SCIS 2017 2017 Symposium on
Cryptography and Information Security
Naha, Japan, Jan. 24 - 27, 2017
The Institute of Electronics,
Information and Communication Engineers

より高速な双線型型変換
Faster Bilinear-Type Conversion

星野 文学 *†
Fumitaka Hoshino

あらまし 非対称ペアリングは対称ペアリングと比較して代数的構造は複雑だが実装の効率はずっと良い．従ってペアリングに基づく暗号方式を構成する際，一旦は対称ペアリング上で設計を行い，後に非対称ペアリングの代数的構造に合わせ再設計し実装を得る事が多くなっている．2016 年阿部らは方式の安全性を維持したままこのタスクを自動実行する効率的なアルゴリズムを発表した [1, 2]．本稿ではこの方式を改良して幾分高速化する．

キーワード ペアリング, 整数計画法, Groth-Sahai 証明

1 はじめに

1.1 研究背景

一般に楕円曲線を用いたペアリングの実装では楕円曲線上の $\mathbb{F}_q$ 有理点の部分群 $E(\mathbb{F}_q)[\ell]$ およびそれと同型な群から有限体の乗法群 $\mathbb{F}_{q^k}^*$ への双準同型を利用してペアリングを構成する．この時ペアリングの安全性は少なくとも楕円曲線 $E(\mathbb{F}_q)[\ell]$ 上の楕円離散対数問題および有限体の乗法群 $\mathbb{F}_{q^k}^*$ 上の離散対数問題よりは易しい．k は埋め込み次数 (embedding degree) と呼ばれる整数で，k が小さいと解析計算量の漸近挙動が準指数的な乗法群の離散対数問題に引きずられ q を大きくする必要があり $E(\mathbb{F}_q)$ の演算効率が落ちるし，k が大き過ぎると $\mathbb{F}_{q^k}^*$ の演算効率が落ちる．従ってペアリングの実装を行なう場合は楕円離散対数問題および乗法群の離散対数問題の解析計算量が両方とも適当な大きさとなるよう q, k を適切な値に設定する事が望ましい [3]．かつては小標数の楕円曲線を用いた対称ペアリングが実装に優れた性質を持つと期待されていたが [4, 5, 6, 7], 近年攻撃技術の進展によりあまり安全で無い事が分かってきた [8, 9, 10]．従って現在では対称ペアリングを実装に用いる際は大きい素体の楕円曲線を使う必要があり，その場合埋め込み次数が 2 で固定される事が知られている．一方非対称ペアリングでは，より大きい埋め込み次数や豊かな代数的構造を利用する事によりサイズの小さい楕円曲線が使用できる [11]．しかし非対称ペアリングは対称ペアリングと比較すると代数的構造が複雑なので，暗号方式の設計には対称ペアリングが良く用いられる．従って方式の設計を一旦は対称ペアリング上で行い非対称ペアリングの複雑な代数的構造に合わせて方式を再設計し実装を得る事が増えており [12, 13, 14], そのような設計方針自体が設計者にとってメリットの有る新たなアプローチとなっている．さらに将来システムが大規模化して行くと非対称ペアリングを直接用いて最適な実装を得る仕事はいずれ人手では困難になると考えられる．

1.2 関連研究

もし，対称ペアリングに基づく設計から非対称ペアリングに基づく実装が自動的に得られるなら，煩わしい再設計の手間が省ける上，かつて対称ペアリング用に設計された大量の方式を見捨てる必要が無くなるであろう．そのような事は可能であろうか？ こうした疑問から自然に

(A) 変換可能性の判定
(B) 実行可能解の求解
(C) 最適解の求解

といった問題が生じ，それらを解くアルゴリズムの研究が始まった．[15, 16, 17, 18, 19, 20] のような初期の研究では (A),(B) に関して発見的な指標を与えるに留まって

* NTT セキュアプラットフォーム研究所, 〒180-8585 東京都武蔵野市緑町 3-9-11, Secure Platform Laboratories, NTT, 3-9-11 Midori-cho, Musashino-shi, Tokyo 180-8585, Japan
† 東京工業大学 情報理工学院, 〒226-8503 神奈川県横浜市緑区長津田町 4259, School of Computing, Tokyo Institute of Technology, 4259 Nagatsuta-cho, Midori-ku, Yokohama, Kanagawa 226-8503, Japan

いた．2013 年 Akinyele らは CCS2013 にて背景理論付き充足可能性判定 (satisfiability modulo theory, SMT) ソルバを用いて，(A)〜(C) を解く自動変換アルゴリズムおよびツール (AutoGroup) を発表したが，この変換は変換後の方式の安全性を保証するものではなかった [21]．2014 年 阿部らは CRYPTO2014 にて，方式の安全性を維持したまま (A)〜(C) を解く為のフレームワーク及びアルゴリズムを発表したが，このアルゴリズムは方式の規模に対して指数時間であった [22]．2015 年丹後らは SCIS2015 にて上記フレームワークの下 (A) を解く多項式時間アルゴリズムを提案したが (B),(C) については未解決のままであった [23]．同年 Akinyele らは CCS2015 にて SMT ソルバを使って上記フレームワークの下 (A)〜(C) を解くアルゴリズムおよびツール (AutoGroup+) を発表した [24]．2016 年阿部らは CRYPTO2016 にて，上記フレームワークの下 (A),(B) を解く多項式時間アルゴリズムおよび, (C) を整数計画 (integer programming, IP)[25] に多項式時間帰着して解く実用的なアルゴリズムを提案した [1]．

