Impossibility of Four-Card AND Protocols with a Single Closed Shuffle*

Shizuru Iino[1], Shota Ikeda[2], Kazumasa Shinagawa[3,4], Yang Li[1], Kazuo Sakiyama[1], and Daiki Miyahara[1,4]

[1] The University of Electro-Communications, Tokyo, Japan
{s.iino,miyahara}@uec.ac.jp
[2] Ibaraki University, Ibaraki, Japan
[3] University of Tsukuba, Ibaraki, Japan
[4] National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

Abstract. Card-based cryptography performs a secure computation using a deck of playing cards through a series of physical actions, such as shuffling a sequence of cards. In 1989, the first card-based protocol called the five-card trick was proposed to compute the two-input AND function. The five-card trick needs a practical shuffling action called a random cut only once, which shifts a sequence of cards by a random offset. Subsequent research aimed to extend the five-card trick to compute any Boolean function and to reduce the number of cards required to compute it, but Mizuki, Kumamoto, and Sone in 2012 showed that the five-card trick itself can be done with four cards. This Mizuki–Kumamoto–Sone protocol uses the minimal number of cards required for computing two-input Boolean functions (as long as we encode an input bit with two cards); however, it needs a practical step of shuffling action twice, and the question of minimizing the number of steps, particularly the number of shuffling, remains an open problem. In this study, we negatively answer this problem; we prove that any four-card AND protocol cannot be realized using only a single practical shuffle. This implies that the Mizuki–Kumamoto–Sone protocol utilizes the minimal number of practical shuffles. For this, we enumerate all possible practical shuffles and prove that applying any one of them only once either cannot compute the AND function or leak information about the input.

Keywords: Card-based cryptography · Secure computation · Logical AND function · Impossibility proof

1 Introduction

Card-based cryptography enables a secure computation using a deck of physical cards. The most well-known card-based protocol is the five-card trick, proposed by Den Boer

* This paper appears in Proceedings of CANS 2025. This version of the contribution has been accepted for publication, after peer review but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1007/978-981-95-4434-9_10. Use of this Accepted Version is subject to the publisher's Accepted Manuscript terms of use: https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms.

in 1989 [3]. Given $a, b \in \{0, 1\}$, this protocol computes $a \wedge b$ without leaking any information about a, b. In this setting, we use a deck consisting of two colors: black $\boxed{\clubsuit}$ and red $\boxed{\heartsuit}$, with identical backs $\boxed{?}$. Boolean values are encoded based on the arrangement of one black and one red card, as follows:

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

This pair is called a *commitment* to x if it represents the value of a bit $x \in \{0, 1\}$. The five-card trick requires two input commitments to a, b along with one helping card $\boxed{\heartsuit}$:

$$\underbrace{\boxed{?}\boxed{?}}_{a}\underbrace{\boxed{?}\boxed{?}}_{b}\boxed{\heartsuit} \to \cdots \to a \wedge b.$$

Here, the first four cards denote the commitments to a, b. The five-card trick is called a *non-committed format* protocol, because the output in the five-card trick is not in the form of a commitment. A protocol is in *committed format* if it outputs a commitment to a desired value.

1.1 Mizuki–Kumamoto–Sone AND protocol

In 2012, Mizuki, Kumamoto, and Sone [10] showed that the five-card trick [3] can be done with four cards, i.e., they proposed a four-card non-committed format AND protocol, which needs no helping card. We call this protocol the *MKS protocol* in short.

$$\underbrace{\boxed{?}\boxed{?}}_{a}\underbrace{\boxed{?}\boxed{?}}_{b} \to \cdots \to a \wedge b.$$

Clearly, this protocol uses the minimal number of cards as long as we follow the encoding rule in Eq. (1), because it only uses two input commitments. The MKS protocol proceeds as follows.

1. Apply a *random bisection cut*, which bisects the sequence and randomly swaps the two halves and is denoted by $[\cdots|\cdots]$ as follows:

$$\left[\boxed{?}\boxed{?}\middle|\boxed{?}\boxed{?}\right] \to \boxed{?}\boxed{?}\boxed{?}\boxed{?}.$$

2. Apply a *random cut* to the middle two cards, which applies a random cyclic shift to the sequence and is denoted by $\langle\cdots\rangle$ as follows:

$$\boxed{?}\left\langle\boxed{?}\boxed{?}\right\rangle\boxed{?} \to \boxed{?}\boxed{?}\boxed{?}\boxed{?}.$$

3. Reveal the second card;
   (a) if it is $\boxed{\clubsuit}$, then reveal the forth to obtain the value of $a \wedge b$ as follows:

$$\boxed{?}\boxed{?}\boxed{?}\boxed{?} \to \begin{cases} \boxed{?}\boxed{\clubsuit}\boxed{?}\boxed{\heartsuit} & \text{if } a \wedge b = 0, \\ \boxed{?}\boxed{\clubsuit}\boxed{?}\boxed{\clubsuit} & \text{if } a \wedge b = 1. \end{cases}$$

Table 1: Sequences of cards in the MKS protocol [10]

(a, b)	Initial sequence	After step 2	After step 3
$(1, 1)$	♡♣♡♣	♡♣♡♣ or ♡♡♣♣	?♣?♣ or ♡♡??
$(0, 0)$	♣♡♣♡	♣♣♡♡ or ♣♡♣♡	?♣?♡ or ♣♡??
$(0, 1)$	♣♡♡♣	♡♣♣♡ or ♣♡♡♣	?♣?♡ or ♣♡??
$(1, 0)$	♡♣♣♡	♡♣♣♡ or ♣♡♡♣	?♣?♡ or ♣♡??

(b) if it is ♡, reveal the first:

$$??\!??\!?? \rightarrow \begin{cases} ♣♡?? & \text{if } a \wedge b = 0, \\ ♡♡?? & \text{if } a \wedge b = 1. \end{cases}$$

See Table 1 to show the correctness and the security: The initial sequence has four possibilities, which have a one-to-one correspondence with the four input values. Through the application of a random bisection cut and a random cut in steps 1 and 2 respectively, each of the four possible sequences is randomized as shown in the third column. For example, the initial sequence ♡♣♡♣ when $(a, b) = (1, 1)$ is randomized into the two sequences of ♡♣♡♣ and ♡♡♣♣ with equal probability, because the two middle cards are swapped randomly. Note that for each input, the initial sequence is randomized into exactly two sequences with equal probability. Note furthermore that these actions are performed with all the cards facing down, and hence, no information about the input value is leaked. After step 3, observe that only the case of $(a, b) = (1, 1)$ differs from the other three input cases, regardless of whether the color of the second card revealed is red or black. Moreover, the other three input cases are indistinguishable from the colors that appeared. For these reasons, the MKS protocol can determine $a \wedge b$ and leaks no further information.

