上下カードプロトコルにおけるカード枚数の上界と下界

飯野 静流[1,a)]　李 陽[1]　崎山 一男[1]　宮原 大輝[1,2]

概要：物理的なカード組を用いて秘密計算を行うカードベース暗号は，1989 年に黒と赤の 2 色のデッキを用いて論理積を計算する AND プロトコルが考案されて以降，主に理論的側面から研究が進展してきた．近年では情報セキュリティ教育など，応用面に関しても徐々に広がりを見せている．本稿では，1 ビットを矢印の向きによって符号化する上下カード組を用いるプロトコルに着目する．上下カード組を用いるカードベースプロトコルにおいて，n 入力 AND および 3 入力多数決の計算に必要なカード枚数の下界に関する同著者らの成果がある．いずれも，2025 年に拡張された計算モデルを基にする厳密な証明である．本稿では，n 個の入力ビットの和がある値以上かどうかのみを出力するしきい値関数に焦点を当てる．まず，既存の 2 色カード組プロトコルに基づき，$n+1$ 枚のプロトコルを構成できることを示す．さらにこの $n+1$ 枚が最小枚数であることを証明する．しきい値関数は多数決関数を含む広い関数であり，今後はさらに広い対称関数が研究対象として挙げられる．

キーワード：カードベース暗号, 上下カードプロトコル, 閾値関数

Tight Bounds for Card-Based Protocols Using Up-Down Cards

Shizuru Iino[1,a)]　Yang Li[1]　Kazuo Sakiyama[1]　Daiki Miyahara[1,2]

Abstract: Since the proposal of an AND protocol in 1989 using a red-and-black two-color deck, the field of card-based cryptography (i.e, secure computation using physical playing cards) has been primarily studied from a theoretical perspective. In recent years, additionally, its scope has gradually expanded to include practical applications such as information security education. The present study focuses on protocols employing up-down cards, wherein a bit is encoded by the direction of an arrow. For such protocols, the authors have previously established lower bounds on the number of cards required to compute the n-input AND and the 3-input majority functions. These results were rigorously substantiated through a computational model that was extended in 2025. In this study, we direct our focus toward threshold functions, which generate a binary output indicative of whether the sum of n input bits attains a specified threshold. We demonstrate that an n-input threshold function can be realized using n+1 cards based on existing two-color deck protocols. We subsequently demonstrate that $n+1$ is the minimum number of cards required for secure computation of such functions. As threshold functions generalize majority functions, this result provides a foundation for future investigations into broader classes of symmetric functions.

1. はじめに

カードベース暗号は，主にトランプなどの物理的なカード組を用いて秘密計算を達成する暗号プロトコルである．初めてのカードベースプロトコルは 1989 年に Den Bore により考案された Five-Card Trick [2] であり，裏面が同一である黒と赤の 2 色のカードを使用し，2 つの入力の論理積を計算する．2 色のカードを用いてブール値は次のように表される．

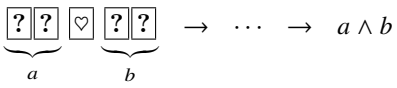
$$\boxed{\clubsuit}\boxed{\heartsuit} = 0, \quad \boxed{\heartsuit}\boxed{\clubsuit} = 1. \tag{1}$$

カード 2 枚を裏面に置いたとき，それらがビット $x \in \{0,1\}$ を示すならば，この 2 枚は x の**コミットメント**と呼ばれ，

[1] 電気通信大学
The University of Electro-Communications
[2] 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology
[a)] s.iino@uec.ac.jp

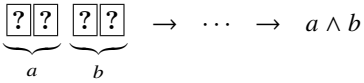
次のように表記する．

計算結果がコミットメントの出力で得られるプロトコルを**コミット型**プロトコル，コミットメントで得られないプロトコルを**非コミット型**プロトコルと呼ぶ．Five-Card Trick [2] は，2 入力の論理積を計算する非コミット型 **AND プロトコル**の一つであり，次のように追加のカードを 1 枚使用する．

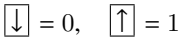


1.1 モチベーション

カードベース暗号における研究の主軸となるものは，プロトコルで使用するカード枚数の削減である．前述の Five-Card Trick [2] はカード 1 枚追加した計 5 枚のプロトコルであるが，2012 年に Mizuki ら [5] によって入力コミットメントのみ，すなわち追加カード無しで AND プロトコルを構成できることが示された．



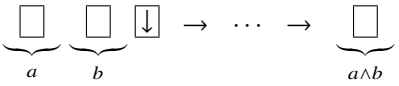
以上の 2 つのプロトコルはいずれも 2 色のカードによる符号化方式を用いるが，式 (1) を 1 色のカードのみの符号化方式に変更することでさらにカード枚数を削減できる．Mizuki と Shizuya [7] により考案された 1 色デッキ，すなわち**上下カード**は 1 枚のカードでビットを表現する．



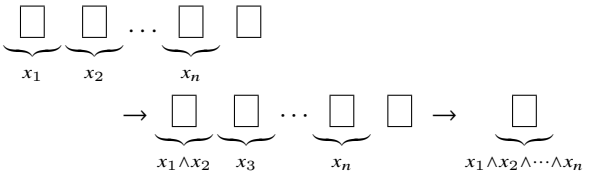
(2)

ここで裏面は上下対称の □ であり，2 色デッキ同様に裏面から表面の絵柄は区別できない．上下カードでは，裏面になった 1 枚のカードをコミットメントという．

この符号化方式を用いる 2 入力の AND プロトコルに，3 枚のコミット型プロトコル [7] が存在する．

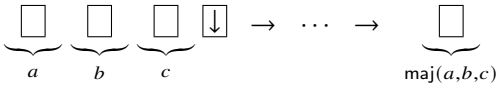


このプロトコルはコミット型であるので，$n\,(\geq 3)$ 入力 AND プロトコルについても追加カード 1 枚で構成できる．



2014 年に上下カードが考案されて以降，この追加カード 1 枚を削減できるかどうかは未解決であったが，2025 年に我々 [17] によって否定的に解決され，上下カードプロトコルの定式化 [12] に基づく厳密な証明により，その不可能性が示された．すなわち，n 入力 AND プロトコルの構成には追加のカードが 1 枚必要である．

