SCIS 2015 The 32nd Symposium on
Cryptography and Information Security
Kokura, Japan, Jan. 20 - 23, 2015
The Institute of Electronics,
Information and Communication Engineers

暗号方式のペアリング群タイプの変換可能性に対する多項式時間判定アルゴリズムの提案

Polynomial-Time Algorithm for Deciding Possibility of Converting Cryptographic Schemes from Type-I to III Pairing Groups

丹後偉也 * Takeya Tango　阿部正幸 * † Masayuki Abe　岡本龍明 * † Tatsuaki Okamoto　大久保美也子 ‡ Miyako Ohkubo

あらまし 楕円曲線上で定義されるペアリングには３つのタイプがあり，そのうちのタイプ１は近年の離散対数問題アルゴリズムの進展により，効率性が悪化してしまった．そこで，既存のタイプ１のペアリングで構成された暗号方式について，効率の良いタイプ３を用いた暗号方式に変換するアルゴリズムが設計，実装され，この変換アルゴリズムでは最適な変換パターンを出力する．しかし，計算するためにペアリングノード数に対して指数時間かかってしまうという問題があった．本研究ではタイプ１からタイプ３へ変換できるかどうかのみを多項式時間で判定するアルゴリズムについて設計，実装を行う．そして，実際にいくつかの暗号方式について変換不可能性の判定を行い，その有用性について考察を行う．

Keywords: Conversion, Symmetric Bilinear Groups, Asymmetric Bilinear Groups

1 はじめに

1.1 背景

楕円曲線上で定義されるペアリングによって，三者間公開鍵共有法や ID ベース暗号を構成でき，それらを応用することによって多くの暗号方式が提案されてきた．Galbraith，Paterson，Smart らは [1] で，このペアリングに入力される群によって３つのタイプに分類した．そのうちのタイプ１は $\mathbb{G} = \tilde{\mathbb{G}}$ で，タイプ３は $\mathbb{G} \neq \tilde{\mathbb{G}}$ であり，$\mathbb{G}, \tilde{\mathbb{G}}$ 間の効率的に計算できる同型写像が無い．現在では，暗号方式を構築するためにタイプ３のペアリング群を用いることによって，他のタイプと較べて群要素のサイズは最小となり，計算効率が最大となるが，タイプ１ではペアリング群が対称的であるという単純な構造であるため，多くの暗号方式ではタイプ１が用いられている．

しかし，最近の離散対数問題アルゴリズムが進展してきたことにより，タイプ１を用いた暗号方式で安全性を保つためにはより大きい群要素のサイズが求められるようになり，計算効率が悪くなってしまった．タイプ３ではその影響を受けないため，サイズを抑えられる．そこで，あるタイプ１のペアリング群で構成された暗号方式を，手順を維持したままタイプ３の暗号方式に変換するアルゴリズムが [2] で設計，[3] で実装された．この変換アルゴリズムでは，可能な変換パターンについて全通り調べ，全パターンのうち最も計算効率が良い変換パターンを抽出し出力する．しかし，入力する暗号方式のペアリングノード数に対して計算量が指数時間となってしまうという問題があった．入力される暗号方式によっては，全パターンに対して変換不可能な場合があるが，このように変換可能か不可能かのみを判定したい場合でも全パターン調べる必要性がある．

1.2 目的

本論文では，タイプ１のペアリング群で構成されたある暗号方式が，タイプ３の暗号方式へ変換可能か不可能かを多項式時間で判定するアルゴリズムを提案する．この変換可能性判定アルゴリズムにより，ある暗号方式が変換可能か不可能かのみを判定するために指数時間かかっていたという問題を解決させる．

2 依存関係グラフ

判定アルゴリズムには依存関係グラフ $\mathcal{G} = (V, E)$ が入力される．これは暗号方式内での変数同士の依存関係

* 京都大学大学院　情報学研究科　社会情報学専攻, 京都府京都市左京区吉田本町, Kyoto University, Yoshidahonmachi, Kyoto, Japan.
† NTT セキュアプラットフォーム研究所, 東京都武蔵野市, NTT Secure Platform Laboratory, Musashino, Tokyo, Japan.
‡ NICT ネットワークセキュリティ研究所, 東京都小金井市, Security Fundamentals Lab, NSRI, NICT, Koganei, Tokyo, Japan.

と，暗号方式内でペアリングの計算に用いられる変数の依存関係を示している．ある変数を計算するために必要な変数がある場合，それぞれの変数のノードを追加し，計算に必要な変数から計算される変数へエッジを追加する．例えば，暗号方式内で $C := A \cdot B$ という計算を行う場合は，$A, B, C \in V$，$(A, C), (B, C) \in E$ となる．ペアリングに用いられる変数がある場合，それぞれの変数のノードとペアリングノードを追加し，計算に必要な変数からペアリングノードへエッジを追加する．ペアリングノードは定義順に添字が割り振られた２つのノード $P_1[0]$，$P_1[1]$ として追加する．例えば，暗号方式内で $e(A, B)$ という計算を行う場合は，$A, B, P_1[0], P_1[1] \in V$，$(A, P_1[0]), (B, P_1[1]) \in E$ となる．全てのペアリングノード $P_i[\cdot]$ は $(\cdot, P_i[\cdot]) \in E$ となるエッジを必ずそれぞれ１つだけ持つ．また，$(P_i[\cdot], \cdot) \notin E$ である．

この依存関係グラフを２つのグラフ $\mathcal{G}_0 = (V_0, E_0)$, $\mathcal{G}_1 = (V_1, E_1)$ に分割させる．分割してできたグラフ $\mathcal{G}_0, \mathcal{G}_1$ はそれぞれ $\mathbb{G}, \tilde{\mathbb{G}}$ 上での変数同士の依存関係とペアリングの計算の依存関係を表す．つまりタイプ３のペアリング群に対応した暗号方式を構成することができる．ある依存関係グラフが変換可能であるかを判定するためには，二重化禁止ノード $V_p \subseteq V$ を指定した上で，依存関係グラフが分割可能であれば良い．