1.2 Motivation

The number of cards in the MKS protocol [10] is optimal with respect to the encoding scheme because four cards are needed to input two bits. To discuss the number of shuffles in the MKS protocol, we need to give a formal definition of shuffles.

Let S_n denote the symmetric group on degree $n \in \mathbb{N}$. A *shuffle* acted on a sequence of n cards is defined as a pair of a permutation set $\Pi \subseteq S_n$ and a probability distribution $\mathcal{F}$ on Π, and is denoted by $(\mathsf{shuf}, \Pi, \mathcal{F})$ [11]. By applying a shuffle $(\mathsf{shuf}, \Pi, \mathcal{F})$ to a sequence of n cards, a permutation π is drawn from Π according to $\mathcal{F}$ and the sequence is rearranged by the chosen permutation π. Here, the chosen permutation π is hidden from all parties, and only Π and $\mathcal{F}$ are public information in the shuffle. A shuffle $(\mathsf{shuf}, \Pi, \mathcal{F})$ is called *uniform* if $\mathcal{F}$ is a uniform distribution over Π, *closed* if Π forms a group under composition of permutations, and *uniform closed* if it is uniform and closed. When $(\mathsf{shuf}, \Pi, \mathcal{F})$ is uniform (or uniform closed), we omit $\mathcal{F}$ and denote it as (shuf, Π). Since all easy-to-implement shuffles (e.g., random cuts, random bisection cuts, pile-shifting shuffles, and pile-scramble shuffles) in the literature are uniform

closed, it is considered to be the necessary condition for shuffles to be *practical*. Indeed, there are various methods [4,7,8,14] to implement uniform closed shuffles physically[5].

Following this formalization, the shuffling actions in steps 1 and 2 in the MKS protocol are denoted by $(\mathsf{shuf}, \{\mathsf{id}, (1\,3)(2\,4)\})$ and $(\mathsf{shuf}, \{\mathsf{id}, (2\,3)\})$, respectively, where id denotes the identity permutation and a probability distribution is omitted when it is uniform. In this formalization, one can observe that the two shuffling actions above can be described as a single action [11], as follows:

$$(\mathsf{shuf}, \{\mathsf{id}, (1\,3)(2\,4), (2\,3), (1\,3\,2\,4)\}), \tag{2}$$

because they are acted sequentially. However, this action is not considered practical because the permutation set $\{\mathsf{id}, (1\,3)(2\,4), (2\,3), (1\,3\,2\,4)\}$ does not form a group. In summary, the MKS protocol uses two practical shuffles, and it can be viewed as a protocol with a single non-closed shuffle.

A natural question is stated as: *Is it possible to construct a four-card AND protocol with a single practical (i.e., uniform closed) shuffle* (shuf, Π)*?*

1.3 Contributions

In this study, we present a negative answer to the above problem, which is summarized in the following theorem.

Theorem 1. *Any four-card non-committed format AND protocol cannot be realized using only one practical shuffle.*

This result implies that the two practical shuffles used in the MKS protocol [10] constitute the minimal requirement to perform a secure computation. To establish this, we enumerate all practical shuffles and show that any single shuffle either fails to compute the logical AND function correctly or leaks information about the inputs (and thus needs an additional shuffle as in the MKS protocol).

Our approach leverages a state tree proposed by Koch et al. [8], which formalizes the transition of the "actual" sequence of cards during the execution of a protocol, similar to the transition shown in Table 1. We enumerate all closed shuffles for a sequence of four cards and show that there are 30 subgroups in S_4, say $G_i \subseteq S_4$ for $1 \leq i \leq 30$, as shown later. Our impossibility proof using a state tree shows that any "output state" cannot be derived if we apply any uniform closed shuffle (shuf, G_i) to the "start state" only once. In the proof, we do not depict a state tree for every case of applying (shuf, G_i), but employ the subset and conjugacy relation among G_i to reduce the number of cases, allowing us to simplify the proof. Then we extend the impossibility proof to the case of $(\mathsf{shuf}, G_i, \mathcal{F}_i)$ for any probability distribution $\mathcal{F}_i$ on G_i, i.e., non-uniform closed shuffles.

[5] Koch et al. [8] showed that any uniform closed shuffle can be implemented by the use of private permutations, which apply a permutation covertly. Koch [7] showed that any uniform closed shuffle can be implemented from pile-shifting shuffles. Shinagawa et al. [14] showed that (possibly any) uniform closed shuffle can be implemented by cyclic shuffles, based on group-theoretic factorization [4].

Table 2: The minimum number of cards required for committed format AND protocols

	Cards	Finite Runtime	Uniform	Closed
Mizuki–Sone [12]	6	✓	✓	✓
Koch et al. [8]	4			✓
Koch et al. [8]	5	✓		
Abe et al. [1]	5		✓	✓
Koch [6] & R.–I. [13]	4		✓	
Koch [6] & R.–I. [13]	5	✓	✓	

1.4 Related Work

Both committed- and non-committed-format AND protocols have a long history of improvements. We summarize the history for committed format ones in Table 2. As introduced above, Koch et al. [8] classified shuffles based on two elements: uniform and closed. They then proved that no four-card AND protocol exists that terminates within a finite runtime regardless of shuffles used, meaning that a finite runtime AND protocol requires at least five cards. They also proposed two protocols: one is a four-card Las Vegas protocol using a closed shuffle, and the other is a five-card finite-runtime protocol using a non-uniform non-closed shuffle. Building on this perspective, Kastner et al. [5] focused on the shuffle classification and proved the following two results:

- No five-card finite-runtime AND protocol exists using only closed shuffles. This result implies that a six-card finite-runtime AND protocol proposed by Mizuki and Sone [12] with the use of a random bisection cut uses the minimal number of cards. This also implies that Koch et al.'s five-card finite-runtime protocol uses the minimal number of cards. However, the case of using only uniform shuffles remains unknown.
- No four-card AND protocol exists using only uniform and closed shuffles. This result implies that Koch et al.'s four-card Las Vegas protocol uses the minimal number of cards. However, the construction of a five-card AND protocol using only uniform closed shuffles remains unknown.