この問題を契機に，上下カードプロトコルに必要なカード枚数の解明に取り組み，3 入力の多数決関数に関しても必要十分条件を示している [18]．まず 3 入力多数決プロトコルには，2 色カード組を用いる既存プロトコル [11] を基にして，4 枚のコミット型プロトコルを構成できる．



ここで $\mathsf{maj}\colon \{0,1\}^3 \to \{0,1\}$ は 3 入力多数決関数を表す．これに対して，コミット型・非コミット型のいずれについても 3 入力多数決プロトコルを構成するには，追加のカードが 1 枚必要となることを示した．

1.2 貢献

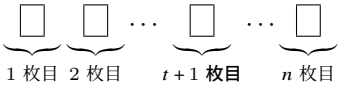
本稿では，$n\,(\geq 3)$ 入力多数決関数を計算するプロトコルのカード枚数の下界について証明を与える．さらに，多数決の仕組みを一般化したものとして，入力の 1 の数が予め指定したしきい値以上かどうかを判定する**しきい値関数**がある．本研究では，このしきい値関数に関する不可能性についても触れる．すなわち，上下カードを用いた n 枚 n 入力しきい値プロトコルが存在しないことを拡張計算モデル [6,12] に従って示す．追加のカード無しではしきい値関数を正しく計算することができず，また $n+1$ 枚のプロトコルも提案することで，$n+1$ 枚が最小枚数であることを示す．

しきい値関数とは，入力の合計値がある値 $t\,(\geq 0)$ を超えるか否か判定する関数のことである．n 入力しきい値関数 $\mathsf{TH}^t_n\colon \{0,1\}^n \to \{0,1\}$ は次の式で表される．

$$\mathsf{TH}^t_n(x_1,x_2,\ldots,x_n) = \begin{cases} 0 & \text{if } \sum_{i=1}^n x_i \leq t \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

ここで $t = \lfloor n/2 \rfloor$ のとき，これは多数決関数となり，$t = n-1$ のときは AND 関数となる．

提案プロトコルは，入力ビットを降順に**ソート**することで TH^t_n を計算する．すなわち，値が 1 となる入力コミットメントを左に，0 となる入力コミットメントを右に揃うようにソートを行う．このとき，$t+1$ 枚目のカードが正に TH^t_n のコミットメントとなるため，それを出力する．

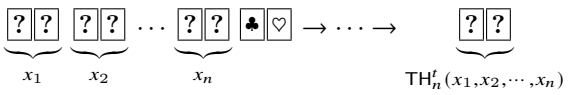


このアイデアによって計算する既存の 2 色デッキ $2n+2$ 枚プロトコル [16] があり，これを参考に 3 節で $n+1$ 枚の n 入力しきい値プロトコルを構成する．

証明では，しきい値の境目となる入力に着目する．しきい値が $t \geq \lceil n/2 \rceil$ のときは，1 の個数が t 個になる入力が境目の入力となり，しきい値が $t < \lceil n/2 \rceil$ のときには，1 の個数が $t+1$ 個の入力が境目の入力となる．しきい値 TH_n^t プロトコルの出力が 0 となる入力集合と 1 となる入力集合をそれぞれ $\mathcal{B}_0, \mathcal{B}_1$ とするとき，境目となる入力を $b_0 \in \mathcal{B}_0, b_1 \in \mathcal{B}_1$ とすると，それらの反転もそれぞれ同じ入力集合に属すこと（すなわち，$\overline{b_0} \in \mathcal{B}_,, \overline{b_1} \in \mathcal{B}_1$）を利用する．

1.3 関連研究

既存のプロトコルの多くは 2 色デッキを用いて構成されており，しきい値関数や多数決関数を計算するプロトコルにおいても，2 色デッキの研究が数多く行われてきた．Haga ら [3] により，初めてソーティングプロトコルが考案され，それを用いると容易にコミット型しきい値プロトコルを構築できることが示された．このプロトコルは入力カードと目印カードの合計 $2n+3$ 枚使用し，シャッフルを期待値 $2n+1+(n+1)\sum_{i=1}^{n}\frac{1}{i}$ 回行う．その後，先述した 2 色デッキのしきい値プロトコル [16] が考案され，カード枚数とシャッフル回数が大幅に削減された．



ここで用いられるソーティング手法を上下カード版に拡張したプロトコルを 3.1 節で構築する．

多数決プロトコルはより多くの研究が行われ，3 入力については最小枚数が判明している．まず追加カード 2 枚を用いるコミット型多数決プロトコルが Nishida ら [11] によって提案され，その後，Toyoda ら [14] は追加カードを用いない最小カード枚数プロトコルを考案した．n 入力については，追加カード 2 枚で多数決関数を含む対称関数を計算するコミット型プロトコルが Nishida ら [10] によって 2015 年に提案されて以降，進展は無い．

一方で秘匿操作[*1]を用いる**プライベート型**についても研究が進んでいる．まず 2017 年に Nakai ら [9] により 3 入力多数決関数を 4 枚のカードで計算可能であることが示された．続いて 2018 年，Watanabe ら [15] は同条件下で必要なカード枚数を 3 枚にまで削減するプロトコルを提案した．さらに 2022 年，Abe ら [1] は出力公開方法に条件を付与することで，3 入力多数決プロトコルを 3 枚のみで構成することに成功し，加えて n 入力多数決プロトコルを n 枚のみで構成可能であることを示した．そして Nakai ら [8] は n 入力しきい値プロトコルに対して，多数決の構成を拡張し，わずか $n+1$ 枚のカードで構成できることを示した．

