SCIS 2019 2019 Symposium on
Cryptography and Information Security
Shiga, Japan, Jan. 22 - 25, 2019
The Institute of Electronics,
Information and Communication Engineers

ペアリング積等式の一括検証の最適化
Optimization of Batch Verification for Pairing Product Equations

星野 文学 *†
Fumitaka Hoshino

あらまし 高機能暗号方式を構成する為の要素技術として Groth-Sahai 証明 [1] や群構造維持署名 [2] の研究が盛んに行われている．そのような方式では, 大量のペアリング積等式の検証がしばしば必要となる．それらの等式は各個に検証するより一括して検証する方が効率的である．しかし, ペアリング積等式の一括検証には沢山の計算方法が存在し, その数は一般にペアリングの数に対して指数関数的に増大する．そして計算方法により計算の効率が大きく変わる事が知られている．指数的多数の計算方法の候補から, 最も良い計算方法を単純に探索する事は, 一括検証の規模が大きくなると事実上実行不可能となる．本研究の目的は, 上記のように大量にある各々の検証式の型が全て既知である場合, 指数的多数の計算方法の候補から最も良い, あるいはそれに匹敵するくらい良い計算方法を見つけ出すアルゴリズムを与える事である．

Keywords: pairing product equation, batch verification, Groth-Sahai proof, integer programming

1 はじめに

Groth-Sahai 証明 (Groth-Sahai proof, GS proof) [1] や群構造維持署名 (structure preserving signature, SPS) [2] の検証では, ペアリングや変数の個数等が必ずしも同じ型とは限らない大量のペアリング積等式を評価する事がある．この際に一括検証 (batch verification) を導入し, 検証の高速化を図りたい．しかしながら, 実はペアリング積等式の一括検証には沢山の計算方法が存在し, その数は一般にペアリングの数に対して指数関数的に増大する．そして最適な計算方法は, 元の検証式に含まれるパラメータや変数および関連する演算の回数や速度比に強く依存するため, 見つけ出すのが容易ではない．しかも, これらの方式はさらに上位の方式から呼び出される抽象度の高い暗号プリミティブ, 即ち暗号モジュールとして使用することが意図されている．複数の暗号モジュールを組み合わせて目的の方式を得る暗号方式のモジュール設計 (modular design) においては, 最適な計算方法は目的の方式毎に変化し得る．従って, GS proof や SPS を組み合わせて何らかの大規模な方式が設計された後で, それを何らかのシステムに応用する直前に最適な一括検証のアルゴリズムが自動合成され, 使用される事が望ましい．

* Secure Platform Laboratories, NTT Corporation, Japan
† School of Computing, Tokyo Institute of Technology, Japan

1.1 関連研究

1989 年, Fiat は RSA の法演算の計算量を削減する為 RSA の変種である Batch RSA を考案した [3]．これが暗号学で一括計算 (batching) が検討された最初の研究の一つと言われている．1998 年, Bellare, Garay および Rabin は冪乗に基づく等式の一括検証 (batch verification) のテクニックを体系的に研究し, 3 つの汎用的な方法 random subset test, small exponents test, および bucket test に分類した [4]．2009 年, Ferrara らはペアリングに基づく等式の集合を安全に一括検証する方法を研究した [5]．2010 年, Blazy らは一括検証のテクニックを GS proof に応用し, その検証コストを大きく削減した [6]．2017 年, Herold らは GS proof の構造に着目し, Schwartz-Zippel の補題を使った新しい一括検証のテクニックを開発した [7]．これらの研究では, 何らかの証明系や署名方式を一つ決定し, その方式に対して, ある効率の良い一括検証方法が決定され, そこから具体的一括検証方法を手で書き下すという事が想定されている．しかし前述のように, 暗号方式のモジュール設計の考え方においては, 方式を設計する度に一括検証方法を検討し直すのは幾分面倒である．2012 年, Akinyele らは署名方式の一括検証のプログラムを自動生成するアルゴリズムを発表し, 実際に署名方式の高レベル記述から Python あるいは C++ のプログラムを自動生成するツール AutoBatch を公開した [8, 9]．

1.2 本研究の成果

Akinyele らの方法ではペアリング積等式等に適用できる 10 個程度の一括検証テクニックをヒューリスティックな方法で適用し一括検証のプログラムを生成する [8]. 本研究では大量のペアリング積等式の最適な一括検証を整数計画法を用いて自動合成するアルゴリズムを検討する. このような方法では, ヒューリスティックなアルゴリズムに直接は依存しないので, 適切なソルバを用いれば最適性の保証が可能となる.

2 準備

2.1 ペアリング

暗号学的に言うと, ペアリングとはいくつかの特殊な機能を持った離散対数問題の拡張であり, 暗号方式の設計において, 様々なな応用が可能な暗号プリミティブ (暗号用原始関数 (方式)) である. ペアリングは形式的には概ね次の様な代数的構造を持つ符号を出力する確率的多項式時間アルゴリズム $\mathcal{G}$ であると定義される.

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

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 λ に関する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 $e : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_T$.

一般に暗号認証技術の設計において安全性の証明を行う際には, 暗号プリミティブに対して定義される何らかの暗号学的問題を考察する. λ は安全変数と呼ばれ, 暗号学的問題を解こうと試みる確率的多項式時間攻撃者 $\mathcal{A}$ に対して定義される利得が, 如何なる $\mathcal{A}$ に対しても λ に関して無視可能となる事が暗号プリミティブが安全である事の差し当たっての定義である. 暗号プリミティブの安全性を数学的に証明する事は多くの場合は困難であり, 通常は経験的に成立すると予想される仮説が用いられる. そのような仮説を暗号学的仮定と呼ぶ. 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. 特に $\mathbb{G}_0, \mathbb{G}_1$ を入力群 (source group), $\mathbb{G}_T$ を標的群 (target group) と呼ぶ. 補助出力 aux は $\mathcal{G}$ の入力および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, 積あるいは乗算などと呼ばれ $g, h \in \mathbb{G}_x$ 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 もペアリングと呼ばれる. q を $q > 2^{\Theta(\lambda)}$ なる十分大きな素数から取り, $\mathbb{L}$ として $\mathbb{Z}/q\mathbb{Z}$ を, $\mathbb{G}_0, \mathbb{G}_1$ としてそれぞれ適当な楕円曲線の有理点群の位数 q の巡回部分群を, $\mathbb{G}_T$ として適当な有限体の乗法群の位数 q の巡回部分群を取ると上記のような $\mathcal{G}$ を具体的に構成出来る事が知られている. ペアリングに関する暗号学的仮定はそのような構成のうち, 幾つかのものが持つと予想される性質を抽象して定義され, そのような暗号学的仮定は少なくとも以下を含意する.