These open problems were soon positively resolved; In 2018, Abe et al. [1] proposed a five-card Las Vegas protocol using uniform closed shuffles. Regarding the other open problem concerning uniform shuffles, Koch [6] and Ruangwises and Itoh [13] independently devised both a four-card Las Vegas protocol and a five-card finite-runtime protocol using only uniform shuffles. These existing studies have clarified the trade-off between the type of shuffling actions and the number of cards required in commitment format AND protocols. More recently, protocols have been developed by restricting shuffle actions to only random cuts (cf. [2, 9, 16]). However, the minimal number of shuffles in (non-)committed format AND protocols remains unsolved.

2 Preliminaries

In this section, we introduce a computational model for card-based cryptographic protocols. We use this model to prove our result in a rigorous way.

2.1 Model of Card-Based Protocols

In this paper, we deal with the Mizuki–Shizuya model [11] as a model for card-based protocols. Let $\mathcal{D}$ be a non-empty finite multiset not containing the back symbol ?, and call it a *deck*. The number of elements in $\mathcal{D}$, $d = |\mathcal{D}|$, is called the number of cards.

For $c \in \mathcal{D}$, $\frac{c}{?}$ denotes a *face-up card* and $\frac{?}{c}$ denotes a *face-down card*. A face-up or face-down card is also called a *lying card*. We define $\mathsf{atom}(\frac{c}{?}) \coloneqq c$ and $\mathsf{atom}(\frac{?}{c}) \coloneqq c$ to denote an *atomic card*. The surface of the lying card is defined as $\mathsf{top}\left(\frac{c}{?}\right) \coloneqq c$ and $\mathsf{top}\left(\frac{?}{c}\right) \coloneqq ?$. A sequence of lying cards $\Gamma = (\alpha_1, \alpha_2, \cdots, \alpha_d)$ is called a *sequence* of the deck $\mathcal{D}$, if it satisfies $[\mathsf{atom}(\alpha_1), \mathsf{atom}(\alpha_2), \ldots, \mathsf{atom}(\alpha_d)] = \mathcal{D}$. Denote the set of all sequences in a deck $\mathcal{D}$ as $\mathsf{Seq}^{\mathcal{D}}$. We define the visible sequence $\mathsf{vis}(\Gamma)$ of the card sequence $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_d)$ as follow:

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

Denote the set of all visible sequences in a deck $\mathcal{D}$ as $\mathsf{Vis}^{\mathcal{D}} \coloneqq \{\mathsf{vis}(\Gamma) | \, \Gamma \in \mathsf{Seq}^{\mathcal{D}}\}$.

Let us define an action swap to be $\mathsf{swap}\left(\frac{c}{?}\right) \coloneqq \frac{?}{c}$ and $\mathsf{swap}\left(\frac{?}{c}\right) \coloneqq \frac{c}{?}$. For $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_d)$, the action of turning over the lying cards at the position specified by the set $T \subseteq \{1, 2, \ldots, d\}$ is called a *turn* action. The turn action $\mathsf{turn}_T(\cdot)$ is defined as follows:

$$\mathsf{turn}_T((\alpha_1, \alpha_2, \ldots, \alpha_d)) \coloneqq (\beta_1, \beta_2, \ldots, \beta_d)$$

such that

$$\beta_i = \begin{cases} \mathsf{swap}(\alpha_i) & \text{if } i \in T; \\ \alpha_i & \text{otherwise.} \end{cases}$$

For $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_d)$, the action of rearranging the lying cards based on the permutation $\pi \in S_d$ is called the *permutation* action. The permutation action $\mathsf{perm}_\pi(\cdot)$ is defined as follows:

$$\mathsf{perm}_\pi((\alpha_1, \alpha_2, \ldots, \alpha_d)) \coloneqq (\alpha_{\pi^{-1}(1)}, \alpha_{\pi^{-1}(2)}, \ldots, \alpha_{\pi^{-1}(d)}).$$

For $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_d)$, based on the the pair $(\Pi, \mathcal{F})$ (*permutation set* $\Pi \subseteq S_d$ and the probability distribution $\mathcal{F}$ on Π), the action of applying the permutation $\Pi \subseteq S_d$ chosen with probability distribution $\mathcal{F}$ for the lying cards is called *shuffle* action. The shuffle action $\mathsf{shuf}_{\Pi,\mathcal{F}}(\cdot)$ is defined as follows:

$$\mathsf{shuf}_{\Pi,\mathcal{F}}((\alpha_1, \alpha_2, \ldots, \alpha_d)) \coloneqq \mathsf{perm}_\pi((\alpha_1, \alpha_2, \ldots, \alpha_d)).$$

Here, $\pi \in \Pi$ is a permutation chosen randomly following $\mathcal{F}$. Then, we define SP^d as the entire set $(\Pi, \mathcal{F})$ of permutation sets Π and probability distributions $\mathcal{F}$ on Π.

A card-based protocol is defined by the quadruple $\mathcal{P} = (\mathcal{D}, U, Q, A)$:

- $\mathcal{D}$ is a deck;
- $U \subseteq \mathsf{Seq}^{\mathcal{D}}$ is an input set;
- Q is a state set containing initial state q_0 and final state q_f;
- $A \colon (Q \setminus \{q_f\}) \times \mathsf{Vis}^{\mathcal{D}} \to Q \times (2^{\{1,2,\ldots,d\}} \cup S_d \cup \mathsf{SP}^d)$ is an action function.

In the Mizuki–Kumamoto–Sone AND protocol [10], $\mathcal{D} = [\clubsuit, \clubsuit, \heartsuit, \heartsuit]$, and $d = 4$, $U = \left\{(\frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}), (\frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit}), (\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}), (\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit})\right\}$. The action function takes as input the current state q of the protocol and the visible sequence $\mathsf{vis}(\Gamma)$ and returns the next state q' and the next action. The next actions are the turn action turn, the permutation action perm, and the shuffle action shuf, each of which outputs as follows.

- $A(q, \mathsf{vis}(\Gamma)) = (q', T \in 2^{\{1,2,\ldots,d\}}) : \mathsf{turn}_T(\Gamma)$;
- $A(q, \mathsf{vis}(\Gamma)) = (q', \pi \in S_d) : \mathsf{perm}_\pi(\Gamma)$;
- $A(q, \mathsf{vis}(\Gamma)) = (q', (\Pi, \mathcal{F}) \in \mathsf{SP}^d) : \mathsf{shuf}_{\Pi,\mathcal{F}}(\Gamma)$.

