Private Permutation を用いない 金持ち比べカードベースプロトコルの効率化

宮原 大輝[1]　林 優一[2]　水木 敬明[3]　曽根 秀昭[3]

概要：カードベース暗号を用いた秘密計算の重要なアプリケーションとして，2016 年に Nakai らは，金持ち比べ問題を対象にしたカードベースプロトコルを提案した．すなわち，彼らは既存の AND プロトコル等を組み合わせて，2 入力の大小関係を秘密計算するプロトコルを与えたとともに，背面での秘匿操作を想定した Praivate Permutation (PP) を許容することにより効率的なプロトコルも考案している．本稿では，PP を用いないという場面設定に焦点を絞り，既存プロトコルよりも効率的なプロトコルを提案する．

キーワード：カードベース暗号，秘密計算，金持ち比べ問題

1. はじめに

m を自然数とする．Alice と Bob がそれぞれ $a, b \in \{1, 2, \ldots, m\}$ という額の財産を所持し，それらの値を互いに秘匿したまま，その大小関係のみを知りたいという状況を考えよう．これは**金持ち比べ問題**と呼ばれ，1982 年に Yao によって提唱されたと同時に，公開鍵暗号方式に基づく解決方法が提案された [1]．この問題をコンピュータを用いずに解決する方法として，物理的なカード組を利用したプロトコルが，2016 年に Nakai らによって提案された [2]．カード組をどのように利用するのかについての説明から始める．

1.1 カードベースプロトコル

用いられるカード組は，次のような性質を満たすものである．

(1) 表面が同種類のカード（黒 $\boxed{\clubsuit}$ や赤 $\boxed{\heartsuit}$ 等）同士は互いに区別できない．
(2) 全てのカードの裏面は同一の模様 $\boxed{?}$ であり，互いに区別できない．

このようなカードを用いると，ビットを

$$\boxed{\clubsuit}\boxed{\heartsuit} = 0,\ \boxed{\heartsuit}\boxed{\clubsuit} = 1 \qquad (1)$$

のように表現することができる．この符号化に基づき，Alice と Bob は，$n = \lceil \log m \rceil$ とする（log の底は 2 とする）とき，各々の財産の 2 進表現 $a = (a_n, \cdots, a_1)_2$ と $b = (b_n, \cdots, b_1)_2$ を，それぞれ相手に値を秘密にしたまま，$2n$ 枚のカードを用いてテーブルの上に置くことができる．

$$\underbrace{\boxed{?}\boxed{?}}_{a_n} \cdots \underbrace{\boxed{?}\boxed{?}}_{a_1} \quad \underbrace{\boxed{?}\boxed{?}}_{b_n} \cdots \underbrace{\boxed{?}\boxed{?}}_{b_1} \qquad (2)$$

ここで，裏に置かれたカード 2 枚

$$\underbrace{\boxed{?}\boxed{?}}_{x}$$

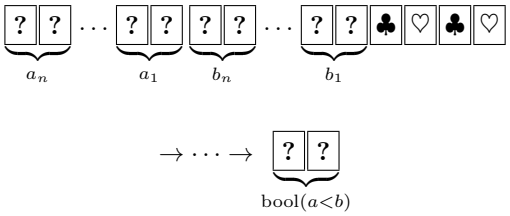
はビット x の**コミットメント**と呼ばれ，$x = 0$ のとき $\boxed{\clubsuit}\boxed{\heartsuit}$，$x = 1$ のとき $\boxed{\heartsuit}\boxed{\clubsuit}$ という並びになっていることを意味する．上記 (2) のようなカード列が与えられたとき，シャッフル操作や並び替え操作等を行って，$a < b$ か否かだけを求めたい．

1.2 金持ち比べ問題への適用

上述のようなカード組を使う**カードベースプロトコル**を用いると，どんな関数も秘密計算できることが知られている [3]．従って，上記のカード列 (2) に十分な追加カードを加えることで，理論上は $a < b$ の値だけを知ることができる．実際，Nakai ら [2] は，既存の AND プロトコル [4] と COPY プロトコル [4]（これら二つの既存プロトコルは 2.1 と 2.2 節で詳しく説明する）を組み合わせて，金持ち比べ問題を具体的に解く手順を与えている．すなわち，入力としてカード列 (2) と 4 枚の追加カードが与えられたとき，AND プロトコルと COPY プロトコルを複数回使用して，$a < b$ の真理値を表すコミットメントを出力できる