1.3 本研究の成果

本研究では [1] の帰着を改良して幾分高速化する方法を提案する．計算機実験を行なったところ，大きい例に関して [1] に対し，非商用ソルバを用いた場合には凡そ 5 倍，商用ソルバを用いた場合でも凡そ 1.5 倍の高速化が実現出来た．

2 準備

2.1 ペアリング

暗号学においてペアリングとは概ね次のような代数的構造を持つ符号を生成する確率的多項式時間アルゴリズム $\mathcal{G}$ の事である．

$$\mathcal{G} : 1^\lambda \xmapsto{\$} (\mathbb{L}, \mathbb{G}_0, \mathbb{G}_1, \mathbb{G}_T, \mathsf{aux})$$

1. $\mathbb{L}$ は $\#\mathcal{L} > 2^{\Theta(\lambda)}$ なる有限可換環 $\mathcal{L}$ の符号．$\mathbb{G}_0, \mathbb{G}_1, \mathbb{G}_T$ はそれぞれ $\#\mathcal{M} > 2^{\Theta(\lambda)}$ なる同じ $\mathcal{L}$-加群 $\mathcal{M}$ の符号．aux は補助出力．
2. 次の確率的多項式時間アルゴリズムが利用可能．
   - $\mathbb{L}, \mathbb{G}_0, \mathbb{G}_1, \mathbb{G}_T$ 上の標本．
   - $\mathbb{L}$ 上の環演算．
   - 和 : $\mathbb{G}_x \times \mathbb{G}_x \to \mathbb{G}_x$, $\forall x \in \{0,1,T\}$.
   - スカラー倍 : $\mathbb{L} \times \mathbb{G}_x \to \mathbb{G}_x$, $\forall x \in \{0,1,T\}$.
   - 非退化双準同型 $e : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_T$.
3. $\mathbb{G}_0, \mathbb{G}_1, \mathbb{G}_T$ 上の CDH 問題はそれぞれ計算困難．

λ は安全変数と呼ばれ，確率的多項式時間攻撃者に対して定義される何らかの利得が λ に関して無視可能となる事が期待される．

習慣により $\mathbb{L}$ は離散対数などと呼ばれ, $\mathbb{G}_x$, $\forall x \in \{0,1,T\}$ はそれぞれ群と呼ばれる．補助出力 aux は $\mathcal{G}$ の乱数テープの適当な関数である．$\mathcal{G}$ の具体的な出力 $(\mathbb{L}, \mathbb{G}_0, \mathbb{G}_1, \mathbb{G}_T, \mathsf{aux})$ は広い意味で共通参照文字列あるいはパラメタなどと呼ばれるが, これをペアリングと呼ぶ事もある．

$\mathbb{G}_x$ の加群としての和は積などと呼ばれ $g, h \in \mathbb{G}_x$ に対して $g \times h$, $g \cdot h$, gh 等と記述される．$\mathbb{G}_x$ の $\mathcal{L}$-加群としてのスカラー倍は冪乗などと呼ばれ $a \in \mathbb{L}$, $g \in \mathbb{G}_x$ に対して g^a と記述される．積と冪乗はまとめて群演算と呼ばれる．また非退化双準同型 e はペアリングと呼ばれる．

歴史的経緯と習慣によりペアリングと言った時それが e を指すのか，$(\mathbb{L}, \mathbb{G}_0, \mathbb{G}_1, \mathbb{G}_T, \mathsf{aux})$ を指すのか，$\mathcal{G}$ を指すのか，あるいは特にそれらを区別していないのかは文脈によって判断する必要がある．$\mathbb{G}_0, \mathbb{G}_1$ を入力群 (source group)，$\mathbb{G}_T$ を標的群 (target group) と呼ぶ．本稿では特に明示しない限り，単に群および群要素などと呼ぶ時は，それぞれ入力群および入力群要素を意味するとする．Galbraith らは，$\mathbb{G}_0, \mathbb{G}_1$ 間の非退化準同型 $\varphi : \mathbb{G}_1 \to \mathbb{G}_0$ に想定可能な性質に基づき暗号方式に用いられるペアリングを大雑把に以下の 3 つの型に分類した [26]．

Type 1: φ, φ^{-1} が共に多項式時間．
Type 2: φ が一方向．
Type 3: φ, φ^{-1} が共に多項式時間でない．

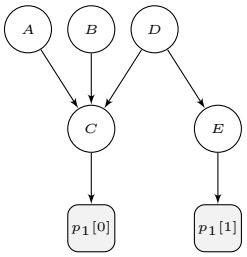
一般に Type 1 を対称ペアリングと呼び，Type 2，Type 3 を非対称ペアリングと呼ぶ．対称ペアリングにおいては $\mathbb{G}_0$ と $\mathbb{G}_1$ はいつでも双方向に多項式時間で行き来出来る為，両者を特に区別する必要はなく，それらは単に $\mathbb{G}$ と記述される．Type 2 は Type 3 を利用して実装できるので，特殊な機能や安全性証明を考える時以外は Type 2 が登場する事はあまり無い．本稿では非対称ペアリングとして特に Type 3 を想定する．このとき $\mathbb{G}_0$ と $\mathbb{G}_1$ は明確に区別される．演算速度や群要素のサイズといった実装上の問題に関して，対称ペアリングよりもずっと効率的な非対称ペアリングの存在が知られている [11]．

2.2 依存関係グラフ [22, 27]

方式に対して，その依存関係グラフ (dependency graph) とは方式の構成 (construction)，構文 (syntax)，帰着 (reduction)，仮定 (assumption) 等で使用される群要素変

数の情報の流れ (data flow) を抽象した有向グラフである．各頂点はプログラム中で使用される群要素変数を，各辺はその依存関係 (終点は始点に依存する) を表現する．図 1 左は群要素 A, B, D を入力とし群演算を使って C, E を計算しそのペアリング $e(C, E)$ を出力するアルゴリズムの例であり，図 1 右はその依存関係グラフである．