For instance, the action function of the first step of the Mizuki–Kumamoto–Sone AND protocol is $A(q_0, (?, ?, ?, ?)) = (q_1, (\{\mathsf{id}, (1\,3)(2\,4)\}, \mathcal{F}))$ (but $\mathcal{F}$ is the uniform distribution) and $(\mathsf{shuf}, \{\mathsf{id}, (1\,3)(2\,4)\})$ is to be applied. Thus, the card-based protocol is to perform one of the following actions on lying cards from the initial state: turn action turn, permutation action perm, or shuffle action shuf, until the final state. In addition to the card-based protocol in this sense, for non-committed format protocols, there is a defined function that determines the output value from the entire history of the visible sequence from the initial state to the final state.

2.2 KWH-tree

A state tree proposed by Koch, Walzer, and Härtel [8] is called the *KWH-tree* and is useful to show a card-based protocol visually. Figure 1 depicts the KWH-tree for the MKS protocol [10].

Let us introduce how the KWH-tree works using Fig. 1. Each "box" from the top to the bottom is called a *state*, in which every possible sequence of symbols[6] (left) is mapped to a probability (right), where X_{ab} denotes a random variable for the input value being $(a, b) \in \{0, 1\}^2$, and each probability associated with a sequence of symbols represents a probability that the atomic sequence of the current sequence placed on the table is exactly that sequence. Note that $X_{11} + X_{10} + X_{01} + X_{00} = 1$. For example, the topmost state means that a probability of the initial sequence being ♡♣♡♣ is X_{11}, ♡♣♣♡ is X_{10}, ♣♡♡♣ is X_{01}, and ♣♡♣♡ is X_{00}. The transition of a state corresponds to the visible sequence trace in the protocol, and each state is transformed into its subsequent state(s) according to directed edge(s) annotated with a unique action that is specified in the protocol for that situation. Thus, permutation and shuffle actions always lead to a single state, but any turn action results in multiple states depending on the symbol appeared, i.e., ♡ or ♣.

More formally, a state for a four-card protocol is a map $\mu\colon S \to P$ where S is a set of all possible sequences of four symbols

$$S := \{♡♣♡♣, ♣♡♣♡, ♡♡♣♣, ♣♣♡♡, ♡♣♣♡, ♣♡♡♣\},$$

and P is a set of homogeneous polynomials of degree 1 with non-negative coefficients

$$P := \{p_{11}X_{11} + p_{10}X_{10} + p_{01}X_{01} + p_{00}X_{00} \mid p_{xy} \geq 0\}.$$

[6] Here, a sequence of symbols rather than cards is employed for simplicity.

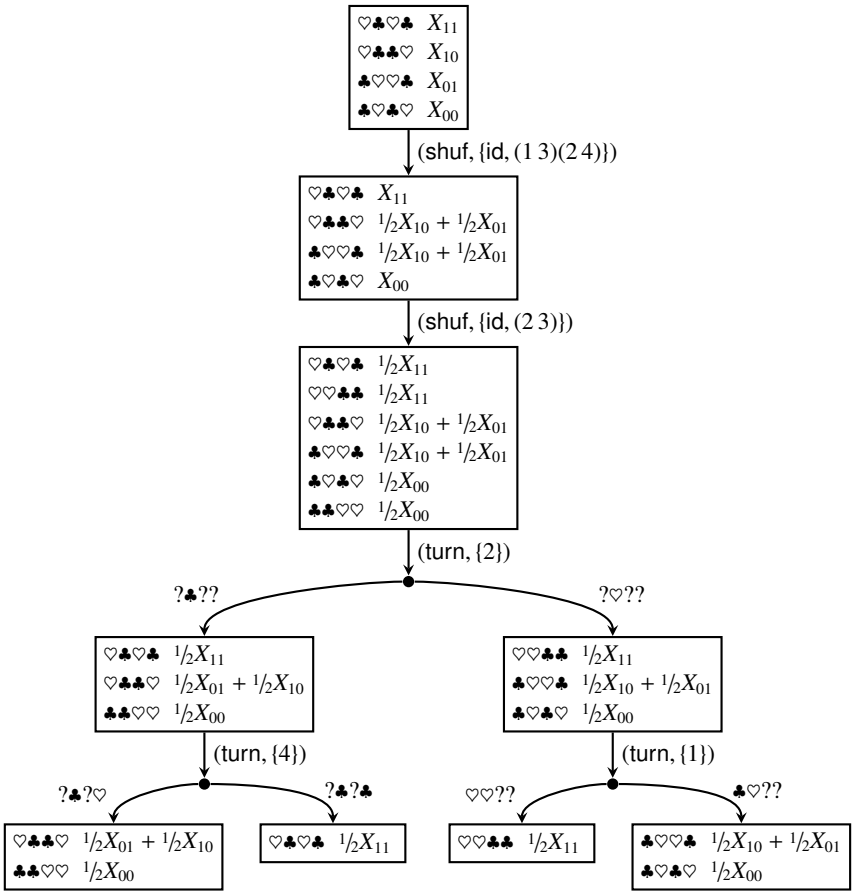

Fig. 1: The KWH-tree [8] for the Mizuki–Kumamoto–Sone AND protocol [10]

For example, the *start* state μ_s shown in Fig. 1 is a unique state and is described as follows:

$$\begin{aligned} \mu_s(\heartsuit\clubsuit\heartsuit\clubsuit) &= X_{11}, \\ \mu_s(\heartsuit\clubsuit\clubsuit\heartsuit) &= X_{10}, \\ \mu_s(\clubsuit\heartsuit\heartsuit\clubsuit) &= X_{01}, \\ \mu_s(\clubsuit\heartsuit\clubsuit\heartsuit) &= X_{00}. \end{aligned} \tag{3}$$

Note that $\mu_s(\clubsuit\clubsuit\heartsuit\heartsuit) = \mu_s(\heartsuit\heartsuit\clubsuit\clubsuit) = 0$. In particular, for a non-committed format AND protocol, the security of the protocol means that *when the output value is 0, it must be indistinguishable whether the input value is (0,0), (0,1), or (1,0)*. This implies that some non-negative real numbers $0 \leq p_0, p_1 \leq 1$ with either $p_0 \neq 0$ or $p_1 \neq 0$ exist for the state μ such that

$$\sum_{s \in S} \mu(s) = p_1 X_{11} + p_0(X_{00} + X_{10} + X_{01}). \tag{4}$$

Here, the part of the equation $p_0(X_{00}+X_{10}+X_{01})$ implies that each coefficient of X_{00}, X_{01}, and X_{10} is the same, i.e., the protocol leaks no information about whether the input value is $(0,0)$, $(0,1)$, or $(1,0)$. Note that Eq. (4) is the necessary and sufficient condition of the security; thus, every secure non-committed format protocol satisfies it in all states.[7]