3. $\mathbb{L}$ を離散対数とする $\mathbb{G}_0, \mathbb{G}_1, \mathbb{G}_T$ 上の CDH 問題はそれぞれ計算困難.

従ってある $\mathcal{G}$ の具体的な構成について, 有意な確率で条件 3 を破る多項式時間アルゴリズムが見つかったならば $\mathcal{G}$ は安全なペアリングではないという事が出来る. ある $\mathcal{G}$ の具体的な構成が条件 1, 2 を満たし暗号学的仮定が成立すると信じられるなら, そのような $\mathcal{G}$ を安全なペアリングの候補と呼ぶ. 歴史的経緯と習慣により, 安全なペアリングの候補の事を単にペアリングと呼ぶ事が多い. また, 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. Galbraith らは, 暗号プロトコル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 [10].

Type 1: $\mathbb{G}_0, \mathbb{G}_1$ 間で双方向に多項式時間同型が存在.
Type 2: $\mathbb{G}_1 \to \mathbb{G}_0$ なる一方向同型が存在.
Type 3: $\mathbb{G}_0, \mathbb{G}_1$ 間で多項式時間同型が存在しない.

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, 暗号方式の設計においては両者は特に区別されず, 単に $\mathbb{G}$ と記述される.

2.2 一括検証

ペアリングを用いた, 署名やゼロ知識証明などの一般化された暗号認証方式が, 方式を構成するアルゴリズム内に

$$\prod_{n=1}^{n_{\max}} e(A_n, B_n) = \prod_{m=1}^{m_{\max}} e(C_m, D_m)$$

の型の式 (ペアリング積等式, pairing product equation, PPE) の検証を含む時, 一般にこれを

$$\prod_{n=1}^{n_{\max}} e(A_n, B_n) \times \prod_{m=1}^{m_{\max}} e(C_m^{-1}, D_m) = 1$$

の検証と見做すことが出来る. 適当な添え字の付け替えを行えば上記検証式は

$$\prod_{\ell=1}^{\ell_{\max}} e(A'_\ell, B'_\ell) = 1$$

と記述出来る. 従って以下ではこの型の検証式を考える. 今, 方式を構成するペアリングの入力群変数の集合が $X_1, \ldots, X_{i_{\max}} \in \mathbb{G}_0$, $Y_1, \ldots, Y_{j_{\max}} \in \mathbb{G}_1$ であるとき, 検証アルゴリズム内に

$$\prod_{\ell=1}^{\ell_{\max}} e(X_{i_\ell}^{\alpha_\ell}, Y_{j_\ell}^{\beta_\ell}) = 1$$

(但し $\alpha_\ell \neq 0$, $\beta_\ell \neq 0$) の型の検証式が多数含まれる場合を考える. 即ち整数 $k \in \{1, \ldots, k_{\max}\}$ について

$$E_k = \prod_{\ell=1}^{\ell_{\max,k}} e(X_{i_{k\ell}}^{\alpha_{k\ell}}, Y_{j_{k\ell}}^{\beta_{k\ell}})$$

とし $\bigwedge_{k=1}^{k_{\max}} (E_k = 1)$ なる命題の検証が検証アルゴリズムに含まれるとする. $\mathcal{R}$ を離散対数 $\mathbb{L}$ の適当な部分集合として, この命題の真理値について

$$\bigwedge_{k=1}^{k_{\max}} (E_k = 1) \quad \Rightarrow \quad \left(\prod_{k=1}^{k_{\max}} E_k^{r_k} \right) = 1, \qquad \forall k \in \{1, \ldots, k_{\max}\},\ r_k \xleftarrow{\$} \mathcal{R}$$

であるが, 逆は必ずしも成立しない. しかし暗号学的に安全なペアリングを用いる場合 $\mathcal{R}$ を十分大きく取ることが出来, 検証時に選ばれる r_k を証明者が当てる確率が小さいと仮定すると, 大きい確率で逆が成り立つとして良い. 一般に $E_k = 1$ なる命題を $k \in \{1, \ldots, k_{\max}\}$ について一つずつ検証するよりも $(\prod_{k=1}^{k_{\max}} E_k^{r_k}) = 1$ を一括で検証する方が効率が良い. これを一括検証と呼ぶ.

2.3 整数計画法

整数計画法 (integer programming, IP) [11] は変数の取りうる値の範囲が整数に制約された制約最適化問題の解法アルゴリズムの総称で, その入力例 (整数計画問題) の種類によって様々なクラスに分類される. 最も良く研究されている整数計画問題のサブクラスが整数線型計画問題で, 不等式および等式の集合を不等式系と呼ぶことにすると, その入力例は概ね目的関数と不等式系によって以下のように表現される.

$i \in \{1, \ldots, m\}$, $j \in \{1, \ldots, n\}$ を添え字として x_j を $\mathbb{Z}$ 上の変数とし, $a_{ij}, b_i, c_j \in \mathbb{R}$ を定数とする. 不等式系

$$Q: \quad \begin{aligned} \sum_j a_{ij} x_j &\geq b_i \quad (i = 1, \ldots, m'), \\ \sum_j a_{ij} x_j &= b_i \quad (i = m'+1, \ldots, m) \end{aligned}$$

による制約の元, 目的関数

$$f(x_1, \ldots, x_n) := \sum_j c_j x_j$$

を最小化せよ, あるいは最小化する $(x_1, \ldots, x_n) \in \mathbb{Z}^n$ を見つけよ.