[1] 東北大学大学院情報科学研究科
[2] 奈良先端科学技術大学院大学情報科学研究科
[3] 東北大学サイバーサイエンスセンター

ことを示している．



ここで，bool($a < b$) は，

$$\mathrm{bool}(a<b) \stackrel{\mathrm{def}}{=} \begin{cases} 0 & \text{if } a \geq b \\ 1 & \text{if } a < b \end{cases}$$

を意味する．入力列（2）のために $4\lceil \log m \rceil$ 枚のカードを使うので，表 1 に示している通り，合計 $4\lceil \log m \rceil + 4$ 枚のカードを用いる．また，表 1 の通りシャッフルの回数は $6\lceil \log m \rceil - 5$ である．これらの枚数や回数については，具体的な手順と共に 2.3 節で詳しく説明する．

文献 [2] での主要な結果は（上述の手順ではなく），プレーヤーの背面での秘匿操作を想定した Praivate Permutation (PP) という操作を許容することで，必要な PP 回数の少ない効率的なプロトコルを実現しているところである．その内の一つは，Yao の解法 [1] のアイデアを利用して，所持金額の情報を $2m$ 枚のカード列に埋め込んで bool($a < b$) の値を得るプロトコルとなっている（「Yao の解法に基づくプロトコル」と呼ばれる）．もう一つのプロトコルでは，Alice と Bob は所持金額の二進数のカード列をそれぞれ持ち，PP によってビット毎に比較して bool($a < b$) の値を得る（「storage プロトコルと呼ばれる」）．

1.3 本稿の貢献

本稿では，公衆の状況下での操作を前提とすることとし，PP を用いないという場面設定に焦点を絞り，2 つの効率的な金持ち比べプロトコルを提案する．1 つ目の「提案プロトコル 1 」は，表 1 に示す通り，既存のプロトコルよりもシャッフル回数を 1/3 程度に削減できており，storage プロトコル [2] と AND プロトコル [4] のアイデアを応用して構成する．2 つ目の「提案プロトコル 2 」は，$3m - 1$ 枚のカードが必要ながらも，シャッフルは 1 回で済むものである．これは，Yao の解法に基づくプロトコル [1] と，最近発表された Koch と Walzer の Chosen Cut [5] というアイデアを融合させることで得られる．

本稿の構成は次の通りである．2 節では，既存の AND プロトコルと COPY プロトコル [4] の手順を説明した後に，文献 [2] による金持ち比べの手順を紹介する．3 節では，提案プロトコル 1 のアイデアと具体的な手順を示す．4 節では，提案プロトコル 2 の具体的な手順を示す．5 節で，本稿をまとめる．

表 1: PP を用いない金持ち比べプロトコル

	枚数	シャッフル回数
文献 [2] の手順	$4\lceil \log m \rceil + 4$	$6\lceil \log m \rceil - 5$
提案プロトコル 1（ 3 節）	$4\lceil \log m \rceil + 2$	$2\lceil \log m \rceil - 1$
提案プロトコル 2（ 4 節）	$3m - 1$	1

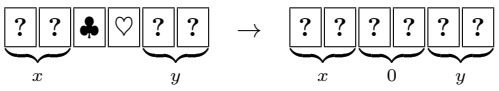
2. 準備

本節では，準備として既存の AND プロトコルと COPY プロトコル [4] を説明をした後に，それらを組み合わせて金持ち比べ問題を解く文献 [2] の手順を紹介する．

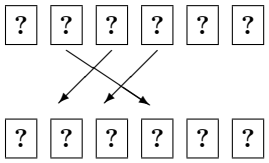
2.1 AND プロトコル

2009 年に Mizuki と Sone により開発された次の 6 枚 AND プロトコル [4] は，ビット x と y のコミットメント（と 2 枚の追加カード）から，シャッフルやカードをめくる操作等を用いて，$x \wedge y$ のコミットメントを出力する．

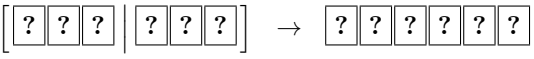
(1) 入力コミットメント，黒と赤の追加カードを次のように並べ，中央の 2 枚を裏返す．