2.3 Turnability for a State

Kastner et al. [5] introduced the *turnability* for a state μ, which defines the condition for not leaking information when applying a turn action to a state. Formally, for a non-committed format AND protocol $([\clubsuit, \clubsuit, \heartsuit, \heartsuit], U, Q, A)$, we say that μ is *turnable for* $i \in \{1,2,3,4\}$ if for any $c \in \{\heartsuit, \clubsuit\}$, there exists non-negative real numbers $0 \leq p_0, p_1 \leq 1$ with either $p_0 \neq 0$ or $p_1 \neq 0$ such that:

$$\sum_{s \in S, s[i]=c} \mu(s) = p_1 X_{11} + p_0(X_{00} + X_{10} + X_{01}), \tag{5}$$

where i denotes the position of a sequence s and $s[i]$ denotes the i-th symbol of s. In other words, a state μ is turnable for i if both of its subsequent states satisfy Eq. (4) when (turn, i) is applied to μ; a state μ is not turnable for i if (turn, i) leaks information about the input.

2.4 $\perp$-Sequence

As seen in Fig. 1, any non-committed format AND protocol finally reaches a state μ such that either $\sum_{s \in S} \mu(s) = p_1 X_{11}$ or $\sum_{s \in S} \mu(s) = p_0(X_{00} + X_{10} + X_{01})$ for some real numbers $0 < p_0, p_1 \leq 1$. Such a state is called a *final* state, and if a protocol reaches the former (resp. latter) state, it means that the output value is 1 (resp. 0). From this viewpoint, a non-committed format AND protocol can be regarded as a mechanism to split a variable X_{11} with X_{00}, X_{10}, and X_{01}. If a protocol violates this, i.e., there is a state μ and a sequence $s \in S$ such that

$$\mu(s) = \sum_{i,j} p_{ij} X_{ij} \quad (p_{11} > 0, p_{00} + p_{10} + p_{01} > 0), \tag{6}$$

then the protocol can no longer split the variable X_{11} with X_{00}, X_{10}, and X_{01}, making it impossible to terminate correctly. Such a sequence $s \in S$ satisfying Eq. (6) is called a $\perp$-*sequence* (for a state μ). Note that a correct protocol never has a $\perp$-sequence.

3 Impossibility Proof

In this section, we prove Theorem 1, i.e., it is impossible to construct a four-card AND protocol using only one practical (i.e., uniform closed) shuffle.

First, constructing an AND protocol without the use of any shuffle (i.e., the number of shuffles is zero) is impossible. This is because the start state is not turnable for any

[7] For a committed format AND protocol, Eq. (4) along with $p_0 = p_1$ is the necessary and sufficient condition of the security, i.e., the coefficient of X_{ab} is the same for all $(a,b) \in \{0,1\}^2$.

$i \in \{1,2,3,4\}$ according to Eq. (5), and a state μ obtained by just applying (perm, π) (for any $\pi \in S_4$) to the start state is also not turnable for any $i \in \{1,2,3,4\}$ because π is a bijective function. Thus, any AND protocol must have at least one shuffle.

Now we consider whether an AND protocol can be constructed by using one practical shuffle. We first claim that any protocol with having permutation actions can be converted into another protocol (computing the same function) without permutation actions as follows. If (perm, π) is applied *before* the shuffle action (shuf, G), this is equivalent to applying a uniform closed shuffle (shuf, G') such that $G' := \pi^{-1}G\pi$, i.e., G' is conjugate to G, and then applying (perm, π). If (perm, π) is applied *before* the turn action $(\mathsf{turn}, \{t_1, \ldots, t_k\})$, this is equivalent to applying just a turn action $(\mathsf{turn}, \{\pi^{-1}(t_1), \ldots, \pi^{-1}(t_k)\})$. Therefore, it suffices to consider the following protocol for an arbitrary group G and an arbitrary set $T \subseteq \{1,2,3,4\}$:

1. Apply a uniform closed shuffle (shuf, G).
2. Apply a turn action (turn, T).

As described in Sect. 1, we examine all 30 subgroups of the symmetric group S_4, say G_i for $1 \le i \le 30$ as listed in Table 3, and show that every state after (shuf, G_i) is applied to the start state either does not satisfy the turnability in Eq. (5) or produces a $\perp$-sequence in Eq. (6). Let μ'_i be a state after (shuf, G_i) is applied to the start state for $1 \le i \le 30$. Among these, μ'_1 can be immediately rejected, as it leads to a protocol equivalent to the case with no shuffling. In the next two subsections, we examine the remaining 29 groups as follows.

- Section 3.1 deals with the five groups of G_3, G_4, G_5, G_6, and G_9, where any $\perp$-sequence does not arise. Our approach shows that, for such groups, none of the five resulting states is turnable for any $i \in \{1,2,3,4\}$.
- Section 3.2 deals with the remaining groups, in which we show that a $\perp$-sequence appears in every resulting state.

3.1 Shuffles Not Producing $\perp$-Sequences

Let us consider the five shuffles using G_3, G_4, G_5, G_6, and G_9 listed in Table 3. First, note that G_5 and G_6 are symmetric to G_4 and G_3 with respect to the order of the two inputs, respectively, and hence, G_5 and G_6 yield the same results; the output of applying (shuf, G_3) to the two input commitments to a, b, is identical to that of applying (shuf, G_6) to the case when the two commitments are swapped beforehand, i.e., b, a, for instance. Therefore, we omit the cases for G_5 and G_6 and prove only the cases for G_3, G_4, and G_9.