定義 1 (依存関係グラフが分割可能である)
依存関係グラフを $\mathcal{G} = (V, E)$ とし，ペアリングノードを $P = \{P_1[0], P_1[1], \ldots, P_{n_p}[0], P_{n_p}[1]\} \subset V$ とし，二重化禁止ノードを $V_p \subseteq V$ とする．依存関係グラフが分割可能であるとは以下の条件を全て満たすということである．

1. *$\mathcal{G}_0, \mathcal{G}_1$ を統合すると $\mathcal{G}$ となる．*
2. *For $i = 0, 1$ とし，$X \in V_i$ とする全てのノード X に対して $\mathsf{Anc}(\mathcal{G}, X) \subseteq \mathcal{G}_i$ である．*[1]
3. *For $i = 1, \ldots, n_p$ とし，*
   $P_i[0]$ と $P_i[1]$ は別々に V_0, V_1 に含まれる．
4. *$V_0 \cap V_1 \cap V_p = \emptyset$ である．*

ここで，$X \in \mathcal{G}$ に対する $\mathsf{Anc}(\mathcal{G}, X)$ とは，ノード X に到達できる経路を持つ部分グラフとなる．この部分グラフのノードを X の祖先と呼び，X 自身も祖先に含まれる．

3　変換可能性判定アルゴリズム

変換可能性判定アルゴリズムは依存関係グラフ $\mathcal{G} = (V, E)$，ペアリングノードの集合 $P \subset V$，二重化禁止ノード $V_p \subseteq V$ の集合を入力とし，以下のアルゴリズムを順次実行する．$V = \{X_1, \ldots, X_k\}$，$V_p = \{Y_1, \ldots, Y_m\}$，$P = \{P_1[0], P_1[1], \ldots, P_{n_p}[0], P_{n_p}[1]\}$ とする．最終的に分割可能性判定アルゴリズムが出力した結果が 1 であれば依存関係グラフは変換可能であると出力し，0 であれば変換不可能であると出力する．

1. 禁止ノード統合アルゴリズム
2. グラフ単純化アルゴリズム
3. 分割可能性判定アルゴリズム

3.1　禁止ノード統合アルゴリズム

依存関係グラフが分割可能であることの条件 2. によって，あるノード X の祖先 $\mathcal{G}' := \mathsf{Anc}(\mathcal{G}, X)$ に含まれるノードは，全て同じ側のグラフに配置する必要がある．さらに条件 4. によって，$\mathcal{G}'$ に２つ以上の禁止ノードが含まれる場合，それらの禁止ノードは全て同じ側のグラフに配置されねばならない．よって，それらの禁止ノードを一つのノードに統合する．

依存関係グラフ $\mathcal{G}$，二重化禁止ノードの集合 V_p，ペアリングノードの集合 P を入力とする．

1. For $i = 1, \ldots, k$ として以下の手順を繰り返す．
   (a) $\mathcal{G}' = (V', E') \leftarrow \mathsf{Anc}(\mathcal{G}, X_i)$
   (b) $V'_p \leftarrow V_p \cap V'$
   (c) $\#|V'_p| \leq 1$ の場合，次の繰り返しに移る．
   そうでなければ下の手順へ続ける．
   (d) $V'_p = \{Y'_1, \ldots, Y'_n\}$ とし，
   For $j = 2, \ldots, n$ として以下の手順を繰り返す．
      (i). $E \leftarrow E \setminus \{(Y'_1, Y'_j), (Y'_j, Y'_1)\}$
      (ii). $V \setminus \{Y'_1, Y'_j\}$ の全てのノード Z について，
      $(Z, Y'_j) \in E$ が存在する場合，
      $E \leftarrow E \setminus \{(Z, Y'_j)\}$ とし，
      $E \leftarrow E \cup \{(Z, Y'_1)\}$ とする．
      (iii). $V \setminus \{Y'_1, Y'_j\}$ の全てのノード Z について，
      $(Y'_j, Z) \in E$ が存在する場合，
      $E \leftarrow E \setminus \{(Y'_j, Z)\}$ とし，
      $E \leftarrow E \cup \{(Y'_1, Z)\}$ とする．
      (iv). $V \leftarrow V \setminus \{Y'_j\}$
      (v). $V_p \leftarrow V_p \setminus \{Y'_j\}$
2. $\mathcal{G} = (V, E)$，V_p，P を出力する

補題 1
禁止ノード統合アルゴリズムに入力する $\mathcal{G}, V_p$ が分割可能である場合，出力する $\mathcal{G}, V_p$ は分割可能であり，入力