(2) 次のように並び替える．

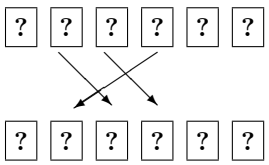


(3) カード列を半分に割り，左右をランダムに入れ替えるシャッフル操作，すなわち**ランダム二等分割カット**を適用する．

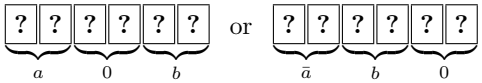


このシャッフル操作後，6 枚のカード列は，そのままの列となる確率が 1/2 ，左右が入れ替わる列となる確率が 1/2 である．

(4) 次のように並び替える．

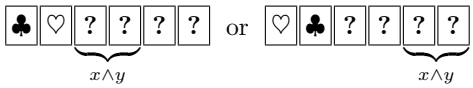


この操作後，6 枚のカード列は次の状態となっている．



(5) 左から 1 枚目と 2 枚目のカードをめくる．見えるカー

ドに応じて，次のように $x \wedge y$ のコミットメントが得られる．なお，めくったカード 2 枚は再利用できる．また，$x \wedge y$ のコミットメントでない方のカード 2 枚もシャッフル後にめくることで再利用できる．



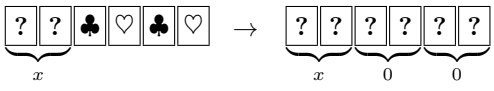
以上のように，$x \wedge y$ のコミットメントを（秘匿した状態で）得ることができる．ステップ (3) で用いられるランダム二等分割カットは，文献 [6] で人間の手によって問題なく安全に実行できることが確かめられている．すなわち，ランダム二等分割カットを実行する当事者でさえも，実行した結果どのような状態のカード列となるのか知り得ない．

なお，OR プロトコルも同様に得られる．

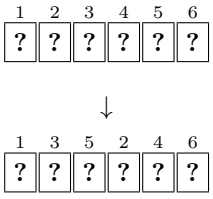
2.2 COPY プロトコル

同じく Mizuki と Sone が提案した次の 6 枚 COPY プロトコルは，ビット x のコミットメントと 4 枚の追加カードから，x のコミットメントを 2 つ出力する．

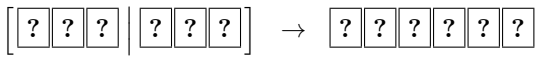
(1) 入力コミットメント，黒と赤の追加カードを次のように並べ，追加カードを裏返す．



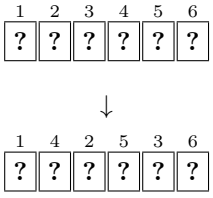
(2) 次のように並び替える．



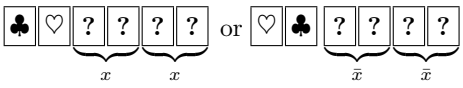
(3) ランダム二等分割カットを行う．



(4) 次のように並び替える．

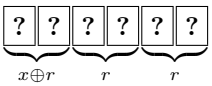


(5) 1 枚目と 2 枚目のカードをめくり，2 個の x のコミットメントを得る（めくったカード 2 枚は次の計算に再利用できる）．



後者の場合は NOT 計算（コミットメントを構成する 2 枚のカードの位置を交換すること）によって，$\bar{x}$ のコミットメントから x のコミットメントを得ることができる．

ステップ (4) の操作後，r を一様ランダムなビットとするとき，カード列は次のような状態となっている．



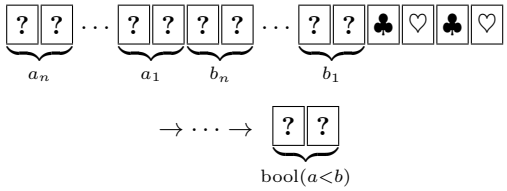
すなわち，ステップ (1) のカード列のそれぞれのコミットメントに，ランダムビット r を付与した状態である．

2.3 既存の金持ち比べプロトコル

2.1 と 2.2 節のプロトコルを組み合わせて金持ち比べ問題を解決するプロトコル [2] は，次のような論理回路に基づいたものとなっている．

文献 [2] の手順
input : $a = (a_n, \ldots, a_1)_2, b = (b_n, \ldots, b_1)_2$;
$f_1 = \bar{a}_1 \wedge b_1$;
for(i : 2 to n) {
  $f_i = (\bar{a}_i \wedge b_i) \vee ((\bar{a}_i \vee b_i) \wedge f_{i-1})$;
}
output : f_n ($= \mathrm{bool}(a < b)$) ;