Sample(a, A, B, D):
  $a \in \mathbb{Z}/p\mathbb{Z}$;
  $A, B, C, D, E \in \mathbb{G}$;
  if $a = 0$ **then**
    $C := A \cdot B$; $E := D$;
  else
    $C := D^a$; $E := D^3$;
  endif
  Output $e(C, E)$;



$\mathbb{G}$

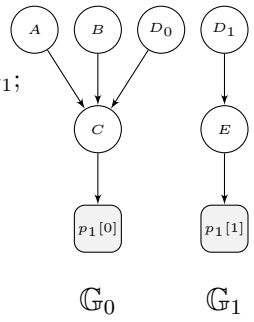
図 1　依存関係グラフの例

依存関係グラフには群演算を介した群要素同士の関係のみが記述され，”**if-then-else**” 命令のようなプログラムの構造や $a \in \mathbb{Z}/p\mathbb{Z}$ のような群要素以外の変数あるいは標的群上の演算等は全て捨象される．

ペアリングは入力群要素 2 つを入力として標的群要素 1 つを出力する関数なので，依存関係グラフ中では出力辺を持たない対となる 2 つの頂点として表現される．ペアリングへの入力に相当する方式中の群要素変数は暗黙に宣言および定義されるとする．例えば図 1 左のプログラム中には A, B, C, D, E の 5 つの群要素変数が宣言および定義されているが，ペアリングへの入力に相当する 2 つの群要素変数は宣言も定義もされていない．この暗黙の群要素変数は図 1 右のグラフでは $p_1[0], p_1[1]$ と表現されている．ペアリングが複数ある場合は i 番目のペアリングは $p_i[0], p_i[1]$ なる暗黙の群要素変数を持つとする．また, 一般にプログラム中の群演算や群要素比較で構成される表現からも，依存関係の文脈で必要なら適当に暗黙の群要素変数が宣言および定義されるとする．

阿部らは対称ペアリングに基づく暗号方式を安全性を維持したまま非対称ペアリングに基づく暗号方式に自動変換する為のフレームワークを提案した [22, 27]．阿部らのフレームワークでは，対称ペアリング上定義された方式の構成，構文，帰着，仮定等をまとめた 1 つの依存関係グラフから，ある制約に従う二つの部分グラフを導く．この部分グラフ達を導出する事を分割と呼ぶ．各部分グラフは変換後の方式を抽象しており，それぞれ $\mathbb{G}_0$ および $\mathbb{G}_1$ の群要素に関する依存関係グラフとなる．そして二つの部分グラフおよび元の方式より非対称ペアリング上定義された方式の構成，構文，帰着，仮定等が再構築される．図 1 のプログラムの分割による変換の例を図 2 に示す．分割

Sample$(a, A, B, (D_0, D_1))$:
  $a \in \mathbb{Z}/p\mathbb{Z}$;
  $A, B, C, D_0 \in \mathbb{G}_0$; $D_1, E \in \mathbb{G}_1$;
  if $a = 0$ **then**
    $C := A \cdot B$; $E := D_1$;
  else
    $C := D_0^a$; $E := D_1^3$;
  endif
  Output $e(C, E)$;



$\mathbb{G}_0$　$\mathbb{G}_1$

図 2　分割による変換の例

による変換を考察するため，概ね [22, 27, 1, 2] に従っていくつか用語や記法を定義する．

定義 1（$\mathbb{G}, \mathbb{G}_0, \mathbb{G}_1$） 前章で Type 1 におけるペアリングの入力群を $\mathbb{G}$ と定義したが，以下 $\mathbb{G}$ を依存関係グラフまたはその頂点集合と再定義する．また，前章で一般の設定におけるペアリングの入力群を $\mathbb{G}_0$ および $\mathbb{G}_1$ と定義したが，以下 $\mathbb{G}_0$ および $\mathbb{G}_1$ を依存関係グラフ $\mathbb{G}$ の部分グラフまたはその頂点集合と再定義する．

以下依存関係グラフ $\mathbb{G}$ から分割によって 2 つの部分グラフ $\mathbb{G}_b$，$b \in \{0, 1\}$ が生成されるとする．

定義 2（祖先） 頂点 $x \in \mathbb{G}$ に対して x に到達可能な頂点 $y \in \mathbb{G}$ (但し x を含まない) を x の祖先と呼ぶ．このとき頂点 x は頂点 y を祖先に持つと言う．

定義 3（子孫） 頂点 $x \in \mathbb{G}$ に対して x から到達可能な頂点 $y \in \mathbb{G}$ (但し x を含まない) を x の子孫と呼ぶ．このとき頂点 x は頂点 y を子孫に持つと言う．

一般に有効な変数定義の連鎖によって構成された単一アルゴリズムのデータフローは有向非巡回グラフを構成するので閉路 (cycle) を持たない．しかし方式の構成，構文，帰着，仮定等の複数のアルゴリズムが統合される場合，依存関係グラフは閉路を持ち得る．依存関係グラフが閉路を持つ場合，ある頂点 y がある頂点 x の祖先でありかつ子孫であるという事がありうる．閉路を構成する頂点同士はどれも互いに祖先でありかつ子孫であるから，依存関係の文脈では閉路自体を単一の頂点と同一視してよい．この性質を考慮し，有向グラフにおける葉の概念を多少拡張し，次のように定める．

定義 4（葉） 出力が無い頂点，または (他の葉に到達可能な) 出力が無い閉路を代表する頂点を葉と呼ぶ.