全ての等式および不等式が変数に関して線型である不等式系を線型不等式系と呼ぶことにすると, 整数線型計画問題とは線型不等式系 Q による制約の下, 線型目的関数 f を最小とする変数の割り当て $x_j|_{j \in \{1,\ldots,n\}}$ を見つける問題と言える. 整数線形計画問題を厳密に解くアルゴリズムは例えば動的計画法 (dinamic programming) [12], 分岐限定法 (branch and bound) [13–15], 切除平面法 (cutting plane) [16], 分岐切除法 (branch and cut) [17–19] などが有名である. また, 何らかの精度保証の下で整数線形計画問題を近似的に解くアルゴリズムは例えば乱択丸め (randomized rounding) [20], 乱択抽出 (randomized sampling) [21] などが知られている. さらに, 精度保証は特に無いが, 何らかの数理的考察の下で整数線形計画問題を近似的に解くアルゴリズムについては例えば貪欲法 (greedy algorithm) [22], 局所探索法 (local search) [23–25], タブー探索 (tabu search) [26, 27], 模擬焼きなまし (simulated annealing) [28–30], 遺伝的アルゴリズム (genetic algorithm) [31,32] など膨大な量の研究が蓄積されており, 現在も精力的に研究されている. また, 特に変数の取りうる範囲が $\{0, 1\}$ に制約された整数計画問題は 0-1 整数計画問題と呼ばれる. 一般に 0-1 整数計画問題においては $\{0, 1\}$ を真理値と見なし制約式あるいは目的関数が異なる二値論理式 A および B に関する表現 $A \wedge B$

(あるいは $A \times B$) を含むとき, 以下の新しい制約を不等式系に導入する事によりそれを新しい二値変数 Z に置き換える事ができる.

$$\left\{ \begin{array}{l} X = A, \\ Y = B, \\ Z - X - Y + 1 \geq 0, \\ X - Z \geq 0, \\ Y - Z \geq 0 \end{array} \right. \tag{1}$$

ここで X および Y は, この展開を再帰的に繰り返す過程で A や B が二回以上評価されない為に導入された $\{0,1\}$ 上の補助変数であり, 二値論理式 A, B が (例えば単なる二値変数である等) それ以降評価されないなら, X, Y を使用せず直接 A, B で展開しても構わない. 同様に, 制約式あるいは目的関数が二値論理 A に関する表現 $\neg A$ を含むとき, 以下の新しい制約を不等式系に導入する事によりそれを新しい二値変数 Z に置き換える事ができる.

$$Z = 1 - A \tag{2}$$

上記 (1) および (2) を再帰的に適用する事により, 如何なる $\wedge$ (and) および $\neg$ (not) で構成される多項式長の論理式も, 最終的に新しい変数とその線型制約に多項式時間で置き換える事ができる. 従って, 如何なる二値論理の如何なる線型結合も全体で多項式長であるなら新しい変数と線型制約を導入する事により効率的に線型化することが出来る. 結果として大きなクラスの計算可能な制約および目的関数に関して, 線型不等式系 Q および線型目的関数 f, 即ち 0-1 整数線型計画問題を構成する事が出来る. 本稿では 0-1 整数線型計画問題しか使わないので, 以降整数計画法と言ったら 0-1 整数線型計画法を指し, 整数計画問題と言ったら 0-1 整数線型計画問題を指すとする.

3 課題とアプローチ

一般に $e(X,Y)^r = e(X^r,Y) = e(X,Y^r)$ であり, $e(X,Y)^r$ の計算方法は一通りではない. さらに $e(X,Y_1)^{r_1} e(X,Y_2)^{r_2} = e(X,Y_1^{r_1}Y_2^{r_2})$ あるいは $e(X_1,Y)^{r_1} e(X_2,Y)^{r_2} = e(X_1^{r_1}X_2^{r_2},Y)$ であり, 引数のどちらか一方が等しいペアリングはまとめる事が可能である. 従って一括検証式 $\prod_{k=1}^{k_{\max}} E_k^{r_k}$ には使用される乱数 r_k をどう配置し, ペアリングをどうまとめるかにより沢山の計算方法が存在し, その数は一般に少なくともペアリングの数に対して指数関数的に増大する. そして計算方法により, まとめる事が可能なペアリング数やペアリングへ入力される式の型が変化するので計算の効率が大きく変わる. 指数的多数の計算方法の候補から最も良い計算方法を単純に探索する事は, 一括検証の規模が大きくなると事実上実行不可能となる. 本研究の課題は, $k \in \{1,\ldots,k_{\max}\}$ に関して予め決められた E_k の型が与えられたとき, この指数的多数の計算方法の候補から最も良い, あるいはそれに匹敵するくらい良い計算方法を見つけ出す事である.

今, ペアリング $e(X^\alpha, Y^\beta)^r$ に対して, 便宜的に $b \in \{0,1\}$ なる変数を考え, $b = 0$ ならば冪を全て Y に集め, $b = 1$ ならば冪を全て X に集めるとする. 従って $\epsilon = \alpha\beta$, $\bar{b} = 1 - b$ とすれば

$$e(X^\alpha, Y^\beta)^r = e(X, Y^{\bar{b}\epsilon r}) \cdot e(X^{b\epsilon r}, Y)$$

と展開出来る. 今, 非対称ペアリングの場合を想定して一括検証

$$\prod_{k=1}^{k_{\max}} E_k^{r_k} = \prod_{k=1}^{k_{\max}} \prod_{\ell=1}^{\ell_{\max,k}} e(X_{i_{k\ell}}^{\alpha_{k\ell}}, Y_{j_{k\ell}}^{\beta_{k\ell}})^{r_k}$$

の全てのペアリングに上記の展開を適用すると.

$$\prod_{k=1}^{k_{\max}} E_k^{r_k} = \left(\underline{\prod_{k=1}^{k_{\max}} \prod_{\ell=1}^{\ell_{\max,k}} e(X_{i_{k\ell}}, Y_{j_{k\ell}}^{\bar{b}_{k\ell}\epsilon_{k\ell}r_k})} \right) \times \left(\prod_{k=1}^{k_{\max}} \prod_{\ell=1}^{\ell_{\max,k}} e(X_{i_{k\ell}}^{b_{k\ell}\epsilon_{k\ell}r_k}, Y_{j_{k\ell}}) \right)$$