すなわち，入力カード列 (2) が与えられ，AND (OR) プロトコル [4] を $4\lceil \log m \rceil - 3$ 回，$\bar{a}_i$ と b_i のコミットメントをそれぞれ二つに複製するための COPY プロトコル [4] を $2\lceil \log m \rceil - 2$ 回使用して，大小関係を表す $\mathrm{bool}(a < b)$ のコミットメントを出力する．



従って，上述のプロトコルに必要なシャッフルの回数は，合計で $6\lceil \log m \rceil - 5$ 回となる（AND と COPY ともに 1 回のランダム二等分割カットを使うことを思い出そう）．また，必要なカード枚数は，$f_2 = (\bar{a}_2 \wedge b_2) \vee ((\bar{a}_2 \vee b_2) \wedge f_1)$ の計算において，a_2 と b_2 のコミットメントをそれぞれ 2 個に複製するために，追加カードが 6 枚必要である．従って，$f_1 = \bar{a}_1 \wedge b_1$ の AND 計算の直後に 6 枚のカードが余っている必要があり，そのためには f_1 の計算の前に 4 枚の追加カードがあればよい．以上により，必要なカード枚数は合計 $4\lceil \log m \rceil + 4$ 枚となる．

3. 金持ち比べプロトコルの効率化

本節では，2.3 節で紹介した既存の金持ち比べプロトコル [2] よりも，必要なシャッフル回数とカード枚数が少ないという意味で効率的な金持ち比べプロトコルを提案する

(「提案プロトコル 1 」と呼ぶ).

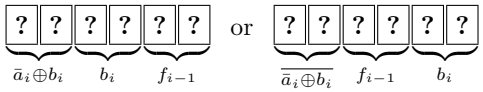
まず 3.1 節で提案プロトコル 1 の主なアイデアを述べ，3.2 節でプロトコルの手順を示す．

3.1 提案プロトコル 1 のアイデア

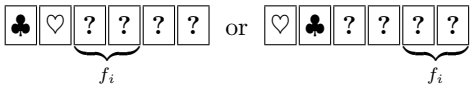
PP を用いている storage プロトコル [2] は，2.3 節で示されている f_i を，次式のように捉えている．

$$f_i = \begin{cases} b_i & \text{if } \bar{a}_i = b_i \\ f_{i-1} & \text{if } \bar{a}_i \neq b_i \end{cases} \tag{3}$$

すなわち，$\bar{a}_i$ と b_i が等しいか否かに応じて，b_i か f_{i-1} を PP により選択するイメージである．これは PP を用いない設定でも，6 枚のカード列を次のような状態

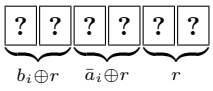


にすることができれば，1,2 枚目をめくり出てきた色に応じて，f_i のコミットメントを次のように得ることができる．

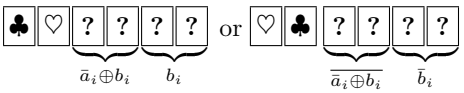


以上の流れは，2.1 節で紹介した 6 枚 AND プロトコル [4] の手順を用いることで実現できる．

更に，$\bar{a}_i \oplus b_i$ と b_i のコミットメントは，2.2 節で紹介した 6 枚 COPY プロトコル [4] を用いることで，a_i と b_i のコミットメントから簡単に得ることができる．すなわち，r をランダムビットとするときに，次のようなカード列の状態



から，1,2 枚目をめくり出てきた色に応じて，$r = b_i$ か $r = \bar{b}_i$ かが決定し，$\bar{a}_i \oplus b_i$ と b_i のコミットメントを次のように得ることができる．



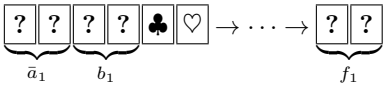
以上のように，COPY プロトコルと AND プロトコルの手順を活用することで，a_i と b_i と f_{i-1} の値を漏らすことなく，式（3）の通りに f_i のコミットメントを得ることができる．

実は，2.3 節で紹介した f_i の論理式を詳しく見ると，f_i は a_i と b_i と f_{i-1} を入力とする多数決関数になっていることが分かる．3 変数多数決関数を実現する効率的なカードベースプロトコルは，2013 年に Nishida らによって提案されていて，上述のアイデアと全く同様のものとなっている [7].