[*1] 他のプレイヤーに情報が開示されない状態でカードの並び順や向きを変更する操作であり，例えばカードを背後に回して処理することで実現される．

2. 準備

本節では，上下カード組に限定した計算モデルを紹介し，プロトコルの基礎である正当性と安全性について定義し，例を挙げて確認する．以降では $i, j \in \mathbb{Z}$ に対して $[i, j]$ を i から j までの整数区間とする．ビット列もしくは組 x に対し，$x[i]$ は i 番目の値を表すものとし，添え字は 1 から始まる．

2.1 計算モデル

本節では，上下カード組を用いるカードベースプロトコルの計算モデル [6,12] を紹介する．上下カードは位置の入れ替えによる単なる置換にとどまらず，回転によってカードの絵柄そのものが変化する．このため，可変カードを許容するモデルとして，リース積により定義される拡張置換を用いて表現される．

$\Sigma \coloneqq \{\downarrow, \uparrow\}$ を**シンボル集合**とする．$c \in \Sigma$ に対して，$\frac{c}{?}$ を**表向きカード**，$\frac{?}{c}$ を**裏向きカード**と呼び，これらを総称して**カード**と呼ぶ．ここで，? は裏面を表す特殊なシンボルとする．

カードに対して次の関数を定義する．

- top：$\mathsf{top}\left(\frac{c}{?}\right) \coloneqq c$，$\mathsf{top}\left(\frac{?}{c}\right) \coloneqq ?$
- swap：$\mathsf{swap}\left(\frac{c}{?}\right) \coloneqq \frac{?}{c}$，$\mathsf{swap}\left(\frac{?}{c}\right) \coloneqq \frac{c}{?}$

Σ 上の全てのカードの集合を次のように表す．

$$C \coloneqq \left\{ \frac{c}{?}, \frac{?}{c} \;\middle|\; c \in \Sigma \right\}$$

d 個のカード組 $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_d) \in C^d$ を d 枚の**カード列**と呼ぶ．**可視**カード列 $\mathsf{vis}(\Gamma)$ を次のように定義する．

$$\mathsf{vis}(\Gamma) \coloneqq \left(\mathsf{top}(\alpha_1), \mathsf{top}(\alpha_2), \ldots, \mathsf{top}(\alpha_d)\right)$$

d 枚の全ての可視カード列の集合を次のように表す．

$$\mathsf{Vis}^d \coloneqq \left\{ \mathsf{vis}(\Gamma) \;\middle|\; \Gamma \in C^d \right\}$$

シンボル集合 Σ 上の関数として，次の 2 つを定義する．

- $\mathsf{id}(\downarrow) = \downarrow$ および $\mathsf{id}(\uparrow) = \uparrow$（恒等関数）．
- $\neg(\downarrow) = \uparrow$ および $\neg(\uparrow) = \downarrow$（シンボルを反転する関数）．

これらを Σ 上の関数から C 上の関数へ自然に拡張し，総称して**カード関数**と呼び，$P \coloneqq \{\mathsf{id}, \neg\}$ とする．

プロトコルは 4 つ組 (d, U, Q, A) で表され，各記号は次の通りである．

- $d \in \mathbb{N}$：カード枚数．
- $U \subseteq C^d$：入力集合．
- Q：状態集合であり，初期状態 q_0 および終了状態 q_f を含む．
- A：動作関数であり，現在の状態と可視カード列から，次の集合と動作を決定する．

$$A\colon (Q \setminus \{q_\mathrm{f}\}) \times \mathsf{Vis}^d \to Q \times \mathsf{Action}^d$$

ここで Action^d は d 枚のカード列に対する操作の集合であり，以降で定義する．

d 枚のカード列 $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_d) \in C^d$ に対し，操作集合 Action^d は次の動作を含む．

turn（**裏返す**） 集合 $T \subseteq [1,d]$ に対して，操作 (turn, T) は T に含まれる位置のカードを全て裏返す．

$$\mathsf{turn}_T(\Gamma) = (\beta_1, \beta_2, \ldots, \beta_d), \quad \beta_i = \begin{cases} \mathsf{swap}(\alpha_i), & i \in T \\ \alpha_i, & i \notin T \end{cases}$$

perm（**拡張置換**） d 次の対称群 S_d の元 $\pi \in S_d$ と，d 個のカード関数 $\vec{\phi} = (\phi_1, \ldots, \phi_d) \in P^d$ の組 $(\pi, \vec{\phi})$ に対し，操作 $(\mathsf{perm}, (\vec{\phi}, \pi))$ はカード列を π に従って並び替え，各位置に対応するカード関数を適用する．

$$\mathsf{perm}_\omega(\Gamma) = \left(\phi_1(\alpha_{\pi^{-1}(1)}), \ldots, \phi_d(\alpha_{\pi^{-1}(d)})\right)$$

ここで $\omega \coloneqq (\vec{\phi}, \pi)$ を**拡張置換**と呼ぶ．これはビット列を π によって並び替え，$\vec{\phi}$ によって各ビットを反転させるのと同値である．

shuf（**シャッフル**） 拡張置換の集合 $\Omega \subseteq P \wr S_d$（ただし $P \wr S_d$ は P と S_d のリース積）と，Ω 上の確率分布 $\mathcal{F}$ に対して，$(\mathsf{shuf}, \Omega, \mathcal{F})$ は Ω から $\mathcal{F}$ に従って拡張置換 $\omega \in \Omega$ を選び，これをカード列に適用する．

$$\mathsf{shuf}_{\Omega,\mathcal{F}}(\Gamma) = \mathsf{perm}_\omega(\Gamma), \quad \omega \leftarrow \mathcal{F}$$