定義によりペアリングへの入力は必ず葉となる.
本稿では整数の集合 $\{0,1\}$ を論理値の集合と同一視し 0 を偽 1 を真とみなす. 論理式は全て古典論理 (ブール代数) に従い演繹できるとする. 概ね慣例に従い次の記法を用いる.

定義 5（$\wedge, \vee, \oplus, \Rightarrow, \Leftrightarrow, \neg, \bar{\ }$） 二つの論理式 x, y に対してその論理積，論理和，排他的論理和，含意，同値をそれぞれ $x \wedge y$, $x \vee y$, $x \oplus y$, $x \Rightarrow y$, $x \Leftrightarrow y$ と記述する. 論理式 x に対してその否定を $\neg x$ または $\bar{x}$ と記述する.

論理的な言明と代数的な関係式の間の往来を容易にする為，次の定義を行なう.

定義 6（割り当て変数 $(x \in \mathbb{G}_b)$） $(x \in \mathbb{G}_b)$ なる表現をその真理値に従う論理変項，即ち $\{0,1\}$ 上の変数とする. $(x \in \mathbb{G}_b)$ を割り当て変数と呼ぶ.

即ち

$$(x \in \mathbb{G}_b) = \begin{cases} 1 & (x \in \mathbb{G}_b \text{ の場合}) \\ 0 & (x \notin \mathbb{G}_b \text{ の場合}) \end{cases}$$

である. 割り当て変数 $(x \in \mathbb{G}_b)$ に関して，写像 $\{(x \in \mathbb{G}_b)\} \to \{0,1\}$ を割り当て変数 $(x \in \mathbb{G}_b)$ への割り当てあるいは代入と呼ぶ. 全ての割り当て変数の集合を $V_{\mathbb{G}}$ とする. 即ち

$$V_{\mathbb{G}} = \{(x \in \mathbb{G}_b) |\ x \in \mathbb{G}, b \in \{0,1\}\}$$

である. 写像 $V_{\mathbb{G}} \to \{0,1\}$ を依存関係グラフ $\mathbb{G}$ への割り当て，あるいは単に割り当てと呼ぶ. 割り当てを決定すれば分割を決定する事が出来る.
$\mathbb{G}$ 中に n 個の頂点が存在する時，$2n$ 個の割り当て変数が存在する. 従って，依存関係グラフ $\mathbb{G}$ への割り当て，即ち写像 $V_{\mathbb{G}} \to \{0,1\}$ は 2^{2n} 個存在する. それらの割り当ての中には方式を変換する上で有効でないものも存在する. どのような割り当てが有効となるかは後に議論する.

定義 7（重複頂点） ある頂点 x が

$$x \in \mathbb{G}_0 \quad \wedge \quad x \in \mathbb{G}_1$$

を満たすなら x を重複頂点 (duplicated node) と呼ぶ.

定義 8（重複不可頂点） 何らかの理由により分割後に重複頂点となる可能性が無い頂点を重複不可頂点 (non-duplicatable node) と呼ぶ.

重複不可頂点はアルゴルズム中の `HashToPoint()` 演算の出力を表現したり，データサイズを節約する為に方式の設計者により設定される事がある. また後述する理由により本稿で扱う変換では葉は全て重複不可頂点となる.
記述を簡素化するため，次の定義を行なう.

定義 9（割り当て変数 x） x が重複不可頂点のとき，x を改めて $\{0,1\}$ 上の変数とみなし

$$x = (x \in \mathbb{G}_1)$$

と定義する.

即ち重複不可頂点に関しては x を $(x \in \mathbb{G}_1)$ の略記とみなす. 重複不可頂点 $x \in \mathbb{G}$ および $b \in \{0,1\}$ に関して

$$x = b \quad \Leftrightarrow \quad x \in \mathbb{G}_b \quad \Leftrightarrow \quad x \notin \mathbb{G}_{\bar{b}}$$

である. 即ち重複不可頂点 $x \in \mathbb{G}$ が $x \in \mathbb{G}_b$ なるとき x とは b の事である.

3 従来研究

3.1 阿部らのフレームワーク [22, 27]

対称ペアリングは代数的構造が簡単な為，多くの暗号方式が対称ペアリングをを前提に設計されている. しかし時間計算量や領域計算量等の効率の観点から，現在では非対称ペアリングを使用することが望ましいと考えられるようになった. 阿部らは構文および帰着がゲーム様対話アルゴリズムにより定義され，計算または判定問題の困難性へのブラックボックス帰着によって安全性が証明された対称ペアリングに基づく暗号方式を考察し，これを安全性を保ったまま非対称ペアリングに基づく暗号方式に自動変換する以下のフレームワークを提案した [22, 27].

Step 1: Type 1 ペアリング上の暗号方式の構成，構文，帰着，仮定等の各アルゴリズムが入力される.

Step 2: 各アルゴリズム中の群要素変数の依存関係グラフ (有向グラフ) をそれぞれ構成し，それを一つに統合する.

Step 3: 後述の制約条件の下で依存関係グラフを 2 つの部分グラフに分割する. (分割が存在しない場合は本フレームワークでは変換不可能とする.)

Step 4: 分割に基づき Type 3 ペアリング上の暗号方式の構成，構文，帰着，仮定等の各アルゴリズムが出力される.

阿部らは，上記の分割に関して次の定理を証明した.

定理 1（阿部ら [22]） 上記の分割が下記の 4 つの制約を満たすとき，導出される方式は元の方式の機能と安全性を Type 3 上で実現する.

制約 1　分割された 2 つの部分グラフの合併は元のグラフと一致する.

制約 2　もし部分グラフが頂点 x を含むなら，その部分グラフは頂点 x への全ての路 (path) を含む.