を得る. $\mathbb{G}_0$ 変数 X_i の添え字 $i \in \{1,\ldots,i_{\max}\}$ の関数 I を

$$I : i \mapsto \{(k,\ell) : i_{k\ell} = i\}$$

と定義すると, 上記下線部について有限個のペアリングの積の演算順序を交換し, 同じ基底 X_i を持つペアリングをまとめる事により,

$$\prod_{k=1}^{k_{\max}} \prod_{\ell=1}^{\ell_{\max,k}} e(X_{i_{k\ell}}, Y_{j_{k\ell}}^{\bar{b}_{k\ell}\epsilon_{k\ell}r_k}) = \prod_{i=1}^{i_{\max}} e(X_i, \underline{\textstyle\prod_{(k,\ell)\in I(i)} Y_{j_{k\ell}}^{\bar{b}_{k\ell}\epsilon_{k\ell}r_k}})$$

として良い. 命題 P の関数 δ を

$$\delta(P) = \left\{ \begin{array}{ll} 1 & P \text{ が真の場合}, \\ 0 & \text{それ以外の場合}, \end{array} \right.$$

と定義すると, 上記下線部について有限個の冪乗の積の演算順序を交換する事により,

$$\prod_{(k,\ell)\in I(i)} Y_{j_{k\ell}}^{\bar{b}_{k\ell}\epsilon_{k\ell}r_k} = \prod_{j=1}^{j_{\max}} Y_j^{\underline{\sum_{(k,\ell)\in I(i)} \bar{b}_{k\ell}\epsilon_{k\ell}r_k\delta(j_{k\ell}=j)}}$$

として良い. 上記下線部について

$$\eta_{ij} = \sum_{(k,\ell)\in I(i)} \bar{b}_{k\ell}\epsilon_{k\ell}r_k\delta(j_{k\ell} = j)$$

と定義すれば,

$$\prod_{k=1}^{k_{\max}} E_k^{r_k} = \left(\prod_{i=1}^{i_{\max}} e(X_i, \prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}})\right) \times \underline{\left(\prod_{k=1}^{k_{\max}} \prod_{\ell=1}^{\ell_{\max,k}} e(X_{i_{k\ell}}^{b_{k\ell}\epsilon_{k\ell}r_k}, Y_{j_{k\ell}})\right)}$$

と出来る. 同様に $\mathbb{G}_1$ 変数 Y_j の添え字 $j \in \{1, \ldots, j_{\max}\}$ の関数 J を

$$J : j \mapsto \{(k, \ell) : j_{k\ell} = j\}$$

と定義し,

$$\xi_{ji} = \sum_{(k,\ell)\in J(j)} b_{k\ell}\epsilon_{k\ell}r_k\delta(i_{k\ell} = i)$$

と定義し, 上記下線部を変形すると

$$\prod_{k=1}^{k_{\max}} E_k^{r_k} = \left(\prod_{i=1}^{i_{\max}} e(X_i, \prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}})\right) \cdot \left(\prod_{j=1}^{j_{\max}} e(\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}}, Y_j)\right)$$

と出来る. 大雑把に見積もると, この一括検証の計算コストは最大 $i_{\max} + j_{\max}$ 個のペアリングの積で構成される多重ペアリングのコストと各ペアリングの引数を構成する多重冪乗のコストの総和により決定される. このとき, $\prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}} = 1$ あるいは $\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}} = 1$ なるペアリングは値が 1 となる事が予め分かっているので, 一般には実際に多重ペアリングを構成する有効なペアリングの個数 n は $i_{\max} + j_{\max}$ よりも小さい. L をペアリングを計算する際の Miller loop のコスト F をペアリングを計算する際の最終冪のコストとすると, 多重ペアリングに必要な演算コストは概ね

$$F + n \cdot L$$

と見積もることが出来る. また $\mathbb{G}_1$ 多重冪乗 $\prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}}$ に必要な演算コストは, $s_{1,i}$ を全指数の最大 bit 数, $m_{1,i}$ を全指数の合計 bit 数とし, S_1 を $\mathbb{G}_1$ 自乗のコスト M_1 を $\mathbb{G}_1$ 乗算のコストとすると, ある定数 $c_1 \in \left(0, \frac{1}{2}\right]$ が存在して,

$$s_{1,i} \cdot S_1 + c_1 \cdot m_{1,i} \cdot M_1$$

と見積もることが出来る. 同様に $\mathbb{G}_0$ 多重冪乗 $\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}}$ に必要な演算コストは $s_{0,j}$ を全指数の最大 bit 数, $m_{0,j}$ を全指数の合計 bit 数とし, S_0 を $\mathbb{G}_0$ 自乗のコスト M_0 を $\mathbb{G}_0$ 乗算のコストとすると, ある定数 $c_0 \in \left(0, \frac{1}{2}\right]$ が存在して

$$s_{0,j} \cdot S_0 + c_0 \cdot m_{0,j} \cdot M_0$$

と見積もることが出来る.

$$s_0 = \sum_j s_{0,j}, \quad s_1 = \sum_i s_{1,i},$$
$$m_0 = \sum_j m_{0,j}, \quad m_1 = \sum_i m_{1,i}.$$

と定義すれば, 一括検証の全コスト B を

$$B = F + n \cdot L + \sum_{d\in\{0,1\}} (s_d \cdot S_d + c_d \cdot m_d \cdot M_d) \quad (3)$$

とする事が出来る. 対称ペアリングの場合も似たような評価が出来る (後述). ところで $\{0,1\}$ 変数の任意の連言と否定は式 (1) 及び 式 (2) の変形により新しい $\{0,1\}$ 変数に置き換え可能で従ってどのような多項式長の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整数計画問題に効率的に変換出来る事が良く知られている. 従ってもし (3) 式の B を $\{0,1\}$ 変数 $b_{k\ell}$ の論理式の線形和で見積もる事が出来るなら, 上記の変換を用いて B の最適化問題を整数計画問題に変換できるはずである. 整数計画問題は必ずしも効率的に解けるとは限らないが, 現在まで膨大な量の研究が行われており, 例えばアカデミックフリーの SCIP [33] や商用の gurobi [34] のような高品質のソルバであれば現実的な問題を現実的な時間で解く事が期待できる. 従って, 以下では (3) 式の B の非定数因子, 即ち n, s_d, m_d をそれぞれ $b_{k\ell}$ の論理式の線形和で記述する事を考える.