ただし，選ばれた ω は公開されない，

result（**結果出力**） これは最終状態においてのみ実行され，裏向きで出力するカードの位置を指定する．$(\mathsf{result}, p_1, p_2, \ldots, p_\ell)$ と書き，$p_i \in [1,d]$ である．

2.2 正当性と安全性

n 入力のブール関数 $f\colon \{0,1\}^n \to \{0,1\}$ を計算するプロトコル $\mathcal{P}$ の正当性を次のように定義する．

定義 1 (正当性)**.** $\mathcal{P} = (d, U, Q, A)$ は以下の条件を満たすとき，ブール関数 f を計算するという．

- $d \geq n$.
- 各入力値 $b \in \{0,1\}^n$ に対して，入力集合 U はただ一つのカード列 $\Gamma^b = (\alpha_1, \ldots, \alpha_d)$ を含む．ここで各 $i \in [1,d]$ について以下が成り立つ．

$$\alpha_i = \begin{cases} \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., & \text{if } b[i] = 0 \\ \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., & \text{if } b[i] = 1 \end{cases} \tag{4}$$

これはすなわち i 番目の入力コミットメントを表している．

- 各入力値 $b \in \{0,1\}^n$ に対して，$\mathcal{P}$ は次のような最終カード列 $\Gamma = (\beta_1, \ldots, \beta_d)$ および動作 $(\mathsf{result}, p_1, \ldots, p_\ell)$ によって終了する．

$$(\beta_{p_1}, \ldots, \beta_{p_\ell}) \in \begin{cases} O_0 & \text{if } f(b) = 0 \\ O_1 & \text{if } f(b) = 1 \end{cases} \tag{5}$$

ここで，$O_0, O_1 \subseteq \left\{\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right\}^\ell$ は ℓ 枚の裏向きのカードからなるカード列の集合であり，$O_0 \cap O_1 = \emptyset$ を満たす．すなわち，O_0 および O_1 は最終カード列おいて出力値を符号化しており，これらが互いに交わらないことで出力値を一意に定める．特に $\mathcal{P}$ がコミット型であることは $\ell = 1$ と同義である．この場合は $O_0 = \{\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\}, O_1 = \{\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\}$ となる．

例 1. 次の 3 枚 3 入力しきい値 TH_3^1 プロトコルは正当性を満たす．

$$\mathcal{P}^{\mathrm{ex1}} = (3, U, \{q_0, q_\mathrm{f}\}, A)$$

ただし，入力集合 U は次のとおりである．

$$U = \left\{\left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right)\right\}$$

動作関数 A は次のとおりである．

$$A(q_0, (?,?,?)) = (q_\mathrm{f}, (\mathsf{result}, 1, 2, 3))$$

すなわち，入力コミットメントがそのまま出力となるプロトコルである．このとき各入力値 $b \in \{0,1\}^3$ における最終カード列 $(\beta_1, \beta_2, \beta_3)$ は式 (5) を満たし，O_0, O_1 は次の集合である．

$$O_0 = \left\{\left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right)\right\}$$

$$O_1 = \left\{\left(\left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\downarrow\end{smallmatrix}\right.\right), \left(\left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right., \left\{\begin{smallmatrix}?\\\uparrow\end{smallmatrix}\right.\right)\right\}$$

次に安全性を定義する．ここで $\mathcal{P}$ におけるカード列の**トレース**を $(\Gamma_0, \Gamma_1, \ldots, \Gamma_\mathrm{f})$ とし，各 Γ_{i+1} は Γ_i に対してある操作を適用することで得られるカード列である．さらに，**可視列トレース**とは $(\mathsf{top}(\Gamma_0), \mathsf{top}(\Gamma_1), \ldots, \mathsf{top}(\Gamma_\mathrm{f}))$ を指す．

定義 2 (安全性)**.** ブール関数 f を計算する $\mathcal{P} = (d, U, Q, A)$ は以下の条件を満たすとき，安全であるという．

- 入力集合 U 上の確率分布を $\mathcal{M}$ とし，入力カード列の確率変数（$\mathcal{M}$ に従う）を M，$\mathcal{P}$ の可視列トレースの確率変数を V とするときに，M と V が確率的に独立である．
- 各 $z \in \{0,1\}$ について，M_z を $f(M) = z$ という条件下での M の条件付き確率分布とする．result 動作によって得られる出力列を表す確率変数を $R^{M,P}$ とする．このとき各 $z \in \{0,1\}$ に対して，$f(M) = z$ が与えられた条件下で M と $R^{M,P}$ が条件付き独立である．

例 2. プロトコル $\mathcal{P}^{\mathrm{ex1}}$ は安全性を満たさない．これは，入力カード列がそのまま出力カード列となるため，$M = R^{M,P}$

が成立し，完全に従属関係にある．より詳細には，式 (5) から得られる出力カード列に対して，次式が成り立つ．

$$\Pr[(\beta_1,\beta_2,\beta_3)=(\tfrac{?}{\downarrow},\tfrac{?}{\downarrow},\tfrac{?}{\downarrow}) \mid b=000]=1$$
$$\neq \Pr[(\beta_1,\beta_2,\beta_3)=(\tfrac{?}{\downarrow},\tfrac{?}{\downarrow},\tfrac{?}{\downarrow}) \mid b=001]=0$$

したがって，プロトコル $\mathcal{P}^{\mathrm{ex1}}$ は安全性の要件を満たしていない．

3. しきい値プロトコル

本節では，上下カードを用いるしきい値関数プロトコルの構成と，その疑似コードを示す．

3.1 構成

上下カードを $n+1$ 枚用いる n 入力しきい値プロトコルを構成する．

(1) 各 $x_i \in \{0,1\}$ の入力コミットメントと追加カード $\boxed{\downarrow}$ を次のように裏面にして並べる．

$$\underbrace{\square}_{x_1}\underbrace{\square}_{x_2}\cdots\underbrace{\square}_{x_n}\boxed{\downarrow} \rightarrow \underbrace{\square}_{x_1}\underbrace{\square}_{x_2}\cdots\underbrace{\square}_{x_n}\underbrace{\square}_{0}$$