3.2 提案プロトコル 1 の構成

3.1 節で説明したアイデアに基づき，提案プロトコル 1 を次のように構成する．入力カード列（2）と追加カード 2 枚が与えられたとき，次のように動作する．

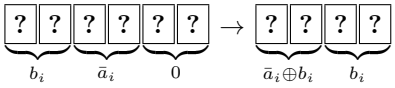
(1) 2.1 節で紹介した 6 枚の AND プロトコル [4] を利用して，$f_1 = \bar{a}_1 \wedge b_1$ を計算する．



このとき，再利用可能なカードが 4 枚残る．

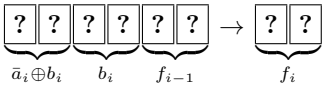
(2) 次の計算を $i = 2$ から $i = n$ まで繰り返す．

(a) 6 枚の COPY プロトコル [4]（と NOT 計算）を利用して，a_i と b_i のコミットメントから，$\bar{a}_i \oplus b_i$ と b_i のコミットメントを得る．



このとき，再利用可能なカードが 2 枚残る．

(b) 6 枚の AND プロトコル [4] を利用して，$\bar{a}_i \oplus b_i$ と b_i と f_{i-1} のコミットメントから，f_i のコミットメントを得る．



(3) $f_n = \mathrm{bool}(a < b)$ のコミットメントが得られている．

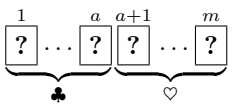
この提案プロトコル 1 は，既存のコミット型プロトコルの組み合わせであるので，安全性は保障されている．2 枚の追加カードを用いているので，入力カード列 (2) と合わせて合計 $4\lceil \log m \rceil + 2$ 枚のカードを使用する．また，f_1 のための AND 計算の後，AND プロトコルと COPY プロトコルのそれぞれの手順を $i = 2$ から $i = n$ まで繰り返すので，必要なシャッフル操作回数は合計で $2\lceil \log m \rceil - 1$ である．

4. Yao の解法に基づくプロトコル

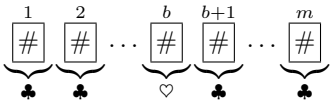
本節では，金持ち比べ問題の Yao の解法 [1] のアイデアを，PP を用いずにカードベースプロトコルへ応用して得られる「提案プロトコル 2 」を構成する．1.2 節で述べた通り，文献 [2] では，PP を用いたプロトコルが提案されている．

提案プロトコル 2 の手順は次の通りである．

(1) Alice は $m-1$ 枚の $\boxed{\clubsuit}$ と m 枚の $\boxed{\heartsuit}$ を手に持ち，1 番目から a 番目までに $\boxed{\clubsuit}$，$a+1$ 番目から m 番目までに $\boxed{\heartsuit}$ を裏にして置き，Alice の入力とする．



Bob は裏面が $\boxed{\#}$ な ♣ のカードを $m-1$ 枚，裏面が $\boxed{\#}$ な ♡ のカードを 1 枚手に持ち，b 番目を ♡ にして置き，Bob の入力とする．



(2) Alice と Bob の入力から 1 枚ずつ交互に並べる．

(3) カード列を巡回的にシャッフルする操作，すなわち**ランダムカット**を適用する（$<\cdot>$ で表す）．

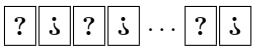
(4) 裏が $\boxed{\#}$ のカード（Bob が置いた m 枚のカード）をめくり，♡ の左にあるカードをめくる．

- $\boxed{\heartsuit}$ の場合，$a < b$ である．
- $\boxed{\clubsuit}$ の場合，$a \geq b$ である．この場合，Bob の ♡ のカードの右にあるカードを更にめくることにより等号成立か否かが分かる．すなわち，そのカードが $\boxed{\heartsuit}$ の場合，$a = b$ である．$\boxed{\clubsuit}$ の場合，$a > b$ である．

ステップ（2）で Bob は b 番目のみを異なる絵柄にして置くので，Alice が置いたカード列の b 番目をステップ（4）でめくることになる．ステップ（1）で Alice は a 番目までに $\boxed{\clubsuit}$ を置き，それ以降に $\boxed{\heartsuit}$ を置いたので，b 番目の絵柄に応じて $a < b$ か $a \geq b$ かが分かる．$a \geq b$ の場合は，Alice の $b+1$ 番目のカードをめくることによって，$a = b$ か否かが分かる．