[1] [2] では $V_i \setminus P$ としているが，証明の便宜上 P も含ませる．

する $\mathcal{G}, V_p$ が分割可能でない場合，出力する $\mathcal{G}, V_p$ は分割可能でない．

証明：入力する $\mathcal{G}, V_p$ が分割可能である場合について考える．全ての条件を満たす $\mathcal{G}_0 = (V_0, E_0), \mathcal{G}_1 = (V_1, E_1)$ が存在する．$\mathcal{G}, \mathcal{G}_0, \mathcal{G}_1$ に対して禁止ノード統合アルゴリズムを適用して得られるグラフをそれぞれ $\mathcal{G}' = (V', E')$, $\mathcal{G}'_0 = (V'_0, E'_0), \mathcal{G}'_1 = (V'_1, E'_1)$ とする．条件 1. より，$V_0 \cup V_1 = V, E_0 \cup E_1 = E$ である．アルゴリズムでは禁止ノードに対して操作を加えるが，$V_0 \cap V_1 \cap V_p = \emptyset$ であるため，V_0, V_1 に対して操作を加えるノードは排他的である．また，V の全てのノード X について $\mathsf{Anc}(\mathcal{G}, X) \cap V_p$ は条件 2. と条件 4. より必ず V_0 か V_1 の部分集合となるので，V に対して操作を加えるノードは V_0, V_1 に対して操作を加えるノードのどちらかに属する．つまり，V_0, V_1 に対して操作されるノードを足したものと，$V_0 \cup V_1 = V$ である V に対して操作されるノードとは同じとなる．よって，$V'_0 \cup V'_1 = V'$ となる．さらに，アルゴリズムでは削除される禁止ノード $X \in V_p$ に対して $(\cdot, X), (X, \cdot)$ に関するエッジに対して操作を加えているが，$V_0 \cap V_1 \cap V_p = \emptyset$ であるため，操作を加えるエッジは排他的である．また，ノードと同様に E に対して操作を加えるエッジは E_0, E_1 に対して操作を加えるノードのどちらかに属する．つまり，E_0, E_1 に対して操作されるエッジを足したものと，$E_0 \cup E_1 = E$ である E に対して操作されるエッジとは同じとなる．よって $E'_0 \cup E'_1 = E'$ となる．$V'_0 \cup V'_1 = V'$ かつ $E'_0 \cup E'_1 = E'$ より，条件 1. が成り立つ．同様の理由で条件 2. についても成り立つ．V_0, V_1 について条件 3. が成り立つ上で，アルゴリズムではペアリングノードの削除を行わないため，V'_0, V'_1 についても条件 3. が成り立つ．$V_0 \cap V_1 \cap V_p = \emptyset$ かつ $V'_0 \subseteq V_0, V'_1 \subseteq V_1$ より，$V'_0 \cap V'_1 \cap V_p = \emptyset$ となり，条件 4. が成り立つ．以上より，入力する $\mathcal{G}, V_p$ が分割可能である場合，出力される $\mathcal{G}', V_p$ が分割可能であることを示した．

次に，入力する $\mathcal{G}, V_p$ が分割可能でない場合について考える．背理法を用いて，$\mathcal{G}, V_p$ をアルゴリズムに入力し，出力された $\mathcal{G}' = (V, E)$ が分割可能であり条件を満たす $\mathcal{G}'_0 = (V'_0, E'_0), \mathcal{G}'_1 = (V'_1, E'_1)$ が存在すると仮定する．この $\mathcal{G}'_0, \mathcal{G}'_1$ について統合したノードを元に戻して得られる $\mathcal{G}_0, \mathcal{G}_1$ が $\mathcal{G}$ 上で分割可能である条件を満たし，前提条件と矛盾することを示していく．$\mathcal{G}$ に対するノードの操作は全てのノード X から得られる $\mathsf{Anc}(\mathcal{G}, X) \cap V_p$ に対して行われ，操作を行った場合必ず $\mathcal{G}'$ は $\#|\mathsf{Anc}(\mathcal{G}', X) \cap V_p| = 1$ となる．よって，この操作を逆に行う場合は $\mathcal{G}'$ の全てのノード X から得られる $\mathsf{Anc}(\mathcal{G}', X) \cap V_p$ に対して行えば，排他的な操作としてみなすことができる．よって，$\mathcal{G}'_0, \mathcal{G}'_1$ の統合したノードを戻す場合それぞれの禁止ノードを戻す操作は排他的であると言え，$V'_0 \cap V'_1 \cap V_p = \emptyset$ を満たすため，グラフ同士でも操作は排他的であると言える．よって，$V'_0 \cup V'_1 = V'$ である V' から V に戻す操作は，V'_0 から V_0 に戻す操作と V'_1 から V_1 に戻す操作を足しあわせたものである．よって，$V_0 \cup V_1 = V$ である．次にエッジについて考える．アルゴリズムでは操作を加えるエッジは排他的であり，この操作を逆に行う場合は $\mathcal{G}'$ が $\#|\mathsf{Anc}(\mathcal{G}', X) \cap V_p| = 1$ を満たす上で，このノードに対するエッジを復元させる．よって，それぞれのグラフのノードに対応するエッジを戻す操作は排他的である．また，$V'_0 \cap V'_1 \cap V_p = \emptyset$ をみたすため，グラフ同士でもエッジを戻す操作は排他的であると言える．つまり，E'_0, E'_1 に対してエッジを戻す操作を足したものと，$E'_0 \cup E'_1 = E'$ である E' に対してエッジを戻す操作とは同じとなる．よって $E_0 \cup E_1 = E$ となる．$V_0 \cup V_1 = V$ かつ $E_0 \cup E_1 = E$ より $\mathcal{G}$ は条件 1. が成り立つ．同様の理由で条件 2. についても成り立つ．V'_0, V'_1 について条件 3. が成り立つ上で，アルゴリズムではペアリングノードの削除を行わないため，逆に操作する場合も同様である．よって V_0, V_1 についても条件 3. が成り立つ．$V'_0 \cap V'_1 \cap V_p = \emptyset$ かつ，逆操作によって復元されるノードは排他的であるため，$V_0 \cap V_1 \cap V_p = \emptyset$ となり，条件 4. が成り立つ．以上より，入力する $\mathcal{G}, V_p$ が分割可能であり，$\mathcal{G}, V_p$ は分割可能でないという前提条件と矛盾するため，$\mathcal{G}, V_p$ が分割可能でない場合は $\mathcal{G}'$, V_p は分割可能でない．