4 非対称ペアリングの場合

r_k を確率変数として実際に多重ペアリングを構成する有効なペアリングの個数 n は

$$\begin{aligned}
n &= \left(\sum_{i=1}^{i_{\max}} \delta\left(\prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}} \neq 1\right)\right) + \left(\sum_{j=1}^{j_{\max}} \delta\left(\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}} \neq 1\right)\right) \\
&= \left(\sum_{i=1}^{i_{\max}} \bigvee_{j=1}^{j_{\max}} \delta\left(\eta_{ij} \neq 0\right)\right) + \left(\sum_{j=1}^{j_{\max}} \bigvee_{i=1}^{i_{\max}} \delta\left(\xi_{ji} \neq 0\right)\right) \\
&= \left(\sum_{i=1}^{i_{\max}} \bigvee_{j=1}^{j_{\max}} \delta\left(\sum_{(k,\ell)\in I(i)} \bar{b}_{k\ell}\epsilon_{k\ell}r_k\delta(j_{k\ell} = j) \neq 0\right)\right) + \\
&\quad \left(\sum_{j=1}^{j_{\max}} \bigvee_{i=1}^{i_{\max}} \delta\left(\sum_{(k,\ell)\in J(j)} b_{k\ell}\epsilon_{k\ell}r_k\delta(i_{k\ell} = i) \neq 0\right)\right) \\
&= \left(\sum_{i=1}^{i_{\max}} \bigvee_{(k,\ell)\in I(i)} \bar{b}_{k\ell}\right) + \left(\sum_{j=1}^{j_{\max}} \bigvee_{(k,\ell)\in J(j)} b_{k\ell}\right)
\end{aligned}$$

である. $\mathbb{G}_1$ 多重冪乗 $\prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}}$ の全指数の最大 bit 数 $s_{1,i}$ は,

$$\begin{aligned} s_{1,i} &= \max_j |\eta_{ij}| = \max_j \left| \sum_{(k,\ell)\in I(i)} \bar{b}_{k\ell}\epsilon_{k\ell}r_k\delta(j_{k\ell}=j) \right| \\ &\sim \max_{(k,\ell)\in I(i)} \bar{b}_{k\ell}|\epsilon_{k\ell}r_k| = \big(\bar{b}_{k_1\ell_1}|\epsilon_{k_1\ell_1}r_{k_1}| \\ &\quad + b_{k_1\ell_1}\left(\bar{b}_{k_2\ell_2}|\epsilon_{k_2\ell_2}r_{k_2}| + b_{k_2\ell_2}(\ldots)\right)\big)\Big|_{(k_n,\ell_n)\in I(i)} \end{aligned}$$

と見積もることが出来る. 但し $|\epsilon_{k_1\ell_1}r_{k_1}| \geq |\epsilon_{k_2\ell_2}r_{k_2}| \geq \ldots$ とする. $\epsilon_{k\ell}$ および r_k がどのような集合から選択されるかは予め分かっているため, $|\epsilon_{k\ell}r_k|$ の値は概算する事が出来, 従って整列する事が出来るとする. 例えば $\mathbb{L} = \mathbb{Z}/q\mathbb{Z}$ のとき, 一般に $|\epsilon_{k\ell}|$ は非常に小さい場合 (例えば $|\epsilon_{k\ell}| \sim 1$ の時) あるいは非常に大きい場合 (例えば $|\epsilon_{k\ell}| \sim |q|$ の時) を想定することが可能で, $|r_k|$ は $|r_k| \sim |q|/2$ あるいは, 特定の k については $|r_k| \sim 1$ 等と想定することが可能である. このとき, $|\epsilon_{k\ell}r_k|$ は表 1 のように見積もれる. 同様に $\mathbb{G}_0$ 多重冪乗 $\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}}$ の全

表 1　$|\epsilon_{k\ell}r_k|$ の見積もり

$\lvert r_k\rvert$ \ $\lvert\epsilon_{k\ell}\rvert$	~ 1	$\sim \lvert q\rvert$
~ 1	~ 1	$\sim \lvert q\rvert$
$\sim \lvert q\rvert/2$	$\sim \lvert q\rvert/2$	$\sim \lvert q\rvert$

指数の最大 bit 数 $s_{0,j}$ は,

$$\begin{aligned} s_{0,j} &= \max_i |\xi_{ji}| = \max_i \left| \sum_{(k,\ell)\in J(j)} b_{k\ell}\epsilon_{k\ell}r_k\delta(i_{k\ell}=i) \right| \\ &\sim \max_{(k,\ell)\in J(j)} b_{k\ell}|\epsilon_{k\ell}r_k| = \big(b_{k_1\ell_1}|\epsilon_{k_1\ell_1}r_{k_1}| \\ &\quad + \bar{b}_{k_1\ell_1}\left(b_{k_2\ell_2}|\epsilon_{k_2\ell_2}r_{k_2}| + \bar{b}_{k_2\ell_2}(\ldots)\right)\big)\Big|_{(k_n,\ell_n)\in J(j)} \end{aligned}$$

と見積もることが出来る. 但し $|\epsilon_{k_1\ell_1}r_{k_1}| \geq |\epsilon_{k_2\ell_2}r_{k_2}| \geq \ldots$ とする. $\mathbb{G}_1$ 多重冪乗 $\prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}}$ の全指数の合計 bit 数 $m_{1,i}$ は,

$$\begin{aligned} m_{1,i} &= \sum_j |\eta_{ij}| = \sum_j \left| \sum_{(k,\ell)\in I(i)} \bar{b}_{k\ell}\epsilon_{k\ell}r_k\delta(j_{k\ell}=j) \right| \\ &\sim \sum_j \max_{(k,\ell)\in I(i)} \bar{b}_{k\ell}|\epsilon_{k\ell}r_k\delta(j_{k\ell}=j)| \\ &= \sum_j \Bigg(\big(\bar{b}_{k_1\ell_1}|\epsilon_{k_1\ell_1}r_{k_1}| + b_{k_1\ell_1}\left(\bar{b}_{k_2\ell_2}|\epsilon_{k_2\ell_2}r_{k_2}|\right. \\ &\qquad \left. + b_{k_2\ell_2}(\ldots)\right)\big)\Big|_{(k_n,\ell_n)\in I(i),\ j_{k_n\ell_n}=j} \Bigg) \end{aligned}$$