(2) $k=2$ から $k=n$ まで次の操作を繰り返す．

(a) 次の “特殊” なシャッフル操作を行う．全てのカード列を同時にランダムな回数だけ回転させ，全てそのままにするか，全ての値を反転させる．さらに反転させる際には同時に，1 枚目から $k-1$ 枚目までを逆順に並び替える．$k=4$ かつ $n=4$ のときの例を示す．ここでスペースの都合上，カードの表記を省略し，各コミットメントの値を $y_i \in \{0,1\}$ として値のみを並べている．

$$y_1y_2y_3y_4y_5 \rightarrow \begin{cases} y_1y_2y_3y_4y_5 & \text{with prob. } 1/2 \\ \overline{y}_3\overline{y}_2\overline{y}_1\overline{y}_4\overline{y}_5 & \text{with prob. } 1/2 \end{cases}$$

(b) k 枚目をめくり $\boxed{\downarrow}$ ならばそのまま，$\boxed{\uparrow}$ ならばそれを左端に配置し，裏に戻す．

$$y_1y_2\cdots y_{k-1}y_k \rightarrow \begin{cases} y_1y_2\cdots y_{k-1}0 & \rightarrow \quad y_1y_2\cdots y_{k-1}0 \\ y_1y_2\cdots y_{k-1}1 & \rightarrow \quad 1y_ky_1y_2\cdots y_{k-1} \end{cases}$$

(3) $k=n+1$ としたときのステップ 2(a) のシャッフル操作を行う．

(4) 右端をめくり $\boxed{\downarrow}$ ならそのままにし，$\boxed{\uparrow}$ なら全てのカードを反転させると同時に 1 枚目から n 枚目までを逆順に並び替える．めくったカードは除外する．

(5) 以上の操作で n 個の入力コミットメントが降順にソートされるため，$t+1$ 枚目が出力となる．

このプロトコルの正当性と安全性を説明する．まず安全性は明らかに成立し，めくるカードは直前のシャッフル操作によって 1/2 の確率で $\boxed{\downarrow}$ もしくは $\boxed{\uparrow}$ であり，入力と出力の値は漏れない．

正当性についてはまず，ステップ 2(a) のシャッフル操作によって，1 枚目から $k-1$ 枚目までのカード列は，x_1 から x_{k-1} のコミットメントを降順にソート（シャッフル操作のそのままの場合に対応），もしくは $\overline{x}_1$ から $\overline{x}_{k-1}$ のコミットメントを昇順にソート（シャッフル操作の反転した場合に対応）したカード列になることが本質である．したがってステップ 2(b) で x_k もしくは $\overline{x}_k$ のどちらの場合の値を得たとしても，0 であればそのままにして，1 であればソートされている列の左端に配置することにより，新たに 1 枚目から k 枚目がソートされたカード列となる．最後にステップ 3,4 によって，フラグの役割を果たしている $n+1$ 枚目をシャッフル後にめくることにより，入力コミットメントが降順にソートされたカード列を得ることができる．

3.2 定式化の具体例

例 3. 具体的に $n=2$ のときのプロトコルを $\mathcal{P}^{\mathsf{TH}}$ として，正当性と安全性を確認する．

$$\mathcal{P}^{\mathsf{TH}}=(3,U,\{q_0,q_1,q_2,q_3,q_4,q_5,q_6,q_7,q_{\mathrm{f}}\},A)$$

ここで入力集合 U は次の通りである．

$$U=\left\{\left(\tfrac{?}{\downarrow},\tfrac{?}{\downarrow},\tfrac{?}{\downarrow}\right),\left(\tfrac{?}{\uparrow},\tfrac{?}{\downarrow},\tfrac{?}{\downarrow}\right),\left(\tfrac{?}{\downarrow},\tfrac{?}{\uparrow},\tfrac{?}{\downarrow}\right),\left(\tfrac{?}{\uparrow},\tfrac{?}{\uparrow},\tfrac{?}{\downarrow}\right)\right\}$$

ここで 3 枚目が 0 の追加コミットメントであり，他が入力コミットメントである．また動作関数 A は各状態 q_i に対して次のように表される．

- $A(q_0,(?,?,?))=(q_1,(\mathsf{shuf},\Omega_1,\mathcal{F}_1))$，ただし
$$\Omega_1=\{((\mathrm{id},\mathrm{id},\mathrm{id}),(\neg,\neg,\neg),\mathrm{id})\}$$
であり，$\mathcal{F}_1$ は Ω_1 上の一様分布である．すなわち，3 枚のカードが同時にランダムな回数だけ回転する．
- $A(q_1,(?,?,?))=(q_2,(\mathsf{turn},\{2\}))$.
- $A(q_2,(?,\downarrow,?)) = (q_3,(\mathsf{perm},((\mathrm{id},\mathrm{id},\mathrm{id}),\mathrm{id})))$ かつ $A(q_2,(?,\uparrow,?))=(q_3,(\mathsf{perm},((\mathrm{id},\mathrm{id},\mathrm{id}),(1\,2))))$.
- $A(q_3,(?,\downarrow,?))=(q_4,(\mathsf{turn},\{2\}))$ かつ $A(q_3,(\uparrow,?,?))=(q_4,(\mathsf{turn},\{1\}))$.
- $A(q_4,(?,?,?))=(q_5,(\mathsf{shuf},\Omega_1,\mathcal{F}_1))$.
- $A(q_5,(?,?,?))=(q_6,(\mathsf{turn},\{3\}))$.
- $A(q_6,(?,?,\downarrow)) = (q_7,(\mathsf{perm},((\mathrm{id},\mathrm{id},\mathrm{id}),\mathrm{id})))$ かつ $A(q_6,(?,?,\uparrow))=(q_7,(\mathsf{perm},((\neg,\neg,\neg),(1\,2))))$.
- $A(q_7,(?,?,\downarrow))=(q_{\mathrm{f}},(\mathsf{result},t+1))$.

