Applied Mathematical Sciences*, **6**(40), pp. 1997–2003, 2012.

[3] D. G. Cantor and H. Zassenhaus: A New Algoritm for Factoring Polynomials over Finite Fields, *Mathematics of Computation*, **36**(154), pp. 587–592, 1981.

[4] S. Gao: *Nromal Basis over Finite Fields*, PhD thesis, Warterloo, 1993.

[5] Joachim Von Zur Gathen and Mark Giesbrecht: Constructing Normal Bases in Finite Fields, *Journal of Symbolic Computation*, **10**, pp. 547–570, 1990.

[6] Jr. H. W. Lenstra: Finding Isomorphisms Between Finite Fields, *Mathematics of Computation*, **56**(193), pp. 329–347, 1991.

[7] M. K. Kyuregyan: Iterated constructions of irreducible polynomials over finite fields with linearly independent roots, *Finite Fields and Their Applications*, **10**(3), pp. 323–341, 2004.

[8] Melsik K. Kyuregyan: Recursive constructions of N-polynomials over GF(2^S), *Discrete Applied Mathematics*, **156**(9), pp. 1554–1559, 2008.

[9] A. K. Lenstra and E. R. Verheul: The XTR Public Key System, *Advances in Cryptology - CRYPTO 2000, Lecture Notes in Computer Science*, **1880**, pp. 1–19, Springer, 2000.

[10] R. Lidl and H. Niederreiter: *Finite Fields*, Encyclopedia of Mathematics and its Applications **20**, Cambridge University Press, Cambridge, UK, 1997.

[11] Y. Nogami, R. Namba, and Y. Morikawa: Finding a Basis Conversion Matrix via Prime Gauss Period Normal Basis, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **92**(6), pp. 1500–1507, 2009.

[12] Y. Nogami, A. Saito, and Y. Morikawa: Finite Extension Field with Modulus of All-one Polynomial and Representation of its Elements for Fast Arithmetic Operations, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **E86-A**(9), pp. 2376–2387, 2003.

[13] Yasuyuki Nogami, Hidehiro Kato, Kenta Nekado, Satoshi Uehara, and Yoshitaka Morikawa: Finding a Basis Conversion Matrix Using a Polynomial Basis Derived by a Small Multiplicative Cyclic Group, *IEEE Transactions on Information Theory*, **58**(7), pp. 4936–4947, 2012.

[14] C. Paar: *Efficient VLSI Architectures for Bit-Parallel Computation in Galois Fields*, PhD thesis, Institute for Experimental Mathematics, University of Essen, Essen, Germany, 1993.

[15] M. Shirase, T. Takagi, D. Choi, D. Han, and H. Kim: Efficient Computation of Eta Pairing over Binary Field with Vandermonde Matrix, *ETRI Journal*, **31**(2), pp. 129–139, 2009.

[16] V. Shoup: A Library for doing Number Theory, Available: `http://www.shoup.net/ntl/`.

[17] B. Sunar: An Efficient Basis Conversion Algorithm for Composite Fields with Given Representations, *IEEE Transactions on Computers*, **54**, pp. 992–997, IEEE Computer Society, Los Alamitos, CA, USA, 2005.

[18] Š. Schwarz: Construction Of Normal Bases in Cyclic Extensions of A Field, *Czechoslovak Mathematical Journal*, **38**(113), pp. 147–158, 1988.

[19] 難波諒, 野上保之, 森川良孝：Optimal Normal Basis を経由する同型な拡大体間の基底変換行列の構成法, 電子情報通信学会技術研究報告. SITE, 技術と社会・倫理, 第 106 巻, pp. 1–6, 一般社団法人電子情報通信学会, 2006.