Figure 2 depicts the KWH-trees when each of (shuf, G_3), (shuf, G_4), (shuf, G_9) is applied to the start state. Here, no $\perp$-sequence is produced, since there exists no $s \in S$ satisfying Eq. (6). However, according to Eq. (5), none of the three resulting states is turnable for any $i \in \{1,2,3,4\}$. More precisely, when (shuf, G_3) is applied to the start

Table 3: List of all the 30 subgroups of the symmetric group S_4

Group	Elements of group	Group	Elements of group
G_1	{id}	G_{11}	{id, (1 2 3), (1 3 2)}
G_2	{id, (1 2)}	G_{12}	{id, (1 2 4), (1 4 2)}
G_3	{id, (1 3)}	G_{13}	{id, (1 3 4), (1 4 3)}
G_4	{id, (1 4)}	G_{14}	{id, (2 3 4), (2 4 3)}
G_5	{id, (2 3)}	G_{15}	{id, (1 2)(3 4), (1 3 2 4), (1 4 2 3)}
G_6	{id, (2 4)}	G_{16}	{id, (1 3)(2 4), (1 2 3 4), (1 4 3 2)}
G_7	{id, (3 4)}	G_{17}	{id, (1 4)(2 3), (1 2 4 3), (1 3 4 2)}
G_8	{id, (1 2)(3 4)}	G_{18}	{id, (1 2), (3 4), (1 2 3 4)}
G_9	{id, (1 3)(2 4)}	G_{19}	{id, (1 3), (2 4), (1 3 2 4)}
G_{10}	{id, (1 4)(2 3)}	G_{20}	{id, (1 4), (2 3), (1 4 2 3)}

Group	Elements of group
G_{21}	{id, (1 2)(3 4), (1 3)(2 4), (1 4)(2 3)}
G_{22}	{id, (1 2 3), (1 3 2), (1 2), (1 3), (2 3)}
G_{23}	{id, (1 2 4), (1 4 2), (1 2), (1 4), (2 4)}
G_{24}	{id, (1 3 4), (1 4 3), (1 3), (1 4), (3 4)}
G_{25}	{id, (2 3 4), (2 4 3), (2 3), (2 4), (3 4)}
G_{26}	{id, (3 4), (1 2), (1 2)(3 4), (1 3)(2 4), (1 4)(2 3), (1 3 2 4), (1 4 2 3)}
G_{27}	{id, (2 3), (1 4), (1 2)(3 4), (1 3)(2 4), (1 4)(2 3), (1 2 3 4), (1 4 3 2)}
G_{28}	{id, (2 4), (1 3), (1 2)(3 4), (1 3)(2 4), (1 4)(2 3), (1 2 3 4), (1 4 3 2)}
G_{29}	All even permutations (i.e., G_{29} is the alternating group A_4)
G_{30}	All permutations (i.e., G_{30} is the symmetric group S_4)

state, the resulting state μ'_3 is not turnable for any i as shown below.

$$\sum_{s\in S, s[i]=\heartsuit} \mu'_3(s) = \begin{cases} X_{11} + 1/2(X_{01} + X_{10}) + 0(X_{00}) & \text{for } i = 1, \\ 0(X_{11}) + 1/2(X_{01} + X_{10}) + X_{00} & \text{for } i = 2, \\ X_{11} + 1/2(X_{01} + X_{10}) + 0(X_{00}) & \text{for } i = 3, \\ 0(X_{11}) + 1/2(X_{01} + X_{10}) + X_{00} & \text{for } i = 4. \end{cases}$$

Observe that in each equation, the coefficient of X_{00} differs from those of X_{01} and X_{10}; thus, it does not satisfy Eq. (5), indicating that μ'_3 is not turnable. Similarly, in both μ'_4 and μ'_9, we observe that the coefficient of X_{00} differ to those of X_{10} and X_{01} as shown in Fig. 2, i.e., they are not turnable. Therefore, we have proved the impossibility of constructing an AND protocol using G_3, G_4, G_5, G_6, and G_9.

3.2 Shuffles Producing ⊥-Sequences

We now verify the remaining cases, i.e., the shuffles other than the five shuffles discussed in the previous section, which produce a ⊥-sequence. First, we examine G_2 and G_7. Figure 3 depicts the KWH-tree for the case where (shuf, G_2) is applied to the start state. Here, we can observe that the sequence ♡♣♡♣ is a ⊥-sequence because it satisfies Eq. (6):

$$\mu'_2(\heartsuit\clubsuit\heartsuit\clubsuit) = {}^1\!/_2 X_{11} + {}^1\!/_2 X_{01}.$$

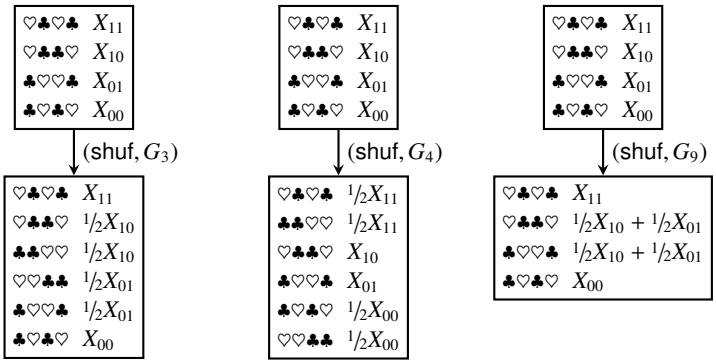

Fig. 2: The KWH-trees when each of (shuf, G_3), (shuf, G_4), and (shuf, G_9) is applied to the start state

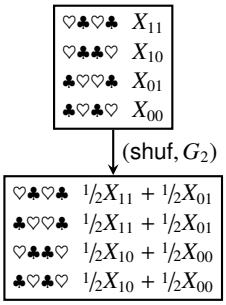

Fig. 3: The KWH-tree constructed with G_2

That is, $\mu'_2(\heartsuit\clubsuit\heartsuit\clubsuit)$ contains both X_{11} and X_{01}, i.e., both the sum of coefficients of X_{10}, X_{01}, and X_{00} and that of X_{11} are greater than 0. The same holds for G_7 since G_2 and G_7 are symmetric with respect to the order of the two inputs, as discussed in Sect. 3.1.

From Table 3, it can be seen that G_2 is a subgroup of G_{18}:

$$G_2 = \{\mathsf{id}, (1\,2)\} \subseteq G_{18} = \{\mathsf{id}, (1\,2), (3\,4), (1\,2\,3\,4)\}.$$

More generally, G_2 and G_7 are subgroups of the following groups:

$$G_2 = G_{18} \cap G_{22} \cap G_{23} \cap G_{26} \cap G_{30},$$
$$G_7 = G_{18} \cap G_{24} \cap G_{25} \cap G_{26} \cap G_{30}.$$

Since both G_2 and G_7 produce a $\perp$-sequence, as shown above, any group that contains them, i.e., G_{18}, G_{22}, G_{23}, G_{24}, G_{25}, G_{26}, and G_{30}, must also produce a $\perp$-sequence.