$\mathcal{P}^{\mathsf{TH}}$ は式 (4) と式 (5) のどちらも満たすため，定義 1 を満たす．また，$\mathcal{P}^{\mathsf{TH}}$ は定義 2 を満たす．これはカードをめくって $\boxed{\downarrow}$ もしくは $\boxed{\uparrow}$ が現れる確率が，常に入力値に依らず 1/2 なためである．

Algorithm 1 Threshold Protocol $\mathcal{P} = (n+1, U, Q, A)$

Require: $U = (x_1, x_2, \ldots, x_n, 0)$
Ensure: result(t)
1: **for** $k = 2$ to n **do**
2: $\quad(\mathsf{shuf}, \Omega_k, \mathcal{F}_k)$
3: $\quad(\mathsf{turn}, \{k\})$
4: $\quad$**if** $\downarrow$ is appeared **then**
5: $\quad\quad(\mathsf{turn}, \{k\})$
6: $\quad$**else**
7: $\quad\quad\mathsf{perm}, (\{\mathsf{id}\}^n, (1\,2\cdots k))$
8: $\quad\quad(\mathsf{turn}, \{1\})$
9: $\quad$**end if**
10: **end for**
11: $(\mathsf{shuf}, \Omega_{n+1}, \mathcal{F}_{n+1})$
12: $(\mathsf{turn}, \{n+1\})$
13: **if** $\uparrow$ is appeared **then**
14: $\quad\mathsf{perm}(\{\neg\}^{n+1}, \Pi_{i=1}^{\lfloor n/2 \rfloor}(i\ n-i+1))$
15: **end if**
16: $(\mathsf{result}, t+1)$

3.3 疑似コード

前節の計算モデルに倣い，提案プロトコルの疑似コードをアルゴリズム 1 に示す．ここで

$$\Omega_k \coloneqq \{(\{\mathsf{id}\}^{n+1}, \mathsf{id}), (\{\neg\}^{n+1}, \Pi_{i=1}^{\lfloor k/2 \rfloor}(i\ k-i))\}$$

であり，$\mathcal{F}_k$ は Ω_k 上の一様分布である．

3.4 議論

提案プロトコルにおけるステップ 2(a) の特殊なシャッフル操作を実装する方法は不明である．2 色カード組の研究 [4] と同様に，実装可能なシャッフル操作[*2]を用いるという条件下でのカード枚数の必要十分条件も興味深く，将来的な研究テーマとして挙げられる．実際に Iwasaki [?] は，上下カードの反転操作と入れ替え操作を同時に含むようなシャッフル操作を**連動操作**と呼び，連動操作を用いない条件下における有限時間の AND プロトコルは 4 枚が最適であることを示している．

4. しきい値プロトコルの不可能性

本節では，n 入力しきい値プロトコルの不可能性を背理法により示す．先に，証明を簡潔に行うために補題をいくつか導く．

4.1 有益な補題

最初の補題は，出力値が異なるような 2 つの入力に対応するカード列は，シャッフル操作によって決して “混ざらない” ことをいう．

補題 1. プロトコル $\mathcal{P} = (d, U, Q, A)$ は関数 $f\colon \{0,1\}^n \to \{0,1\}$ を計算するとする．カード関数 P と d 次の対称群 S_d のリース積に属する任意の $\Omega \subseteq P \wr S_d$ に対するシャッフル操作 $(\mathsf{shuf}, \Omega, \mathcal{F})$ が，最初にカード列に適用されることを仮定する．このとき，$f(b_0) = 0$ かつ $f(b_1) = 1$ となる任意の入力 $b_0, b_1 \in \{0,1\}^n$ と互いに異なる $\omega, \omega' \in \Omega$ について次式が成り立つ．

$$\omega(\Gamma^{b_0}) \neq \omega'(\Gamma^{b_1}), \quad \text{ただし } \Gamma^{b_0}, \Gamma^{b_1} \in U$$

証明． $\omega, \omega' \in \Omega$ に対して $\omega(\Gamma^{b_0}) = \omega'(\Gamma^{b_1})$ と仮定して矛盾を導く．このとき，プロトコルは以降の全ての操作を同一のカード列に対して行うことから，b_0 と b_1 は同一の最終カード列を得る．そこで，result による出力カード列を Γ_{out} とすると，定義 1 から b_0 については $\Gamma_{\text{out}} \in O_0$，$b_1$ については $\Gamma_{\text{out}} \in O_1$ となるため，$\Gamma_{\text{out}} \in O_0 \cap O_1$ であるが，前提から $O_0 \cap O_1 = \varnothing$ であり矛盾する．したがって $\omega(\Gamma^{b_0}) \neq \omega'(\Gamma^{b_1})$ が成り立つ． □

次の補題は安全性に関してであり，カードが 1 枚めくられる際は，どの入力においても $\boxed{\downarrow}$ もしくは $\boxed{\uparrow}$ の 2 通りが確率的に見える必要があることをいう．