制約 3　一つのペアリングに含まれる二つの頂点は互いに異なる部分グラフに含まれる.

制約 4　もし一方の部分グラフが重複不可頂点を含むなら他方の部分グラフはその頂点を含まない.

これら 4 つの制約条件を満たす分割を有効分割 (valid split) と呼び，その割り当てを有効割り当て (valid assignment) と呼ぶ. 阿部らの論文 [22, 27] では Step 3 の分割は，方式に関する時間計算量や領域計算量といった何らかの評価関数を，総当たり探索により評価する事によって決定されるとしている.

3.2　有効分割と最適性の仮定

以下，有効分割に関して考察する.

制約 1　分割された 2 つの部分グラフの合併は元のグラフと一致する.

制約 1 は明らかに

$$\mathbb{G} = \mathbb{G}_0 \cup \mathbb{G}_1$$

を意味する. これは頂点 x が $x \in \mathbb{G}$ であるなら

$$x \in \mathbb{G}_0 \vee x \in \mathbb{G}_1 \tag{1}$$

である事を意味しており，従って

$$x \notin \mathbb{G}_0 \wedge x \notin \mathbb{G}_1$$

は成立しない. 式 (1) の選言は

$$(x \in \mathbb{G}_0) + (x \in \mathbb{G}_1) \geq 1 \tag{2}$$

と同義である. x が重複不可頂点である場合は

$$(x \in \mathbb{G}_0) + (x \in \mathbb{G}_1) = 1 \tag{3}$$

が成立する.

$$(x \in \mathbb{G}_0) \oplus (x \in \mathbb{G}_1) = 1 \tag{4}$$

としても良い.
本稿では評価関数に関して次の仮定を設ける.

仮定 1（最適性の仮定）　$\mathbb{G}$ に関する二つの有効分割が存在しその違いが，”ある頂点 x が重複頂点であるか否か”のみであったとする. この時 x が重複頂点となる分割の評価値は必ずもう一方の分割の評価値と等しいかより悪い.

この仮定は重複頂点が少なければ少ないほど方式に要する領域計算量や時間計算量が小さくなり，方式が効率的になるという観察に基づいている. 評価関数がそうした評価基準とは異なる価値観 (例えば出来るだけ沢山の重複頂点を持ちたい等) を持つ場合には本稿の理論は修正を要する. 本稿ではこの仮定を最適性の仮定 (assumption of optimality) と呼ぶ. この仮定により，冗長な重複頂点を含む分割を考察の対象から除外出来る. 以下，この仮定の下で有効分割に関して考察する.

制約 2　もし部分グラフが頂点 x を含むなら，その部分グラフは頂点 x への全ての路 (path) を含む.

y を x の祖先とする. 制約 2 は明らかに

$$(x \in \mathbb{G}_b) \Rightarrow (y \in \mathbb{G}_b)$$

を意味する. 下記真理値表の通り含意は不等号と同義である.

A	B	$A \Rightarrow B$	$A \leq B$
0	0	1	1
1	0	0	0
0	1	1	1
1	1	1	1

従って

$$(x \in \mathbb{G}_b) \leq (y \in \mathbb{G}_b). \tag{5}$$

y を頂点，D_y を y の子孫の全体とする. 割り当て $(y \in \mathbb{G}_b) \in \{0, 1\}$ は下記を満足する必要がある.

$$(y \in \mathbb{G}_b) \geq (x \in \mathbb{G}_b), \quad \forall x \in D_y. \tag{6}$$

即ち

$$(y \in \mathbb{G}_b) \geq \bigvee_{x \in D_y} (x \in \mathbb{G}_b).$$

最適性の仮定により等号のみを考えれば良い.

$$(y \in \mathbb{G}_b) = \bigvee_{x \in D_y} (x \in \mathbb{G}_b). \tag{7}$$

今，仮に y が子孫を持たないと仮定すると，$x \in D_y$ なる x が存在しないので，式 (7) を考慮せずに $(y \in \mathbb{G}_b)$ の値を決定出来る. この時最適性の仮定により y が重複頂点となる事は除外できる. 即ち，もし y が葉なら，y は重複不可として良い.
次に，全ての葉の割り当てが決定したと仮定する. 葉から繰り返し上流に遡り，上流側の頂点 y の割り当てを式 (7) により決定したい. 最適性の仮定により全ての頂点の割り当てはその頂点から到達可能な全ての葉の割り当てによって決定されるとして良い. 即ち値 $(y \in \mathbb{G}_b) \in \{0, 1\}$

は葉の値のみに依存する．L_y を D_y に含まれる葉の全体とする．式 (7) より

$$(y \in \mathbb{G}_b) = \bigvee_{x \in L_y} (x \in \mathbb{G}_b). \tag{8}$$

制約 3　一つのペアリングに含まれる二つの頂点は互いに異なる部分グラフに含まれる．

x, y をペアリングへの入力対となる頂点とする．x, y は葉なので重複不可として良い．従って x, y を $\{0, 1\}$ 変数として良い．明らかに制約 3 は

$$x \oplus y = 1$$

を意味する．x, y が $\{0, 1\}$ 変数なので

$$x + y = 1 \tag{9}$$

として良い．

制約 4　もし一方の部分グラフが重複不可頂点を含むなら他方の部分グラフはその頂点を含まない．

x, y を重複不可頂点とし，x を y の子孫とする．式 (4) より

$$\begin{aligned}(x \in \mathbb{G}_b) \oplus (x \in \mathbb{G}_{\bar{b}}) &= 1\\ (y \in \mathbb{G}_b) \oplus (y \in \mathbb{G}_{\bar{b}}) &= 1\end{aligned} \tag{10}$$