補題 2
各ペアリングノード P に対して，その祖先グラフ $\mathsf{Anc}(\mathcal{G}, P)$ は高々 1 つの禁止ノードを含む．

証明：$\mathsf{Anc}(\mathcal{G}, P)$ が 2 つ以上の禁止ノードを含む場合は必ず 1.(d) が実行される．1.(d) で，$\mathsf{Anc}(\mathcal{G}, P)$ に含まれていた全ての 2 番目以降の禁止ノードは V, V_p から削除されている．そのため，$\mathsf{Anc}(\mathcal{G}, P)$ は必ず高々 1 つの禁止ノードを含む．

補題 3
上記アルゴリズムはペアリングノードの集合に影響しない．

証明：ノード削除を行っているのは禁止ノードに対してのみである．

このアルゴリズムは最悪の計算量で $O(mk^2)$ である．

3.2 グラフ単純化アルゴリズム

二重化禁止ノード，ペアリングノード以外のノードは二重化できるため，分割可能性を判定する上で無視することができる．そこで，祖先に禁止ノードを持つようなペアリングとその祖先の禁止ノードとその関係のみを残すようにする．その後，ペアでないペアリングノードが

存在すればグラフから削除し，それによってエッジを持たない禁止ノードができれば削除する．

禁止ノード統合アルゴリズムで出力された $\mathcal{G}$，V_p，P を入力とする．

1. 空のグラフ $\tilde{\mathcal{G}} = (\tilde{E}, \tilde{V})$ を用意する．
2. $P = \{P_1, \dots, P_n\}$ とし，
   For $i = 1, \dots, n$ として以下の手順を繰り返す．
   (a) $\mathcal{G}' = (V', E') \leftarrow \mathsf{Anc}(\mathcal{G}, P_i)$
   (b) $V' \leftarrow V' \cap V_p$ とする
   (c) $\#|V'| = 0$ の場合，次の繰り返しに移る．
   $\#|V'| = 1$ の場合，下の手順へ続ける．
   (d) $Z \in V'$ ととり，
   $\tilde{V} \leftarrow \tilde{V} \cup \{P_i, Z\}$
   $\tilde{E} \leftarrow \tilde{E} \cup \{(Z, P_i)\}$ とする．
3. For $i = 1, \dots, n_p$ として以下の手順を繰り返す．
   (a) $P_i[0] \in \tilde{V}$ かつ $P_i[1] \in \tilde{V}$ の場合，次の繰り返しに移る．そうでなければ次の手順へ続ける．
   (b) $\tilde{V} \leftarrow \tilde{V} \setminus \{P_i[0], P_i[1]\}$
   (c) $P \leftarrow P \setminus \{P_i[0], P_i[1]\}$
   (d) $(Z, P_i[0]) \in \tilde{E}, (Z, P_i[1]) \in \tilde{E}$ となるエッジを $\tilde{E}$ から削除する．
4. エッジを持たないノードが存在すれば $\tilde{V}$ から削除する．
5. $V_p \leftarrow V_p \cap \tilde{V}$
6. $\mathcal{G} \leftarrow \tilde{\mathcal{G}}$ とし，$\mathcal{G}$，V_p，P を出力する

2. では，ペアリングの祖先に禁止ノードが存在すればグラフに追加している．3. では，グラフ $\tilde{\mathcal{G}}$ にペアでないペアリングノードが存在すればグラフから削除している．4. では，3. でペアリングノードが削除されたことによってできた子を持たない禁止ノードを削除している．2.(b) について，子孫から見た祖先の禁止ノード数は補題 2 より，$0 \leq \#|V'| \leq 1$ である．

補題 4
出力されたグラフ $\mathcal{G} = (V, E)$ について，$V = V_p \cup P$ であり，各ペアリングノードは一つの二重化禁止ノードからのエッジを持つグラフになる．

証明：$\tilde{V}$ に追加しているところは 2.(d) であり，追加しているのはペアリングノードと二重化禁止ノードのみである．$\tilde{E}$ に追加しているところも 2.(d) であり，Z からペアリングノードへのエッジとなっている．Z は 2.(b) より V_p の要素なので，二重化禁止ノードからペアリングノードへのエッジということになる．3. でペアリングノードを削除することによってペアリングノードへのエッジを持たない二重化禁止ノードが残ることがあるが，4. でそのようなノードを削除している．

補題 5
グラフ単純化アルゴリズムに入力する $\mathcal{G}, V_p$ が分割可能である場合，出力する $\mathcal{G}, V_p$ は分割可能であり，入力する $\mathcal{G}, V_p$ が分割可能でない場合，出力する $\mathcal{G}, V_p$ は分割可能でない．

証明：入力する $\mathcal{G}, V_p$ が分割可能である場合について考える．全ての条件を満たす $\mathcal{G}_0 = (V_0, E_0), \mathcal{G}_1 = (V_1, E_1)$ が存在する．$\mathcal{G}$ に対してグラフ単純化アルゴリズムを適用して得られるグラフを $\mathcal{G}' = (V', E')$ とする．以下のアルゴリズムを導入する．

AlgorithmA($\mathcal{G}$,$\mathcal{G}'$)

1. $V \leftarrow V \cap V'$
2. $E \leftarrow \emptyset$
3. V の全てのノード X について，
   $(\cdot, X), (X, \cdot) \in E'$ となる全てのエッジを E に追加する．
4. $\mathcal{G}$ を出力する．

以上のアルゴリズムを用いて，それぞれ
$\mathcal{G}_0' := \mathrm{AlgorithmA}(\mathcal{G}_0, \mathcal{G}')$，$\mathcal{G}_1' := \mathrm{AlgorithmA}(\mathcal{G}_1, \mathcal{G}')$，とする．