Let us move on to G_{11}, G_{12}, G_{13}, and G_{14}. Since G_{11} and G_{12} are symmetric to G_{14} and G_{13}, respectively, we consider G_{11} and G_{12}, as depicted in Fig. 4. Here, $\heartsuit\clubsuit\heartsuit\clubsuit$ is a $\perp$-sequence, i.e.,

$$\mu'_{11}(\heartsuit\clubsuit\heartsuit\clubsuit) = 1/3X_{11} + 1/3X_{01},$$
$$\mu'_{12}(\heartsuit\clubsuit\heartsuit\clubsuit) = 1/3X_{11} + 1/3X_{01}.$$

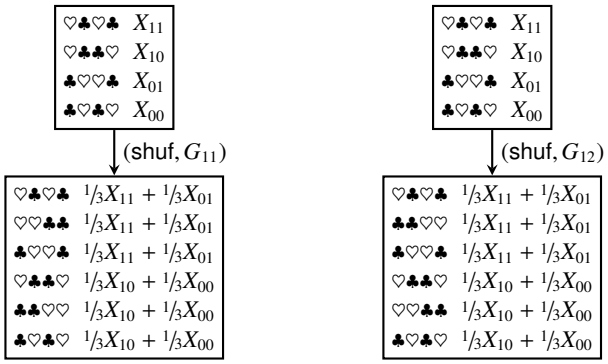

Fig. 4: The KWH-trees when each of (shuf, G_{11}) and (shuf, G_{12}) is applied to the start state

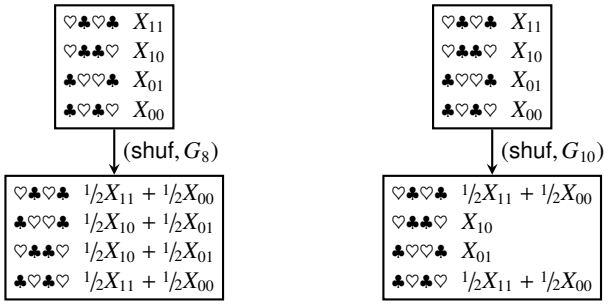

Fig. 5: The KWH-trees when each of (shuf, G_8) and (shuf, G_{10}) is applied to the start state

Therefore, this demonstrates the impossibility for G_{11} and G_{12}, and likewise for G_{13} and G_{14}.

Regarding G_8 and G_{10} shown in Fig. 5, both produce a $\perp$-sequence, as can be seen from the following equations:

$$\mu'_8(\heartsuit\clubsuit\heartsuit\clubsuit) = 1/2X_{11} + 1/2X_{00},$$
$$\mu'_{10}(\heartsuit\clubsuit\heartsuit\clubsuit) = 1/2X_{11} + 1/2X_{00}.$$

Moreover, G_8 and G_{10} are subgroups of the following groups:

$$G_8 = G_{15} \cap G_{21} \cap G_{27} \cap G_{28} \cap G_{29},$$
$$G_{10} = G_{17} \cap G_{21} \cap G_{27} \cap G_{28} \cap G_{29}.$$

Since both G_8 and G_{10} produce a $\perp$-sequence, it follows that these six groups, i.e., G_{15}, G_{17}, G_{21}, G_{27}, G_{28}, and G_{29}, also produce a $\perp$-sequence. Therefore, we have confirmed the impossibility for all of these eight groups.

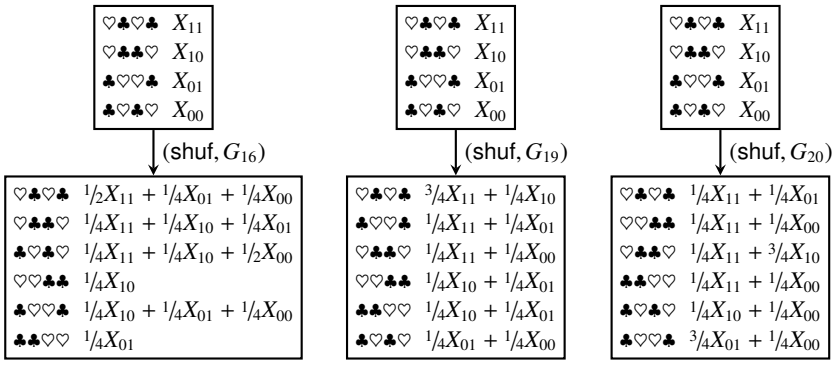

Fig. 6: The KWH-trees when each of (shuf, G_{16}), (shuf, G_{19}), and (shuf, G_{20}) is applied to the start state

Lastly, as shown in Fig. 6, the cases of G_{16}, G_{19}, and G_{20} also produce a $\perp$-sequence:

$$\mu'_{16}(\heartsuit\clubsuit\heartsuit\clubsuit) = \tfrac{1}{2}X_{11} + \tfrac{1}{4}X_{01} + \tfrac{1}{4}X_{00},$$
$$\mu'_{19}(\heartsuit\clubsuit\heartsuit\clubsuit) = \tfrac{3}{4}X_{11} + \tfrac{1}{4}X_{10},$$
$$\mu'_{20}(\heartsuit\clubsuit\heartsuit\clubsuit) = \tfrac{1}{4}X_{11} + \tfrac{1}{4}X_{00}.$$

Thus, we have confirmed the case of (shuf, G_i) for all i, which proves Theorem 1.

3.3 Extension to Closed Shuffles

The impossibility proof presented in the previous sections can be extended to the case of using a single *closed* shuffle instead of a single uniform closed shuffle. This result is formalized in the following theorem.

Theorem 2. *Any four-card non-committed format AND protocol cannot be realized using only a single non-uniform closed shuffle.*

Proof. The proof for non-uniform closed shuffles is similar to that for uniform closed shuffles, i.e., we show that after applying (shuf, $G_i, \mathcal{F}_i$) for each $1 \leq i \leq 30$ and any probability distribution $\mathcal{F}_i$ on G_i, the resulting state is either not turnable or contains a $\perp$-sequence. Here, a $\perp$-sequence arises independently of the distribution of $\mathcal{F}_i$ as shown in Sect. 3.2, and hence, we focus on the five groups of G_3, G_4, G_5, G_6, and G_9, as listed in Sect. 3.1. For any real number $p \in \mathbb{R}$ such that $0 < p < 1$, let $\mathcal{F}_3$ denote a probability distribution on G_3 such that:

$$\mathcal{F}_3 \colon \mathsf{id} \mapsto p,\ (1\,3) \mapsto 1 - p,$$

and $\mathcal{F}_4$ and $\mathcal{F}_9$ are defined analogously. Figure 7 depicts the KWH-tree when each of (shuf, $G_3, \mathcal{F}_3$), (shuf, $G_4, \mathcal{F}_4$), and (shuf, $G_9, \mathcal{F}_9$) is applied. Here, all of the three states are not clearly turnable according to Eq. (5), because the coefficient of X_{00} differs from those of X_{10} and X_{01} in every state. Since we can omit the cases for G_5 and G_6 due to the symmetry of the order of two inputs as discussed in Sect. 3.1, this proves the impossibility for the case of closed shuffles. □