式 (5) より

$$(x \in \mathbb{G}_b) \leq (y \in \mathbb{G}_b).$$

が成立する．$(x \in \mathbb{G}_b) \neq (y \in \mathbb{G}_b)$ を仮定すると，直ちに $(x \in \mathbb{G}_b) = 0$ および $(y \in \mathbb{G}_b) = 1$ が成立する．なぜなら

$$0 \leq (x \in \mathbb{G}_b) < (y \in \mathbb{G}_b) \leq 1.$$

であるから．(10) により $(x \in \mathbb{G}_{\bar{b}}) = 1$ および $(y \in \mathbb{G}_{\bar{b}}) = 0$ が成立する．従って式 (5) 即ち

$$(x \in \mathbb{G}_{\bar{b}}) \leq (y \in \mathbb{G}_{\bar{b}})$$

が成立しない．従って $(x \in \mathbb{G}_b) \neq (y \in \mathbb{G}_b)$ なる仮定は成立しない．よって $(x \in \mathbb{G}_b) = (y \in \mathbb{G}_b)$ 即ち $x = y$ である．即ち，もし x, y が一つの路上の二つの重複不可頂点であるなら

$$x - y = 0 \tag{11}$$

が成立する．

$$x \oplus y = 0$$

としても良い．

3.3　阿部らのアルゴリズム [1, 2]

論文 [22, 27] では Step 3 の分割は，方式に関する時間計算量や領域計算量といった何らかの関数を，総当たり探索により評価する事によって決定されるとしていた．ペアリングの数とペアリングへの入力でない葉の数の総和を n とすると，上記の総当たり探索には 2^n 回の演算が必要であり，n が大きい時には計算量が大きくなりすぎて実行困難となる．阿部らはこの問題を有効分割の求解問題と最適化問題とに分離し，多項式時間求解アルゴリズムと次の最適化アルゴリズムを提案した [1, 2].

Step 1: 制約 (3), (8), (9), (11) をすべて書き下す.

Step 2: (8) を下記の良く知られた方法で線形制約に変換する.

Step 3: 割り当て変数を使って線形目的関数を書き下す.

Step 4: 全ての線形制約および線形目的関数を任意の 0-1 整数計画アルゴリズムに入力し厳密解あるいは近似解を得る.

線形制約への変換　$(d \in \mathbb{G}_1)$ を d と略記すると (8) は

$$\begin{aligned}(x \in \mathbb{G}_0) &= \bigvee_{d \in L_x} \neg d\\ (x \in \mathbb{G}_1) &= \bigvee_{d \in L_x} d\end{aligned} \tag{12}$$

と記述できる．ところで $\{0, 1\}$ 上の変数 y_0, y_1, z について，

$$\begin{aligned}z = y_0 \wedge y_1 \quad &\Leftrightarrow \quad \begin{cases} z - y_0 - y_1 + 1 \geq 0,\\ y_0 - z \geq 0,\\ y_1 - z \geq 0,\end{cases}\\ z = \neg y_0 \quad &\Leftrightarrow \quad z = 1 - y_0.\end{aligned}$$

である．即ち $\{0, 1\}$ 変数の任意の連言と否定は線型制約の元で変数に置き換え可能で，従ってどのような命題変数の論理式もこの置き換えを繰り返す事によって $\{0, 1\}$ 変数の線型制約付き線型式に変換出来る事が良く知られている．特に $y_1, \ldots, y_k, z \in \{0, 1\}$ の時

$$z = \bigvee_{i=1}^{k} y_i \quad \Leftrightarrow \quad \begin{cases} z - \sum_{i=1}^{k} y_i \leq 0,\\ z - y_i \geq 0, \forall i \in \{1, \ldots, k\}.\end{cases} \tag{13}$$

である．従って制約 (12) は線型制約

$$\begin{aligned}&(x \in \mathbb{G}_0) + \sum_{d \in L_x} d \leq \#L_x,\\ &(x \in \mathbb{G}_0) + d \geq 1, \quad \forall d \in L_x,\\ &(x \in \mathbb{G}_1) - \sum_{d \in L_x} d \leq 0,\\ &(x \in \mathbb{G}_1) - d \geq 0, \quad \forall d \in L_x.\end{aligned} \tag{14}$$

と同値である．

4　提案法

阿部らのアルゴリズム [1, 2] では式 (7) から得た式 (8) を (13) を使って無理やり線形制約に置き換えた．本提案法ではこの置き換えを辞めて式 (6) から最適性の仮定を使って直接以下を得る．

y を頂点，L_y を y の子孫のうち葉であるものの全体とする．割り当て $(y \in \mathbb{G}_b) \in \{0,1\}$ は下記を満足する必要がある．

$$(y \in \mathbb{G}_b) \geq (x \in \mathbb{G}_b), \quad \forall x \in L_y. \tag{15}$$

即ち式 (14) の第 1 式と第 3 式は不要となった．従って次のアルゴリズムを得る．

Step 1: 制約 (3), (15), (9), (11) をすべて書き下す．

Step 2: 割り当て変数を使って線形目的関数を書き下す．

Step 3: 全ての線形制約および線形目的関数を任意の 0-1 整数計画アルゴリズムに入力し厳密解あるいは近似解を得る．

5　実験

都合により表 1 の二つの環境で実験を行なった．以下，本

表 1　実験環境

	環境 1	環境 2
CPU	Core i5-3570	Xeon E5-2699 v3
clock	3.40GHz	2.30GHz
turbo	3.80GHz	3.60GHz
solver	SCIP 3.1.1[28] (非商用)	gurobi 6.5.0[29] (商用)
#thread	1	1
OS	Ubuntu 15.04	CentOS 6.5