$\mathcal{G}$ に対してノードの削除が行われた場合，V から要素が削除され，V' となる．削除された要素の集合を V_r とすると，$V_0 \cup V_1 = V' \cup V_r$ となる．AlgorithmA を通すことによって，V_0，V_1 からそれぞれ V_r の要素が削除されるので，$V_0' \cup V_1' = V'$ となる．グラフ単純化アルゴルズムの 1. でエッジを空とし，2. でノードと繋がるエッジを追加し，3. ではノードの削除を行った場合そのノードに関するエッジを削除している．よって，E' があるエッジ (X, Y) をもつ場合，必ず $X, Y \in V'$ である．よって $V_0' \cup V_1' = V'$ である場合，AlgorithmA によって，E_0', E_1' は V' の全てのノードに関するエッジを網羅していることになり，E' は V' の全てのノードに関するエッジを網羅しているので，$E_0' \cup E_1' = E'$ である．よって，条件 1. を満たす．アルゴリズム 2. であらゆるペアリングノードの祖先について着目し，二重化禁止ノードが存在すればペアリングノードと二重化禁止ノードを追加し，祖先関係についてはエッジを追加することで残している．よって，V' のあらゆるノードと V' のあらゆるノード同士の祖先関係は全て保たれている．よって，V_0, V_1 が条

件 2. を満たす上で，AlgorithmA で $V \leftarrow V \cap V'$ とした場合も V_0', V_1' はあらゆる祖先関係が保たれている．よって，条件 2. を満たす．アルゴリズムの 2. ではペアリングノードを抽出し，3. ではその中からペアとなるペアリングノード以外のペアリングノードを削除している．よって，V' は必ずアルゴリズムを通した後でもペアとなるペアリングノードは存在している．そして，V_0 に含まれていたペアリングノードが V_1' に，V_1 に含まれていたペアリングノードが V_0' に含まれることは AlgorithmA で $V \leftarrow V \cap V'$ としている以上，あり得ない．よって条件 3. を満たす．同様の理由により，条件 4. を満たす．以上より，入力する $\mathcal{G}, V_p$ が分割可能である場合，出力される $\mathcal{G}', V_p$ が分割可能であることを示した．

次に，入力する $\mathcal{G}, V_p$ が分割可能でない場合について考える．背理法を用いて，$\mathcal{G}$, V_p をアルゴリズムに入力し，出力された $\mathcal{G}' = (V, E)$ が分割可能であり条件を満たす $\mathcal{G}_0' = (V_0', E_0'), \mathcal{G}_1' = (V_1', E_1')$ が存在すると仮定する．この $\mathcal{G}_0', \mathcal{G}_1'$ について以下で定義する AlgorithmB にそれぞれ AlgorithmB$(\mathcal{G}_0', \mathcal{G})$，AlgorithmB$(\mathcal{G}_1', \mathcal{G})$ と入力して得られる $\mathcal{G}_0, \mathcal{G}_1$ に対して，さらに
$\mathcal{G}_0 \leftarrow AlgorithmC(\mathcal{G}_0, \mathcal{G}_1, \mathcal{G})$ とした，$\mathcal{G}_0, \mathcal{G}_1$ が $\mathcal{G}$ 上で分割可能である条件を満たし，前提条件と矛盾することを示していく．

AlgorithmB($\mathcal{G}'$,$\mathcal{G}$)

1. V' の全てのノード X について，
   $V' \leftarrow V' \cup \mathsf{Anc}(\mathcal{G}, X)$ とする．
2. $E' \leftarrow \emptyset$
3. V' の全てのノード X について，
   $(\cdot, X), (X, \cdot) \in E$ となる全てのエッジを E' に追加する．
4. $\mathcal{G}'$ を出力する．

AlgorithmC($\mathcal{G}_0, \mathcal{G}_1, \mathcal{G}$)

1. $V \leftarrow V \setminus (V_0 \cup V_1)$
2. $E \leftarrow E \setminus (E_0 \cup E_1)$
3. V の全てのペアとなるペアリングノード $P[0], P[1]$ についてそれぞれ以下の手順を行う．
   (a) $(\tilde{V_0}, \tilde{E_0}) := \mathsf{Anc}(\mathcal{G}, P[0])$
   (b) $(\tilde{V_1}, \tilde{E_1}) := \mathsf{Anc}(\mathcal{G}, P[1])$
   (c) $V_0 \leftarrow V_0 \cup \tilde{V_0}$，$E_0 \leftarrow E_0 \cup \tilde{E_0}$
   (d) $V_1 \leftarrow V_1 \cup \tilde{V_1}$，$E_1 \leftarrow E_1 \cup \tilde{E_1}$
   (e) $V \leftarrow V \setminus \tilde{V_0}$，$E \leftarrow E \setminus \tilde{E_0}$
   (f) $V \leftarrow V \setminus \tilde{V_1}$，$E \leftarrow E \setminus \tilde{E_1}$
4. $V_0 \leftarrow V_0 \cup V$ とする．
5. V の全てのノード X について，
   $(\cdot, X), (X, \cdot) \in E$ となる全てのエッジを E_0 に追加する．