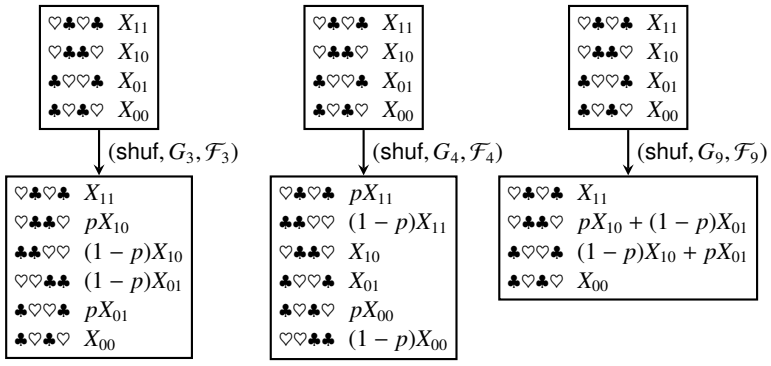

Fig. 7: The KWH-trees when each of non-uniform closed shuffles corresponding to G_3, G_4, and G_9 is applied to the start state, where p is a real number such that $0 < p < 1$.

4 Concluding Remarks

In this study, we proved the impossibility of constructing any four-card non-committed format AND protocol when only a single application of a uniform closed shuffle, i.e., practical shuffle, is allowed. The proof classifies all possible shuffle actions into two categories: those that generate a $\perp$-sequence and those that result in a state which is not turnable. Additionally, our proof employs the KWH-tree [8] to verify each case rigorously. Our result implies that the two-shuffle AND protocol proposed by Mizuki–Kumamoto–Sone [10] achieves the minimum number of shuffles.

Several problems regarding the minimal number of shuffles remain unsolved. We leave them as open problems, as follows.

- Six-card committed-format AND/XOR protocols using a random cut twice are known [2, 16]. Whether there exists such a protocol using only a single random cut remains open. Recall that there is no five-card finite-runtime AND protocol using only closed shuffles [5], though a protocol for computing XOR might be possible.
- There is a six-card non-committed-format protocol for computing the three-input majority function using each of a random cut and a random bisection cut (i.e., twice in total) [15]. Note that these two shuffles are acted sequentially, and hence, they can be described as a single action, as in the MKS protocol [10]. Whether there exists such a protocol using only a single uniform closed shuffle remains open.

Acknowledgments. We thank the anonymous referees, whose comments have helped us to improve the presentation of the paper. This work was supported in part by JSPS KAKENHI Grant Number JP23H00479 and by Institute of Mathematics for Industry, Joint Usage/Research Center in Kyushu University. (FY2023 Short-term Joint Research "Organizing open problems in card-based cryptography through industry-academia collaboration" (2023a020) and FY2024 Short-term Joint Research "Deepening and new frontiers in card-based cryptography through industry-academia collaboration and cooperation in the fields of mathematics and cryptography" (2024a035)).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Abe, Y., Hayashi, Y., Mizuki, T., Sone, H.: Five-card AND protocol in committed format using only practical shuffles. In: ACM ASIA Public-Key Cryptography Workshop. pp. 3–8. ACM, New York (2018), https://doi.org/10.1145/3197507.3197510
2. Abe, Y., Mizuki, T., Sone, H.: Committed-format AND protocol using only random cuts. Nat. Comput. **20**(4), 639–645 (2021), https://doi.org/10.1007/s11047-021-09862-2
3. 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
4. Kanai, K., Miyamoto, K., Nuida, K., Shinagawa, K.: Uniform cyclic group factorizations of finite groups. Communications in Algebra **52**(5), 2174–2184 (2024), https://doi.org/10.1080/00927872.2023.2285908
5. Kastner, J., Koch, A., Walzer, S., Miyahara, D., Hayashi, Y., Mizuki, T., Sone, H.: The minimum number of cards in practical card-based protocols. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology—ASIACRYPT 2017. LNCS, vol. 10626, pp. 126–155. Springer, Cham (2017), https://doi.org/10.1007/978-3-319-70700-6_5
6. Koch, A.: The landscape of optimal card-based protocols. Mathematical Cryptology **1**(2), 115–131 (2022), https://journals.flvc.org/mathcryptology/article/view/130529
7. Koch, A., Walzer, S.: Foundations for actively secure card-based cryptography. In: Farach-Colton, M., Prencipe, G., Uehara, R. (eds.) Fun with Algorithms. LIPIcs, vol. 157, pp. 17:1–17:23. Schloss Dagstuhl, Dagstuhl, Germany (2020), https://doi.org/10.4230/LIPIcs.FUN.2021.17
8. 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
9. Koyama, H., Toyoda, K., Miyahara, D., Mizuki, T.: New card-based copy protocols using only random cuts. In: ACM ASIA Public-Key Cryptography Workshop. pp. 13–22. ACM, NY (2021), https://doi.org/10.1145/3457338.3458297
10. 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
11. 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
12. Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: Deng, X., Hopcroft, J.E., Xue, J. (eds.) Frontiers in Algorithmics. LNCS, vol. 5598, pp. 358–369. Springer, Berlin, Heidelberg (2009), https://doi.org/10.1007/978-3-642-02270-8_36
13. Ruangwises, S., Itoh, T.: AND protocols using only uniform shuffles. In: van Bevern, R., Kucherov, G. (eds.) Computer Science–Theory and Applications. LNCS, vol. 11532, pp. 349–358. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-19955-5_30
14. Shinagawa, K., Kanai, K., Miyamoto, K., Nuida, K.: How to covertly and uniformly scramble the 15 puzzle and rubik's cube. In: Broder, A.Z., Tamir, T. (eds.) Fun with Algorithms. LIPIcs, vol. 291, pp. 30:1–30:15. Schloss Dagstuhl, Dagstuhl, Germany (2024), https://doi.org/10.4230/LIPIcs.FUN.2024.30

15. 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
16. Toyoda, K., Miyahara, D., Mizuki, T., Sone, H.: Six-card finite-runtime XOR protocol with only random cut. In: ACM ASIA Public-Key Cryptography Workshop. pp. 2–8. ACM, New York (2020), https://doi.org/10.1145/3384940.3388961