補題 2. プロトコル $\mathcal{P} = (d, U, Q, A)$ を関数 $f\colon \{0,1\}^n \to \{0,1\}$ を計算するプロトコルとする．任意の $\Omega \subseteq P \wr S_d$ に対するシャッフル操作 $(\mathsf{shuf}, \Omega, \mathcal{F})$ が最初の操作であり，その直後に turn 操作を行うとする．このとき，任意の入力 $b \in \{0,1\}^n$ に対して，次式を満たすような異なる $\omega, \omega' \in \Omega$ が存在する．

$$\omega(\Gamma^b) \neq \omega'(\Gamma^b) \quad \text{ただし } \Gamma^b \in U$$

証明． すべての $\omega, \omega' \in \Omega$ に対して $\omega(\Gamma^b) = \omega'(\Gamma^b)$ が成り立つような入力 $b \in \{0,1\}^n$ の存在を仮定し，矛盾を導く．このとき，シャッフル後のカード列はいずれの $\omega \in \Omega$ が選択されても同一であり，シャッフル操作により生成されるはずのランダム性が確保できていない．次に turn 操作を行うことを考えると，これにより可視化されるカード列は shuf 操作後のカード列により一意に定まる．したがって，可視列トレースから入力 b を推測できてしまう．これは定義 2 に反するため，各入力 b に対して $\omega(\Gamma^b) \neq \omega'(\Gamma^b)$ が成り立つような異なる $\omega, \omega' \in \Omega$ が存在する． □

最後の補題は，カード列の “対称性” に関する性質である．

補題 3. $\Omega \subseteq P \wr S_d$ を置換集合とする．任意の $\Gamma \in C^d$ と $\omega_1, \omega_2 \in \Omega$ に対して，$\omega_1(\Gamma) \neq \omega_2(\Gamma)$ が成立すると仮定する．このとき $\omega_1(\Gamma') \neq \omega_2(\Gamma')$ が成立するような $\Gamma' \in C^d$ が存在し，$\omega = (\{\neg\}^n, \mathsf{id})$ に対して $\Gamma' = \omega(\Gamma')$ である．

証明． $\omega_1(\Gamma') = \omega_2(\Gamma')$ を仮定すると，$\{\neg\}^n$ が可換であることから，次式が成り立つ．

$$\begin{aligned}\omega_1(\Gamma') = \omega_2(\Gamma') &\Leftrightarrow \omega_1(\omega(\Gamma)) = \omega_2(\omega(\Gamma)) \\ &\Leftrightarrow \omega(\omega_1(\Gamma)) = \omega(\omega_2(\Gamma)) \\ &\Leftrightarrow \omega_1(\Gamma) = \omega_2(\Gamma)\end{aligned}$$

[*2] これ自体の範囲も興味深く，これまでにトルネードシャッフルと呼ばれる実装方法が提案されている [13].

これは前提に矛盾する．□

4.2 証明

以上の補題を用いてしきい値プロトコルの不可能性を証明する．

定理 1. TH_n^t を計算する安全な n 枚プロトコルは存在しない．ただし n が奇数のとき，$t \neq \lfloor n/2 \rfloor$ である．

証明. $\mathcal{P} = (n, U, Q, A)$ が TH_n^t を計算すると仮定する．入力集合 U は次のように示される．

$$U = \left\{\Gamma^b \mid b \in \{0,1\}^n\right\}, \quad \text{where } \Gamma^b[i] = \begin{cases} \overset{?}{\downarrow} & \text{if } b[i] = 0 \\ \overset{?}{\uparrow} & \text{if } b[i] = 1 \end{cases}$$

正当性と安全性の例で確認した $\mathcal{P}^{\mathrm{ex1}}$ のように，入力は一部すら開示できず出力できないため，turn 操作や result 操作の前に shuf 操作が必要である．そこで任意の $\Omega \subseteq P \wr S_n$ と確率分布 $\mathcal{F}$ に対して $(\mathsf{shuf}, \Omega, \mathcal{F})$ の適用を考え，これによる各 Γ^b の遷移を観察する．

まず $t \geq \lceil n/2 \rceil$ とし，$b_0 = \{0\}^n$ とすると，$\mathsf{TH}_n^t(b_0) = 0$ である．ここで Γ^{b_0} はシャッフル操作によって次の 2 通りに遷移する．

- 任意の異なる $\omega, \omega' \in \Omega$ に対して $\omega(\Gamma^{b_0}) = \omega'(\Gamma^{b_0})$ のとき，これは補題 2 より新たなシャッフル操作が必要となり，turn 操作や result 操作は許されない．
- $\omega(\Gamma^{b_0}) \neq \omega'(\Gamma^{b_0})$ となる異なる ω, ω' が存在するとき，Ω の任意の元は U 上の全単射置換であるため，$\omega(\Gamma^{b_0}) = \omega'(\Gamma^{b_0'})$ を満たすある入力 $b_0' \in \{0,1\}^n$ が存在する．さらに補題 1 から $\mathsf{TH}_n^t(b_0') = 0$ である．ここで b_0' における 1 の個数がちょうど t 個であるとする．補題 3 より，b_0, b_0' の各ビットを反転させた $\overline{b_0}, \overline{b_0'}$ に対して $\omega(\Gamma^{\overline{b_0}}) = \omega'(\Gamma^{\overline{b_0'}})$ も成立する．しかしながら

$$\mathsf{TH}_n^t(\overline{b_0}) = 1 \neq \mathsf{TH}_n^t(\overline{b_0'}) = 0$$

であり，補題 1 に矛盾する．これは $t \geq \lceil n/2 \rceil$ のとき，$\overline{b_0'}$ における 1 の個数は $n - t \leq t$ であるためである．

また $t < \lfloor n/2 \rfloor$ のときは，$b_1 = \{1\}^n$ として同様の議論により矛盾が導かれる．ただし b_1' における 1 の個数は $t+1$ となる．□

4.3 多数決プロトコルの不可能性

しきい値プロトコルの不可能性から，多数決プロトコルの不可能性が系として得られる．これはしきい値 $t = \lfloor \frac{n}{2} \rfloor$ をとる関数が多数決関数であるからである．

系 1. n が偶数のとき，n 入力多数決関数を計算する安全な n 枚プロトコルは存在しない．

しかし n が奇数のときについては一般的な法則を見つけられていないため，今後の課題とする．

5. おわりに

本稿では，n 入力しきい値プロトコルに対して，追加カードの削減が不可能であることを明らかにした．上下カードの遷移の可能性に注目することで，正当性と安全性を同時に満たすプロトコルが存在しないことを示した．また，しきい値関数の一例として多数決関数の不可能性についても触れた．本研究により，カードベース暗号におけるカード枚数の最適性についての未解決問題が解決された．