条件 1. より，$V_0' \cup V_1' = V'$ であり，グラフ単純化アルゴリズムはノードを削除するのみで，追加を行わないので，$V' \subseteq V$ である．よって，AlgorithmB によって V_0', V_1' の要素を取り，V を参照して祖先ノードを V_0, V_1 に追加しているので，$V_0 \cup V_1 \subseteq V$ である．この時，AlgorithmC に $(V \setminus (V_0 \cup V_1), E \setminus (E_0 \cup E_1))$ を入力することによって，$V_0 \cup V_1$ としても足りない要素を V_0, V_1 に追加しているので，必ず $V_0 \cup V_1 = V$ を満たす．エッジに関しては，グラフ単純化アルゴリズムでエッジの削除，追加を行っているが，AlgorithmB によって，自分のグラフの全てのノードの全てのエッジについて復元している．さらに，AlgorithmC で，足りなかったノードのエッジについても復元している．よって，$E_0 \cup E_1 = E$ を満たし，条件 1. を満たす．同様の理由により，AlgorithmB を通した後の V_0, V_1 のノードについては条件 2. を満たす．AlgorithmB を通しても $V_0 \cup V_1$ としても足りないノード，エッジについては，AlgorithmC で追加しているが，これは足りないノード，エッジに全ての関係を追加するので，足りないノード X 同士で祖先関係があった場合，必ず祖先となるノードも追加されているので，条件 2. を満たす．V_0', V_1' が条件 3. を満たす上で AlgorithmB, AlgorithmC を通しても，V_0, V_1 の条件 3. を満たすペアリングノードについては削除されることも移動することもないので必ず条件 3. を満たす．前述のとおり，$V_0 \cup V_1$ としても V に足りない要素がある．そのような要素のうち，更にペアリングノードについて，$V \setminus (V_0 \cup V_1) \cap P$ とすることで得られる．このようにして得られたペアリングノードについて，AlgorithmC の 3. でそれぞれのペアリングノードを別々に V_0, V_1 に追加している．よって，条件 3. を必ず満たす．まず，V_0', V_1' の要素について考える．この要素は条件 4. を満たすので V_0, V_1 の V_0', V_1' の要素は条件 4. を満たす．AlgorithmB の 1. によって，V_0', V_1' には含まれない要素が追加された場合，それには禁止ノードは含まれない．何故なら，グラフ単純化アルゴリズムによって得られた $\mathcal{G}'$ は補題 4 によって，ノードの祖先を取ったときに既に禁止ノードが必ず含まれており，入力される $\mathcal{G}$ は補題 2 により，祖先の禁止ノードが高々 1 つだからである．よって，ここでは禁止ノードについて考える必要がない．次に，AlgorithmC の 3. について着目する．この時，ペアリングノードの祖先を別々に V_0, V_1 に追加しているため，ノードが重なることがない．AlgorithmC の 4. では，残りの全てのノードについ

て V_0 に追加しているため，同様にノードが重なることがない．よって，V_0, V_1 は条件 4. を満たす．以上より，入力する $\mathcal{G}, V_p$ が分割可能であり，$\mathcal{G}, V_p$ は分割可能でないという前提条件と矛盾するため，$\mathcal{G}, V_p$ が分割可能でない場合は $\mathcal{G}'$, V_p は分割可能でない．

このアルゴリズムは最悪の計算量で $O(n_p)$ である．

3.3 分割可能性判定アルゴリズム

グラフ単純化アルゴリズムで出力された $\mathcal{G} = (V, E)$, V_p，P を入力とし，このグラフが分割可能かどうかの判定を行う．ただし，グラフ単純化アルゴリズムによってペアリングノードが削除されていることがあるので，そのような場合は $n_p := \#|P|/2$ とし，
$P = \{P_1[0], P_1[1], \ldots, P_{n_p}[0], P_{n_p}[1]\}$ として添字を振り直す．

まずは，明らかに入力されたグラフが分割不可能な場合について判定する．

重複ペアリングノード判定アルゴリズム

1. For $i = 1, ..., n_p$ として以下の手順を繰り返す．
   (a) あるノード X について $(X, P_i[0]), (X, P_i[1])$ が E に存在すれば 0 を出力して終了

補題 6
重複ペアリングノード判定アルゴリズムで *0* を出力しなかったグラフにおいて，グラフに含まれる全ての二重化禁止ノードは必ずペアとなるペアリングノードを子孫に持たない．

証明：グラフにペアとなるペアリングノードを子孫に持つ二重化禁止ノードが存在していたとする．重複ペアリングノード判定アルゴリズムでは，そのような二重化禁止ノードが存在していた場合，必ず 0 を出力する．よって前提条件と矛盾し，以上の補題が成り立つ．

ここで，グラフを 2 つのグラフに分離するという表現を，色分けとして表現していく．片方へ割り振ることを白，もう片方へ割り振ることを黒とする．二重化禁止ノードは子が異なるグラフへと割り振られると分割不可能となる．つまり，二重化禁止ノードの子は全て同じ色でなければならない．また，ペアリングノードはペアとなるノードと対の関係でなければならない．つまり，$P_i[b]$ のノードを黒色にすると，もう片方の $P_i[(b+1) \bmod 2]$ のノードは白色にしなければならない．このように，2 つの制約が存在し，この制約同士が衝突することによって分割可能かどうかが決まる．この単純化されたグラフから，制約関係について着目した無向グラフを生成する．

無向グラフ生成アルゴリズム

1. 空の無向グラフ $\mathcal{G} = (E, V)$ を用意する．
2. For $i = 1, \ldots, n_p$ として以下の手順を繰り返す．
   (a) $P_i[0], P_i[1]$ のノードのそれぞれの親 X, Y について，
   $V \leftarrow V \cup \{X, Y\}$
   $E \leftarrow E \cup \{(X, Y)\}$ とする．
3. $\mathcal{G}$ を出力する．

補題 7
無向グラフ $\mathcal{G} = (E, V)$ について，あるノード X から X に向かうようなエッジを持たない．

証明：無向グラフ生成アルゴリズムの 1. でエッジを初期化し，2.(a). でエッジを追加している．補題 6 によって，ペアリングノードの親 X, Y が $X = Y$ となることはない．よって，(X,X) となるようなエッジが追加されない．

ペアリングノードが排除され，その代わりに二重化禁止ノード同士のエッジとして情報が残されている．ここでいう情報とは，お互いの色は異なる色でなければならないという制約情報である．この無向グラフは，エッジで繋がれた同士は異なる色でなければならないという制約がある．これはグラフ理論における 2 点彩色問題そのものである．つまり，2 点彩色可能であれば，分割可能であり，2 点彩色不可能であれば分割不可能であるといえる．2 彩色可能なグラフはそのグラフが 2 部グラフであるということが証明されている．さらに，2 部グラフであることと奇閉路を持たないこととは同値であることが証明されている．そこで，上で生成した無向グラフが奇閉路を持つか持たないかを判定するアルゴリズムを示す．これは補題 7 が成り立つことを前提とするアルゴリズムである．