ステップ（1）で Alice は a 枚の $\boxed{\clubsuit}$ を置くが，最初に Alice が a 枚の $\boxed{\clubsuit}$ を手に持っていたとしたら，そのカード枚数から a の値がばれることになる．従って，ステップ（1）で Alice は $m-1$ 枚の $\boxed{\clubsuit}$ を手にもつ必要がある（$\boxed{\heartsuit}$ も同様）．また，ステップ（3）でランダムカットを行うので，ステップ（4）で Bob が置いたカード列をめくっても，どの位置に Bob が異なる絵柄のカードを置いたのかは秘匿される（Alice やプロトコルを見ている人にばれない）．更に，Alice が置いたカード列の b 番目（と $b+1$ 番目）をステップ（4）でめくるだけなので，a と b の値は秘匿される．$a = b$ の場合，Alice と Bob はお互いの値を知ることになるが，周りでプロトコルを見ている人にその値は漏れないことに注意しよう．

ステップ（2）のように，特定の位置にだけ異なる絵柄を置いてランダムカットを行うというアイデアは，文献 [5] の “Chosen Cut”[i] から着想を得た．また，裏面の絵柄が上下非対称であれば，それを利用することで同じ “デッキ” からプロトコルを実行することもできる．すなわち，次のように入力する．



また，提案プロトコル 2 は市販されているトランプカードを用いても実装することができる．すなわち，Alice と Bob は，$\boxed{\clubsuit}$ の代わりにトランプのスペード ♠ とクラブ ♣ を使用し，$\boxed{\heartsuit}$ の代わりにトランプのダイヤ ♢ とハート ♡ を使用することにし，それぞれが入力する前に各自手持ちカードをシャッフルしておく．それによりトランプの数字によって a と b の値が漏れることはない．

5. 終わりに

本稿では，PP を用いずに金持ち比べ問題を解決するカードベースプロトコルを 2 つ提案した．特に，4 節で提案したプロトコルは，シャッフルはランダムカットを 1 回行うだけであり，トランプカードを用いても実装可能であり，正当性や安全性が自明であるので，高校生や非専門家でも容易に理解し実行することができる．従って，オープンキャンパスでの実演や出前授業等に加え，日常生活における利用が期待できる．なお，ランダムカットは，“Hindu cut” により簡単に安全に実行できる [6]．

今後の課題として，3 節で与えた提案プロトコル 1 を踏まえ，大人数での効率的な金持ち比べプロトコルの実現方法を考察することが挙げられる．

[i] Koch と Walzer [5] は，特定の位置にだけ異なる絵柄をもつ追加カード列を利用することで，シャッフル操作の特定の置換を（安全性を保ったまま）“選ぶ” ことが可能であることを示した．

参考文献

[1] A.C. Yao, “Protocols for secure computations,” 2013 IEEE 54th Annual Symposium on Foundations of Computer Science, vol.00, pp.160–164, 1982.

[2] T. Nakai, Y. Tokushige, Y. Misawa, M. Iwamoto, and K. Ohta, “Efficient card-based cryptographic protocols for millionaires’ problem utilizing private permutations,” The 15th Cryptology and Network Security, eds. by S. Foresti and D. Persiano, vol.10052, pp.500–517, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2016.

[3] C. Crépeau and J. Kilian, “Discreet solitary games,” Advances in Cryptology — CRYPTO ’93, ed. by D.R. Stinson, vol.773, pp.319–330, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1994.

[4] T. Mizuki and H. Sone, “Six-card secure AND and four-card secure XOR,” Frontiers in Algorithmics, eds. by X. Deng, J.E. Hopcroft, and J. Xue, vol.5598, pp.358–369, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009.

[5] A. Koch and S. Walzer, “Foundations for actively secure card-based cryptography,” Cryptology ePrint Archive, Report 2017/423, 2017.

[6] I. Ueda, A. Nishimura, Y. Hayashi, T. Mizuki, and H. Sone, “How to implement a random bisection cut,” Theory and Practice of Natural Computing, eds. by C. Martín-Vide, T. Mizuki, and M.A. Vega-Rodríguez, vol.10071, pp.58–69, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2016.

[7] T. Nishida, T. Mizuki, and H. Sone, "Securely computing the three-input majority function with eight cards," Theory and Practice of Natural Computing, eds. by A.-H. Dediu, C. Martín-Vide, B. Truthe, and M.A. Vega-Rodríguez, vol.8273, pp.193–204, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2013.