と見積もることが出来る. 但し $|\epsilon_{k_1\ell_1}r_{k_1}| \geq |\epsilon_{k_2\ell_2}r_{k_2}| \geq \ldots$ とする. 同様に $\mathbb{G}_0$ 多重冪乗 $\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}}$ の全指数の合計 bit 数 $m_{0,j}$ は,

$$\begin{aligned} m_{0,j} &= \sum_i |\xi_{ji}| = \sum_i \left| \sum_{(k,\ell)\in J(j)} b_{k\ell}\epsilon_{k\ell}r_k\delta(i_{k\ell}=i) \right| \\ &\sim \sum_i \max_{(k,\ell)\in J(j)} b_{k\ell}|\epsilon_{k\ell}r_k\delta(i_{k\ell}=i)| \\ &= \sum_i \Bigg(\big(b_{k_1\ell_1}|\epsilon_{k_1\ell_1}r_{k_1}| + \bar{b}_{k_1\ell_1}\left(b_{k_2\ell_2}|\epsilon_{k_2\ell_2}r_{k_2}|\right. \\ &\qquad \left. + \bar{b}_{k_2\ell_2}(\ldots)\right)\big)\Big|_{(k_n,\ell_n)\in J(j),\ i_{k_n\ell_n}=i} \Bigg) \end{aligned}$$

と見積もることが出来る. 但し $|\epsilon_{k_1\ell_1}r_{k_1}| \geq |\epsilon_{k_2\ell_2}r_{k_2}| \geq \ldots$ とする. 従って,

$$\begin{aligned} s_0 &= \sum_j s_{0,j}, \qquad s_1 = \sum_i s_{1,i}, \\ m_0 &= \sum_j m_{0,j}, \quad m_1 = \sum_i m_{1,i}. \end{aligned}$$

および一括検証の全コスト

$$B = F + n\cdot L + \sum_{d\in\{0,1\}} (s_d\cdot S_d + c_d\cdot m_d\cdot M_d)$$

を考えると, B を $\{0,1\}$ 変数 $b_{k\ell}$ の論理式の線形和で記述できた.

5　対称ペアリングの場合

対称ペアリングにおいては $\mathbb{G}_0 = \mathbb{G}_1 = \mathbb{G}$ と考えて良い. 従って方式を構成するペアリングの入力群変数の集合 $X_1,\ldots,X_{i_{\max}} \in \mathbb{G}_0$, および $Y_1,\ldots,Y_{j_{\max}} \in \mathbb{G}_1$ を同一視し $i_{\max} = j_{\max}$ とし全ての $i \in \{1,\ldots,i_{\max}\}$ について $X_i = Y_i$ と定義し直す. また $I(i)$ と $J(i)$ の disjoint union $U(i)$ を

$$\begin{aligned} &U(i) = I(i) \sqcup J(i) = \\ &\quad \{(k,\ell,t) | (k,\ell)\in I(i) \wedge t = 1 \vee (k,\ell)\in J(i) \wedge t = 0\} \end{aligned}$$

と定義し,

$$b_{k\ell t} = \begin{cases} b_{k\ell} & (t=0), \\ \bar{b}_{k\ell} & (t=1). \end{cases}$$

と定義し $\bar{b}_{k\ell t} = \neg b_{k\ell t}$ と定義する.

$$\begin{aligned} &\prod_{k=1}^{k_{\max}} E_k^{r_k} \\ &= \left(\prod_{i=1}^{i_{\max}} e(X_i, \prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}})\right) \cdot \left(\prod_{j=1}^{j_{\max}} e(\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}}, Y_j)\right) \\ &= \left(\prod_{i=1}^{i_{\max}} e(X_i, \prod_{j=1}^{i_{\max}} X_j^{\eta_{ij}})\right) \cdot \left(\prod_{i=1}^{i_{\max}} e(\prod_{j=1}^{i_{\max}} X_j^{\xi_{ij}}, X_i)\right) \end{aligned}$$

$$= \prod_{i=1}^{i_{\max}} e(X_i, \prod_{j=1}^{i_{\max}} X_j^{\zeta_{ij}})$$

である．但し，

$$\zeta_{ij} = \eta_{ij} + \xi_{ij} = \left(\sum_{(k,\ell)\in I(i)} \bar{b}_{k\ell}\epsilon_{k\ell}r_k\delta(j_{k\ell}=j) \right) + \left(\sum_{(k,\ell)\in J(i)} b_{k\ell}\epsilon_{k\ell}r_k\delta(i_{k\ell}=j) \right)$$

とする．このとき有効なペアリングの数 n は

$$n = \sum_{i=1}^{i_{\max}} \delta\left(\prod_{j=1}^{i_{\max}} X_j^{\zeta_{ij}} \neq 1 \right) = \sum_{i=1}^{i_{\max}} \bigvee_{j=1}^{i_{\max}} \delta(\zeta_{ij}\neq 0)$$
$$= \sum_{i=1}^{i_{\max}} \bigvee_{(k,\ell,t)\in U(i)} b_{k\ell t}$$

$\mathbb{G}$ 多重冪乗 $\prod_{j=1}^{i_{\max}} X_j^{\zeta_{ij}}$ の全指数の最大 bit 数 s_i は，

$$s_i = \max_j |\zeta_{ij}| \sim \max_{(k,\ell,t)\in U(i)} b_{k\ell t}|\epsilon_{k\ell}r_k|$$
$$= \big(b_{k_1\ell_1 t_1}|\epsilon_{k_1\ell_1}r_{k_1}| + \bar{b}_{k_1\ell_1 t_1}\left(b_{k_2\ell_2 t_2}|\epsilon_{k_2\ell_2}r_{k_2}| + \bar{b}_{k_2\ell_2 t_2}(\ldots)\right)\big)\big|_{(k_n,\ell_n,t_n)\in U(j)}$$