まず，順次色付け関数を定義する．これは再帰関数となっている．colors は，それぞれのノードに対してどの色が塗られたかという情報が格納された配列である．まだ色が塗られていない場合は null が入っている．

fill node(node,color)

1. colors[node] が null ではなく，color と異なる場合，奇閉路を持つと出力し，終了する．
2. colors[node] が null の場合，colors[node] $\leftarrow$ color とする．
3. color $\leftarrow$ (color + 1) mod 2
4. $(node, X) \in E$ となる $X_1, \ldots, X_n$ について
   それぞれ順に fill node(X_i, color) を呼び出す．

以上の fill node 関数を用いて以下のアルゴリズムを構成する．$V = \{X_1, \dots, X_n\}$ とする．

奇閉路判定アルゴリズム

1. For $i = 1, \dots, n$ として以下の手順を繰り返す．
   (a) colors[X_i] が null の場合，fill node(X_i,0) を呼び出す．
2. 奇閉路を持たないと出力する．

このアルゴリズムによって，グラフが奇閉路を持つか持たないかを判定することができる．そして，前述の通り，奇閉路を持たない場合は分割可能なので 1 を出力し，奇閉路を持つ場合は分割不可能なので 0 を出力する．

補題 8

分割可能性判定アルゴリズムに入力する $\mathcal{G}, V_p$ が分割可能である場合，*1* が出力され，入力する $\mathcal{G}, V_p$ が分割可能でない場合，*0* が出力される．

証明：入力する $\mathcal{G}, V_p$ が分割可能である場合について考える．全ての条件を満たす $\mathcal{G}_0 = (V_0, E_0), \mathcal{G}_1 = (V_1, E_1)$ が存在する．アルゴリズム内で得られた無向グラフを $\mathcal{G}'$ とする．重複ペアリングノード判定アルゴリズムに分割可能なグラフを入力して 0 を出力すると仮定すると，ある禁止ノード X について $(X, P_i[0]), (X, P_i[1])$ となるエッジが存在することになる．条件 3. より，V_0, V_1 のどちらかに $P_i[0], P_i[1]$ が 1 つずつ含まれているはずだが，条件 2. により，どちらの V_0, V_1 にも二重化禁止ノード X を含むこととなり，条件 4. に反し，$\mathcal{G}_0, \mathcal{G}_1$ は $\mathcal{G}$ の有効な分割グラフでなくなり，前提条件に矛盾する．よって，重複ペアリングノード判定アルゴリズムに分割可能なグラフを入力して 0 を出力して終了することはない．次に，無向グラフ生成アルゴリズムによって無向グラフが生成される．補題 4 を満たすグラフに対して 2. を行うため，この無向グラフのノードは全て禁止ノードであり，それぞれのペアとなるペアリングノードの親同士でエッジをつないでいる．ここで，V_0 に含まれる禁止ノードの色を 0，V_1 に含まれる禁止ノードの色を 1 とする．そうすると，条件 1. と条件 4. より，色分けが重なることはなく，全ての禁止ノードについて色が付けられる．このとき，必ずエッジで繋がれた二重化禁止ノード同士は異なる色となる．つまり，どちらかのノードが V_0 に含まれ，もう片方が V_1 に含まる．これは条件 3. を満たすため，必ずペアとなるペアリング同士は V_0, V_1 のどちらかのグラフに含まれ，条件 1. より，親となる二重化禁止ノードもどちらかのグラフに含まれるからである．よって，この無向グラフは 2 彩色可能であり，2 彩色可能であれば奇閉路を持たないので，奇閉路判定アルゴリズムは 1 を出力する．次に，入力する $\mathcal{G}, V_p$ が分割可能でない場合，0 が出力されることを示す．背理法を用いて，$\mathcal{G}, V_p$ をアルゴリズムに入力し，1 が出力された場合，$\mathcal{G}, V_p$ は分割可能であり条件を満たす $\mathcal{G}_0 = (V_0, E_0), \mathcal{G}_1 = (V_1, E_1)$ が存在することを示し，前提条件と矛盾することを示していく．1 が出力されるということは，生成された無向グラフを奇閉路判定アルゴリズムに入力し，奇閉路を持たないと出力したということである．つまり，生成された無向グラフは 2 彩色可能であり，それぞれの二重化禁止ノードに 0 か 1 の色を付けることができ，エッジで繋がれた二重化禁止ノードは全て異なる色である．ここで，空のグラフ $\mathcal{G}_{\prime}, \mathcal{G}_{\infty}$ を生成し，0 と色を付けられている二重化禁止ノードを V_0 に追加し，1 と色を付けられている二重化禁止ノードを V_1 に追加する．さらに，それぞれのグラフ V_i の全ての二重化禁止ノード X について，$(X, P) \in E$ となるエッジが存在すれば E_i に追加し，P を V_i に追加する．この $\mathcal{G}_{\prime}, \mathcal{G}_{\infty}$ が分割可能であることを示していく．補題 4 を満たすグラフに対して 2. を行うため，この無向グラフのノードは全て禁止ノードであり，V の全ての禁止ノードを含む．そして，このノードに関しては V_0, V_1 のどちらかに全て追加した．さらに，全ての禁止ノードを親に持つペアリングノードをそれぞれのグラフに追加した．補題 4 を満たすため，V 全てのペアリングノードが V_0, V_1 に追加される．エッジについても補題 4 より，全ての二重化禁止ノードからのエッジで全てのエッジが含まれる．よって，E_0, E_1 に全てのエッジが含まれる．よって，$V_0 \cup V_1 = V, E_0 \cup E_1 = E$ を満たし，条件 1. を満たす．また，ペアリングノードに着目しても $(\cdot, P) \in E$ となるエッジは全て E_0, E_1 に追加されており，補題 4 を満たすため，必ず条件 2. を満たす．無向グラフにおけるエッジは，ペアとなるペアリングノードを子に持つ二重化禁止ノード同士 X, Y で繋がれている．また，繋がれている二重化禁止ノードは全て異なる色なので，必ずどちらかが V_0 に含まれ，片方が V_1 に含まれる．補題 4 より，それぞれの二重化禁止ノードはペアリングノードを子として持っており，無向グラフ上でエッジで繋がっているということは，片方が $P_i[0]$ を子として持ち，片方が $P_i[1]$ を子として持つ．補題 7 より，$X = Y$ となることもない．よって，条件 3. を満たす．無向グラフのノードが全ての二重化禁止ノードであり，それぞれのノードの色が 0 なら V_0，色が 1 なら V_1 に追加としているので，必ず条件 4. を満たす．以上より，入力する $\mathcal{G}, V_p$ が分割可能であり，$\mathcal{G}, V_p$ は分割可能でないという前提条件と矛盾するため，$\mathcal{G}, V_p$ が分割可能でない場合は 0 を出力する．