実験のためランダムに生成した問題例の集合を $\mathcal{I}$ とし，$i \in \mathcal{I}$ に対して，従来法 [1, 2] および提案法による最適分割に要する時間 (最適化時間) をそれぞれ $T_0(i), T_1(i)$ とする．ランダムな問題例の作成方法は [1] の方法を用いた．頂点数毎に問題例を 10 例づつ作成し，最適化時間即ち $T_0(i), T_1(i)$ を各環境で測定した．図 3 および図 5 は各環境で測定した最適化時間即ち $T_0(i), T_1(i)$ を頂点数に対してプロットしたものである．図 4 および図 6 は各環境における従来法に対する提案法の速度利得即ち $T_0(i)/T_1(i)$ を頂点数に対してプロットしたものである．速度利得が 1 より大きければ提案法には効果があり，1 以下なら効果が

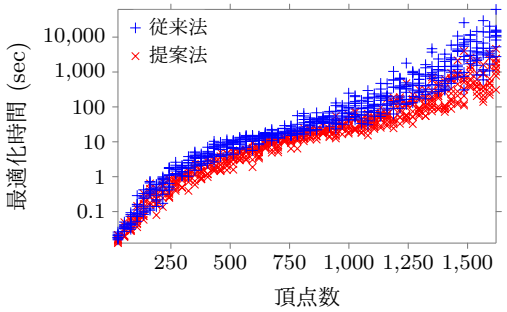


図 3　非商用ソルバ上での最適化時間

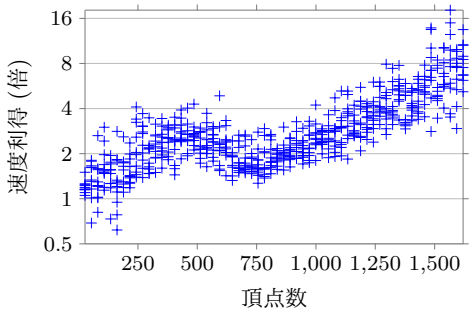


図 4　非商用ソルバ上での速度利得

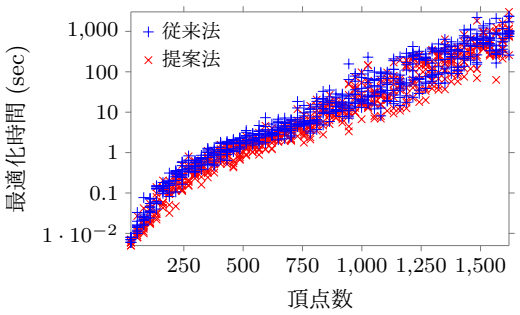


図 5　商用ソルバ上での最適化時間

無いあるいは逆効果であった事を示す．非商用ソルバを用いた環境 1 では概ね効果が認められる．特に分割に小一時間ほどを要するような規模の大きい方式では大きい効果が得られた．一方商用ソルバを用いた環境 2 では効果は限定的で概ね 1.5 倍ほどにとどまった．

6　謝辞

実験で非商用ソルバの代表として SCIP[28] を利用したので，ここで謝辞を申し上げる．

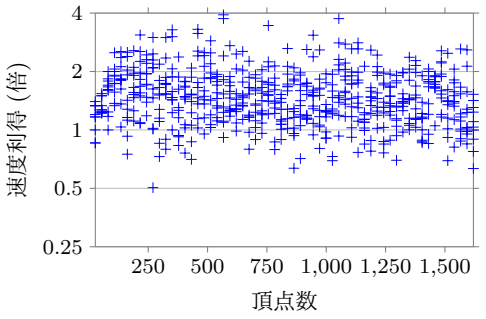


図 6　商用ソルバ上での速度利得

参考文献

[1] Masayuki Abe, Fumitaka Hoshino, and Miyako Ohkubo. Design in Type-I, Run in Type-III: Fast and Scalable Bilinear-Type Conversion Using Integer Programming. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 387–415. Springer, 2016.

[2] Fumitaka Hoshino, Masayuki Abe, and Miyako Ohkubo. Optimal Conversion from Symmetric Pairing-based Scheme to Asymmetric One. In *Proc. of SCIS 2016 2016 Symposium on Cryptography and Information Security Kumamoto, Japan, Jan. 19 - 22, 2016*. IEICE, 2016.

[3] Diego F. Aranha, Paulo S. L. M. Barreto, Patrick Longa, and Jefferson E. Ricardini. The realm of the pairings. In Tanja Lange, Kristin E. Lauter, and Petr Lisonek, editors, *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 of *Lecture Notes in Computer Science*, pages 3–25. Springer, 2013.

[4] Dan Page and Nigel P. Smart. Hardware implementation of finite fields of characteristic three. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*, volume 2523 of *Lecture Notes in Computer Science*, pages 529–539. Springer, 2002.

[5] Iwan M. Duursma and Hyang-Sook Lee. Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$. In Chi-Sung Laih, editor, *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings*, volume 2894 of *Lecture Notes in Computer Science*, pages 111–123. Springer, 2003.

[6] Yuto Kawahara, Masaaki Shirase, Tsuyoshi Takagi, and Eiji Okamoto. Efficient Software Implemenetation of η_T Pairing. In *Proc. of SCIS 2007 The 2007 Symposium on Cryptography and Information Security Sasebo, Japan, Jan. 23-26, 2007*. IEICE, 2012.

[7] Gen Takahashi, Fumitaka Hoshino, and Tetutaro Kobayashi. Efficient GF(3^m) Multiplication Algorithm for η_T Pairing. In *Proc. of SCIS 2008 The 2008 Symposium on Cryptography and Information Security Miyazaki, Japan, Jan. 22-25, 2008*. IEICE, 2012.