参考文献

[1] Abe, Y., Nakai, T., Kuroki, Y., Suzuki, S., Koga, Y., Watanabe, Y., Iwamoto, M., Ohta, K.: Efficient card-based majority voting protocols. New Gener. Comput. **40**, 173–198 (2022), https://doi.org/10.1007/s00354-022-00161-7

[2] Boer, B.D.: More efficient match-making and satisfiability the five card trick. In: Quisquater, J.J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 208–217. Springer, Heidelberg (1990), https://doi.org/10.1007/3-540-46885-4_23

[3] Haga, R., Toyoda, K., Shinoda, Y., Miyahara, D., Shinagawa, K., Hayashi, Y., Mizuki, T.: Card-based secure sorting protocol. In: Cheng, C.M., Akiyama, M. (eds.) Advances in Information and Computer Security. LNCS, vol. 13504, pp. 224–240. Springer, Cham (2022), https://doi.org/10.1007/978-3-031-15255-9_12

[4] Koch, A., Walzer, S., Härtel, K.: Card-based cryptographic protocols using a minimal number of cards. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology—ASIACRYPT 2015. LNCS, vol. 9452, pp. 783–807. Springer, Berlin, Heidelberg (2015), https://doi.org/10.1007/978-3-662-48797-6_32

[5] Mizuki, T., Kumamoto, M., Sone, H.: The five-card trick can be done with four cards. In: Wang, X., Sako, K. (eds.) Advances in Cryptology—ASIACRYPT 2012. LNCS, vol. 7658, pp. 598–606. Springer, Berlin, Heidelberg (2012), https://doi.org/10.1007/978-3-642-34961-4_36

[6] Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. Int. J. Inf. Secur. **13**(1), 15–23 (2014), https://doi.org/10.1007/s10207-013-0219-4

[7] Mizuki, T., Shizuya, H.: Practical card-based cryptography. In: Ferro, A., Luccio, F., Widmayer, P. (eds.) Fun with Algorithms. LNCS, vol. 8496, pp. 313–324. Springer, Cham (2014), https://doi.org/10.1007/978-3-319-07890-8_27

[8] Nakai, T., Misawa, Y., Tokushige, Y., Iwamoto, M., Ohta, K.: Secure computation for threshold functions with physical cards: Power of private permutations. New Gener. Comput. **40**, 95–113 (2022), https://doi.org/10.1007/s00354-022-00153-7

[9] Nakai, T., Shirouchi, S., Iwamoto, M., Ohta, K.: Four cards are sufficient for a card-based three-input voting protocol utilizing private permutations. In: Shikata, J. (ed.) Information Theoretic Security. LNCS, vol. 10681, pp. 153–165. Springer, Cham (2017), https://doi.org/10.1007/978-3-319-72089-0_9

[10] Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Card-based protocols for any Boolean function. In: Jain, R.,

Jain, S., Stephan, F. (eds.) Theory and Applications of Models of Computation. LNCS, vol. 9076, pp. 110–121. Springer, Cham (2015), https://doi.org/10.1007/978-3-319-17142-5_11

[11] Nishida, T., Mizuki, T., Sone, H.: Securely computing the three-input majority function with eight cards. In: Dediu, A.H., Martín-Vide, C., Truthe, B., Vega-Rodríguez, M.A. (eds.) Theory and Practice of Natural Computing. LNCS, vol. 8273, pp. 193–204. Springer, Berlin, Heidelberg (2013), https://doi.org/10.1007/978-3-642-45008-2_16

[12] Shinagawa, K., Nuida, K.: Card-based protocols imply psm protocols. In: Beyersdorff, O., Pilipczuk, M., Pimentel, E., Thàng, N.K. (eds.) 42nd International Symposium on Theoretical Aspects of Computer Science (STACS 2025). Leibniz International Proceedings in Informatics (LIPIcs), vol. 327, pp. 72:1–72:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2025). DOI: 10.4230/LIPIcs.STACS.2025.72, https://drops.dagstuhl.de/opus/volltexte/2025/18999/, article No. 72

[13] Shinagawa, K., Nuida, K., Nishide, T., Hanaoka, G., Okamoto, E.: Committed AND protocol using three cards with more handy shuffle. In: 2016 International Symposium on Information Theory and Its Applications. pp. 700–702. IEEE (2016), https://ieeexplore.ieee.org/document/7840515/

[14] Toyoda, K., Miyahara, D., Mizuki, T.: Another use of the five-card trick: Card-minimal secure three-input majority function evaluation. In: Adhikari, A., Küsters, R., Preneel, B. (eds.) Progress in Cryptology—INDOCRYPT 2021. LNCS, vol. 13143, pp. 536–555. Springer, Cham (2021), https://doi.org/10.1007/978-3-030-92518-5_24

[15] Watanabe, Y., Kuroki, Y., Suzuki, S., Koga, Y., Iwamoto, M., Ohta, K.: Card-based majority voting protocols with three inputs using three cards. In: 2018 International Symposium on Information Theory and Its Applications. pp. 218–222 (2018), https://doi.org/10.23919/ISITA.2018.8664324

[16] 四方 隼人, 水木 敬明: 効率的なコミット型閾値関数カードベースプロトコル. In: Symposium on Cryptography and Information Security (2024)

[17] 飯野 静流, 李 陽, 崎山 一男, 宮原 大輝: n 枚 AND プロトコルの不可能性. In: Symposium on Cryptography and Information Security (2025)

[18] 飯野 静流, 李 陽, 崎山 一男, 宮原 大輝: 上下カードプロトコルの枚数下界に関する一考察. In: 情報セキュリティ研究会. ISEC2025-67 (2025)