このアルゴリズムは最悪の計算量で $O(n_p + m)$ である．

3.4 考察

以上の補題 1 から補題 8 までの全ての補題が成り立つことによって，以下の定理が成り立つ．

定理 1
入力される依存関係グラフ $\mathcal{G} = (V, E)$，二重化禁止ノード $V_p \subseteq V$，ペアリングノード $P \subseteq V$ が変換可能な場合，変換可能性判定アルゴリズムは変換可能であると出力する．変換不可能な場合，変換可能性判定アルゴリズムは変換不可能であると出力する．

全てのアルゴリズムを実行したとき，最悪の計算量で $O(n_p + mk^2)$ となる．n_p はペアリングノード数，m は二重化禁止ノード数，k は全てのノード数である．よって，変換可能性判定アルゴリズムは多項式時間で終了するアルゴリズムである．

実際にこのアルゴリズムを実装し，標準的な PC (Windows 7, Intel(R) Core(TM) i7-3720QM CPU@ 2.60GHz, 8.0GB RAM) で計算を行った．Dual System Encryption (Waters09)[4] において，DLIN 仮定と DBDH 仮定でインスタンスとして与えられる変数を禁止ノードに指定する．この依存関係グラフと禁止ノードを既存の最適な変換を求めるアルゴリズムに入力したところ，計算が終了するまでに約 5800 秒かかり，分割可能なパターンは一つもないと出力された．つまり変換不可能ということである．対して，分割可能性判定アルゴリズムに入力したところ，計算が終了するまでに 31 ミリ秒かかり，変換不可能であると出力された．このように，単純に比較すると，今回の場合は約 19 万倍の速度で計算することが出来た．最適な変換を求めるアルゴリズムでは，さらにペアリングノードや次のエッジを持たないボトムノードの個数が増えると，指数的に計算時間が増大してしまうが，分割可能性判定アルゴリズムでは多項式的に増大する．

4 今後の展望

分割可能性判定アルゴリズム内で用いた無向グラフは，[3] の最適な変換パターンを求めるアルゴリズムにも用いることができる．最適な変換パターンを求めるアルゴリズムでは，二重化禁止ノードによって分割不可能な変換パターンについても全て計算するため，計算量が膨大なものとなっているが，分割不可能な変換パターンによってはこの無向グラフによって削減することによって，大幅に計算量を削減することができる．このように，多項式時間で得られた無向グラフを入力して，試行する変換パターンを抑えられるアルゴリズムを提案していきたい．

5 おわりに

楕円曲線上で定義されるペアリングを用いることによって多くの暗号方式が提案されてきたが，離散対数問題アルゴリズムが進展してきたことにより，タイプ 1 のペアリング群を用いた暗号方式は計算効率が悪くなってしまった．タイプ 3 ではその影響を受けないため，サイズを抑えられる．そこで，あるタイプ 1 のペアリング群で構成された暗号方式を，手順を維持したままタイプ 3 の暗号方式に変換するアルゴリズムが実装，提案された．この変換アルゴリズムでは，可能な変換パターンについて全通り調べ，全パターンのうち最も計算効率が良い変換パターンを抽出し出力するが，入力する暗号方式のペアリングノード数に対して計算量が指数時間となってしまうという問題があった．入力される暗号方式によっては，全パターンに対して分割不可能であり，暗号方式を変換することが出来ない場合があるが，このように変換可能か不可能かのみを判定したい場合でも全パターン調べる必要性がある．そこで，依存関係グラフと二重化禁止ノードを入力することによって，その依存関係グラフが変換可能か不可能かを判定するアルゴリズムを提案し，そのアルゴリズムが正しく動作することを証明した．

参考文献

[1] S. D. Galbraith, K. G. Paterson, and N. P. Smart. “Pairings for cryptographers.” Discrete Applied Mathematics, 156(16):3113-3121, 2008.

[2] Masayuki Abe, Jens Groth, Miyako Ohkubo and Takeya Tango. “Converting Cryptographic Schemes from Symmetric to Asymmetric Bilinear Groups.” Advances in Cryptology — CRYPTO 2014, pages 241-260. 2014.

[3] Takeya Tango, Masayuki Abe, and Tatsuaki Okamoto. “Implementation of Automated Translation for Schemes on Symmetric Bilinear Groups.” SCIS2014. 2014.

[4] B. Waters. “Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions.” CRYPTO 2009, LNCS 5677, pp. 619—636, 2009.