[8] Faruk Göloglu, Robert Granger, Gary McGuire, and Jens Zumbrägel. On the function field sieve and the impact of higher splitting probabilities: Application to discrete logarithms in $\mathbb{F}_{2^{1971}}$. *IACR Cryptology ePrint Archive*, 2013:74, 2013.

[9] Razvan Barbulescu, Pierrick Gaudry, Antoine Joux, and Emmanuel Thomé. A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. *IACR Cryptology ePrint Archive*, 2013:400, 2013.

[10] Antoine Joux. A new index calculus algorithm with complexity $L(1/4 + o(1))$ in very small characteristic. *IACR Cryptology ePrint Archive*, 2013:95, 2013.

[11] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In Bart Preneel and Stafford E. Tavares, editors, *Selected Areas in Cryptography, 12th International Workshop, SAC 2005, Kingston, ON, Canada, August 11-12, 2005, Revised Selected Papers*, volume 3897 of *Lecture Notes in Computer Science*, pages 319–331. Springer, 2005.

[12] Dennis Hofheinz. Algebraic partitioning: Fully compact and (almost) tightly secure cryptography. *IACR Cryptology ePrint Archive*, 2015:499, 2015.

[13] Atsushi Fujioka, Koutarou Suzuki, and Berkant Ustaoglu. Ephemeral key leakage resilient and efficient id-akes that can share identities, private and master keys. In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, *Pairing-Based Cryptography - Pairing 2010 - 4th International Conference, Yamanaka Hot Spring, Japan, December 2010. Proceedings*, volume 6487 of *Lecture Notes in Computer Science*, pages 187–205. Springer, 2010.

[14] Atsushi Fujioka, Fumitaka Hoshino, Tetsutaro Kobayashi, Koutarou Suzuki, Berkant Ustaoglu, and Kazuki Yoneyama. id-eCK Secure ID-Based Authenticated Key Exchange on Symmetric and Asymmetric Pairing. *IEICE Transactions*, 96-A(6):1139–1155, 2013.

[15] Nigel P. Smart and Frederik Vercauteren. On computable isomorphisms in efficient asymmetric pairing-based systems. *Discrete Applied Mathematics*, 155(4):538–547, 2007.

[16] Sanjit Chatterjee and Alfred Menezes. On cryptographic protocols employing asymmetric pairings - the role of Ψ revisited. *IACR Cryptology ePrint Archive*, 2009:480, 2009.

[17] Masaaki Shirase. Symmetric Pairing on Ordinary Elliptic Curves. In *Proc. of CSS 2010 Computer Security Symposium 2010 in Okayama, Japan, Oct. 19-21, 2010*. IPSJ, SIG CSEC, 2010.

[18] Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3):141–167, 2010.

[19] Sanjit Chatterjee and Alfred Menezes. On cryptographic protocols employing asymmetric pairings - the role of Ψ revisited. *Discrete Applied Mathematics*, 159(13):1311–1322, 2011.

[20] Tetsutaro Kobayashi, Fumitaka Hoshino, and Koutarou Suzuki. Symmetric Pairing based on Asymmetric Pairing. In *Proc. of SCIS 2012 The 29th Symposium on Cryptography and Information Security Kanazawa, Japan, Jan. 30 - Feb. 2, 2012*. IEICE, 2012.

[21] Joseph A. Akinyele, Matthew Green, and Susan Hohenberger. Using SMT solvers to automate design tasks for encryption and signature schemes. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 399–410. ACM, 2013.

[22] Masayuki Abe, Jens Groth, Miyako Ohkubo, and Takeya Tango. Converting cryptographic schemes from symmetric to asymmetric bilinear groups. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 241–260. Springer, 2014.

[23] Takeya Tango, Masayuki Abe, Tatsuaki Okamoto, and Miyako Ohkubo. Polynomial-Time Algorithm for Deciding Possibility of Converting Cryptographic Schemes from Type-I to III Pairing Groups. In *Proc. of SCIS 2015 The 32nd Symposium on Cryptography and Information Security Kokura, Japan, Jan. 20 - 23, 2015*. IEICE, 2015.

[24] Joseph A. Akinyele, Christina Garman, and Susan Hohenberger. Automating Fast and Secure Translations from Type-I to Type-III Pairing Schemes. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-6, 2015*, pages 1370–1381. ACM, 2015.

[25] George B. Dantzig. On the significance of solving linear programming problems with some integer variables. *Econometrica*, 28(1):30–44, 1960.

[26] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[27] Takeya Tango, Masayuki Abe, and Tatsuaki Okamoto. Implementation of Automated Translation for Schemes on Symmetric Bilinear Groups. In *Proc. of SCIS 2014 The 31st Symposium on Cryptography and Information Security kagoshima, Japan, Jan. 21 - 24, 2014*. IEICE, 2014.

[28] Gerald Gamrath, Tobias Fischer, Tristan Gally, Ambros M. Gleixner, Gregor Hendel, Thorsten Koch, Stephen J. Maher, Matthias Miltenberger, Benjamin Müller, Marc E. Pfetsch, Christian Puchert, Daniel Rehfeldt, Sebastian Schenker, Robert Schwarz, Felipe Serrano, Yuji Shinano, Stefan Vigerske, Dieter Weninger, Michael Winkler, Jonas T. Witt, and Jakob Witzig. The scip optimization suite 3.2. Technical Report 15-60, ZIB, Takustr.7, 14195 Berlin, 2016.

[29] Gurobi Optimization, Inc. Gurobi optimizer reference manual. In http://www.gurobi.com/.