と見積もる事ができる．但し $|\epsilon_{k_1\ell_1}r_{k_1}| \geq |\epsilon_{k_2\ell_2}r_{k_2}| \geq \ldots$ とする．また，$\mathbb{G}$ 多重冪乗 $\prod_{j=1}^{i_{\max}} X_j^{\zeta_{ij}}$ の全指数の合計 bit 数 m_i は，

$$m_i = \sum_j |\zeta_{ij}|$$
$$\sim \sum_j \max_{(k,\ell,t)\in U(i)} b_{k\ell t}|\epsilon_{k\ell}r_k\delta\big((j_{k\ell}=j)\wedge(t=1)\vee(i_{k\ell}=j)\wedge(t=0)\big)|$$
$$= \sum_j \left(\big(b_{k_1\ell_1 t_1}|\epsilon_{k_1\ell_1}r_{k_1}| + \bar{b}_{k_1\ell_1 t_1}\left(b_{k_2\ell_2 t_2}|\epsilon_{k_2\ell_2}r_{k_2}| + \bar{b}_{k_2\ell_2 t_2}(\ldots)\right)\big)\Big|_{\substack{(k_n,\ell_n,t_n)\in U(j),\\ (j_{k_n\ell_n}=j)\wedge(t_n=1)\vee\\ (i_{k_n\ell_n}=j)\wedge(t_n=0)}} \right)$$

と見積もることが出来る．但し $|\epsilon_{k_1\ell_1}r_{k_1}| \geq |\epsilon_{k_2\ell_2}r_{k_2}| \geq \ldots$ とする．S を $\mathbb{G}$ 自乗のコスト M を $\mathbb{G}$ 乗算のコストとすると，ある定数 $c \in \left(0, \frac{1}{2}\right]$ が存在して $\mathbb{G}$ 多重冪乗 $\prod_{j=1}^{i_{\max}} X_j^{\zeta_{ij}}$ の演算コストを

$$s_i \cdot S + c \cdot m_i \cdot M$$

と見積もることが出来る．従って，

$$s = \sum_i s_i, \quad m = \sum_i m_i$$

とすると一括検証の全コスト

$$B = F + n \cdot L + s \cdot S + c \cdot m \cdot M$$

を考えると，B を $\{0,1\}$ 変数 $b_{k\ell}$ の論理式の線形和で記述できた．

6 まとめと今後の課題

本研究では整数計画法を用いた一括検証の自動生成アルゴリズムを提案した．提案した内容を簡単にまとめると以下のようになる．

Step 0: $\{E_k\}$ が入力される．

Step 1: 目的関数 B を $\{0,1\}$ 変数 $b_{k\ell}$ の論理式の線形和で書き下す．

Step 2: B の最適化問題を整数計画問題に変換する．

Step 3: 整数計画問題を任意の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．

Step 4: $b_{k\ell}$ の値に従い下記一括検証式を構成する．

(非対称ペアリングの場合)

$$\prod_{k=1}^{k_{\max}} E_k^{r_k} = \left(\prod_{i=1}^{i_{\max}} e(X_i, \prod_{j=1}^{j_{\max}} Y_j^{\eta_{ij}}) \right) \cdot \left(\prod_{j=1}^{j_{\max}} e(\prod_{i=1}^{i_{\max}} X_i^{\xi_{ji}}, Y_j) \right)$$

(対称ペアリングの場合)

$$\prod_{k=1}^{k_{\max}} E_k^{r_k} = \prod_{i=1}^{i_{\max}} e(X_i, \prod_{j=1}^{i_{\max}} X_j^{\zeta_{ij}})$$

一括検証自動生成系の実装評価は今後の課題である．

参考文献

[1] J. Groth and A. Sahai, “Efficient non-interactive proof systems for bilinear groups,” Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings, ed. N.P. Smart, Lecture Notes in Computer Science, vol.4965, pp.415–432, Springer, 2008. doi:10.1007/978-3-540-78967-3_24.

[2] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo, “Structure-preserving signatures and commitments to group elements,” Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings, ed. T. Rabin, Lecture Notes in Computer Science, vol.6223, pp.209–236, Springer, 2010. doi:10.1007/978-3-642-14623-7_12.

[3] A. Fiat, “Batch RSA,” Advances in Cryptology - CRYPTO ’89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings, ed. G. Brassard, Lecture Notes in Computer Science, vol.435, pp.175–185, Springer, 1989. doi:10.1007/0-387-34805-0_17.

[4] M. Bellare, J.A. Garay, and T. Rabin, “Fast batch verification for modular exponentiation and digital signatures,” Advances in Cryptology - EUROCRYPT ’98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding, ed. K. Nyberg, Lecture Notes in Computer Science, vol.1403, pp.236–250, Springer, 1998. doi:10.1007/BFb0054130.

[5] A.L. Ferrara, M. Green, S. Hohenberger, and M.O. Pedersen, "Practical Short Signature Batch Verification," Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings, ed. M. Fischlin, Lecture Notes in Computer Science, vol.5473, pp.309–324, Springer, 2009. doi:10.1007/978-3-642-00862-7_21.

[6] O. Blazy, G. Fuchsbauer, M. Izabachène, A. Jambert, H. Sibert, and D. Vergnaud, "Batch Groth-Sahai," Applied Cryptography and Network Security, 8th International Conference, ACNS 2010, Beijing, China, June 22-25, 2010. Proceedings, ed. J. Zhou and M. Yung, Lecture Notes in Computer Science, vol.6123, pp.218–235, 2010. doi:10.1007/978-3-642-13708-2_14.

[7] G. Herold, M. Hoffmann, M. Klooß, C. Ràfols, and A. Rupp, "New Techniques for Structural Batch Verification in Bilinear Groups with Applications to Groth-Sahai Proofs," Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017, ed. B.M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, pp.1547–1564, ACM, 2017. doi:10.1145/3133956.3134068.

[8] J.A. Akinyele, M. Green, S. Hohenberger, and M.W. Pagano, "Machine-Generated Algorithms, Proofs and Software for the Batch Verification of Digital Signature Schemes," the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012, ed. T. Yu, G. Danezis, and V.D. Gligor, pp.474–487, ACM, 2012. doi:10.1145/2382196.2382248.

[9] J.A. Akinyele, G. Barthe, B. Grégoire, B. Schmidt, and P. Strub, "Certified Synthesis of Efficient Batch Verifiers," IEEE 27th Computer Security Foundations Symposium, CSF 2014, Vienna, Austria, 19-22 July, 2014, pp.153–165, IEEE Computer Society, 2014. doi:10.1109/CSF.2014.19.

[10] S.D. Galbraith, K.G. Paterson, and N.P. Smart, "Pairings for cryptographers," Discrete Applied Mathematics, vol.156, no.16, pp.3113–3121, 2008. doi:10.1016/j.dam.2007.12.010.

[11] G.B. Dantzig, "On the Significance of Solving Linear Programming Problems with Some Integer Variables," Econometrica, vol.28, no.1, pp.30–44, 1960. URL: http://www.jstor.org/stable/1905292.

[12] R.E. Bellman, "An introduction to the theory of dynamic programming." The RAND Corporation, Report R-245, 1953. URL: https://apps.dtic.mil/dtic/tr/fulltext/u2/074903.pdf.

[13] A.H. Land and A.G. Doig, "An Automatic Method of Solving Discrete Programming Problems," Econometrica, vol.28, no.3, pp.497–520, 1960. URL: http://www.jstor.org/stable/1910129.

[14] M. Held and R.M. Karp, "The Traveling-Salesman Problem and Minimum Spanning Trees," Operations Research, vol.18, no.6, pp.1138–1162, 1970. URL: http://www.jstor.org/stable/169411.

[15] M. Held and R.M. Karp, "The Traveling-Salesman Problem and Minimum Spanning Trees: Part II," Mathematical Programming, vol.1, no.1, pp.6–25, Dec 1971. doi:10.1007/BF01584070.

[16] R.E. Gomory, "Outline of an algorithm for integer solutions to linear programs," Bull. Amer. Math. Soc., vol.64, no.5, pp.275–278, 09 1958. URL: https://projecteuclid.org:443/euclid.bams/1183522679.

[17] M. Padberg and G. Rinaldi, "Optimization of a 532-city Symmetric Traveling Salesman Problem by Branch and Cut," Oper. Res. Lett., vol.6, no.1, pp.1–7, March 1987. doi:10.1016/0167-6377(87)90002-2.

[18] M. Padberg and G. Rinaldi, "A Branch-and-Cut Approach to a Traveling Salesman Problem with Side Constraints," Management Science, vol.35, no.11, pp.1393–1412, 1989. URL: http://www.jstor.org/stable/2632284.

[19] M. Padberg and G. Rinaldi, "A Branch-and-cut Algorithm for the Resolution of Large-scale Symmetric Traveling Salesman Problems," SIAM Rev., vol.33, no.1, pp.60–100, Feb. 1991. doi:10.1137/1033004.

[20] P. Raghavan and C.D. Thompson, "Randomized Rounding: A Technique for Provably Good Algorithms and Algorithmic Proofs," Combinatorica, vol.7, no.4, pp.365–374, 1987. doi:10.1007/BF02579324.

[21] K.L. Clarkson, "Las Vegas Algorithms for Linear and Integer Programming when the Dimension is Small," J. ACM, vol.42, no.2, pp.488–499, March 1995. doi:10.1145/201019.201036.

[22] J. Edmonds, "Matroids and the greedy algorithm," Mathematical Programming, vol.1, no.1, pp.127–136, Dec 1971. doi:10.1007/BF01584082.

[23] G.A. Croes, "A Method for Solving Traveling-Salesman Problems," Operations Research, vol.6, no.6, pp.791–812, 1958. URL: http://www.jstor.org/stable/167074.

[24] S. Lin, "Computer Solutions of the Traveling Salesman Problem," The Bell System Technical Journal, vol.44, no.10, pp.2245–2269, Dec 1965. doi:10.1002/j.1538-7305.1965.tb04146.x.

[25] S. Lin and B. W. Kernighan, "An Effective Heuristic Algorithm for the TSP," Operations Research, vol.21, pp.498–, 04 1973. doi:10.1287/opre.21.2.498.

[26] F. Glover, "Tabu Search–Part I," ORSA Journal on Computing, vol.1, no.3, pp.190–206, 1989. arXiv:https://doi.org/10.1287/ijoc.1.3.190, doi:10.1287/ijoc.1.3.190.

[27] F. Glover, "Tabu Search–Part II," ORSA Journal on Computing, vol.2, no.1, pp.4–32, 1990. arXiv:https://doi.org/10.1287/ijoc.2.1.4, doi:10.1287/ijoc.2.1.4.

[28] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," The Journal of Chemical Physics, vol.21, no.6, pp.1087–1092, 1953. arXiv:https://doi.org/10.1063/1.1699114, doi:10.1063/1.1699114.

[29] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by Simulated Annealing," Science, vol.220, no.4598, pp.671–680, 1983. arXiv:http://science.sciencemag.org/content/220/4598/671.full.pdf, doi:10.1126/science.220.4598.671.

[30] V. Černý, "Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm," Journal of Optimization Theory and Applications, vol.45, no.1, pp.41–51, Jan 1985. doi:10.1007/BF00940812.

[31] J.H. Holland, "Genetic Algorithms and the Optimal Allocation of Trials," SIAM J. Comput., vol.2, no.2, pp.88–105, 1973. doi:10.1137/0202009.

[32] J.H. Holland, "Erratum: Genetic Algorithms and the Optimal Allocation of Trials," SIAM J. Comput., vol.3, no.4, p.326, 1974. doi:10.1137/0203026.

[33] T. Achterberg, "CIP: Solving constraint integer programs," Mathematical Programming Computation, vol.1, no.1, pp.1–41, 2009. URL: http://mpc.zib.de/index.php/MPC/article/view/4.

[34] Gurobi Optimization, Inc., "Gurobi optimizer reference manual." In http://www.gurobi.com/. URL: http://www.gurobi.com/documentation/8.1/refman.pdf.