

Another Use of the Five-Card Trick: Card-Minimal Secure Three-Input Majority Function Evaluation*

Kodai Toyoda¹, Daiki Miyahara^{2,3}, and Takaaki Mizuki^{1,3}

¹ Tohoku University, Sendai, Japan
mizuki+lncs[atmark]tohoku.ac.jp




² The University of Electro-Communications, Tokyo, Japan

³ National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

Abstract. Starting from the five-card trick proposed by Den Boer (EU-ROCRYPT'89), many card-based protocols performing secure multiparty computations with a deck of physical cards have been devised. However, the five-card trick is considered to be still the most elegant, easy-to-understand and practical protocol, which enables two players to securely evaluate the AND value of their private inputs using five cards. In other words, for more than thirty years, in the research area of card-based cryptography, we have not discovered any protocols that are as simple and beautiful as the five-card trick.

In this study, making use of the five-card trick, we design a novel easy-to-understand protocol which securely evaluates the three-input majority function using six cards. That is, by applying a simple shuffle, we reduce a secure three-input majority computation to evaluating the AND value. By virtue of a direct application of the five-card trick, our proposed majority protocol is extremely simple enough for lay-people to execute. In addition, one advantage is that ordinary people such as high school students will be able to learn the concept of logical AND/OR operations and the majority function as well as their relationship through our majority protocol, providing a nice tool of pedagogical significance. Thus, we believe that our new protocol is no less practical and beautiful than the five-card trick.

1 Introduction

To perform cryptographic tasks such as secure multiparty computations, *card-based cryptography* uses a deck of physical cards such as black  and red cards  where their backs are all identical . Using these cards, Boolean values are typically represented as follows:

$$\img alt="black spade card" data-bbox="432 788 449 805"/>  = 0, \img alt="red heart card" data-bbox="505 788 522 805"/>  = 1. \tag{1}$$

* This paper appears in Proceedings of INDOCRYPT 2021. The final authenticated version is available online at https://doi.org/10.1007/978-3-030-92518-5_24.

When a one-bit value $x \in \{0, 1\}$ is encoded by two face-down cards according to the encoding rule (1) above, we call such a pair of cards a *commitment* to x , which is denoted by

$$\underbrace{\boxed{?}\boxed{?}}_x.$$

This paper begins with introducing the first card-based protocol in history, called the *five-card trick*, designed by Den Boer [6].

1.1 The Five-Card Trick

Assume that Alice and Bob hold private input bits $x \in \{0, 1\}$ and $y \in \{0, 1\}$, respectively, and that each of them creates a commitment to his/her private bit. The five-card trick securely evaluates the AND value $x \wedge y$ of their private bits, given commitments to bits $x, y \in \{0, 1\}$ along with an additional card \heartsuit :

$$\heartsuit \underbrace{\boxed{?}\boxed{?}}_x \underbrace{\boxed{?}\boxed{?}}_y \rightarrow \cdots \rightarrow x \wedge y.$$

The procedure is as follows. (Here, we slightly rearrange the order of cards from the original reference [6] for the sake of later explanation.)

1. By swapping the two cards constituting the commitment to y (which means the NOT computation), obtain a commitment to the negation \bar{y} , and turn over the additional card:

$$\heartsuit \underbrace{\boxed{?}\boxed{?}}_x \underbrace{\boxed{?}\boxed{?}}_y \rightarrow \underbrace{\boxed{?}\boxed{?}}_{\heartsuit} \underbrace{\boxed{?}\boxed{?}}_x \underbrace{\boxed{?}\boxed{?}}_{\bar{y}}.$$

Note that the three \heartsuit s are cyclically consecutive (i.e., the first, second, and fifth cards are red) if and only if $x \wedge y = 1$:

$$\begin{aligned} \underbrace{\boxed{?}\boxed{?}\boxed{?}\boxed{?}}_{\heartsuit \clubsuit \heartsuit \clubsuit} & \text{ if } (x, y) = (0, 0), \\ \underbrace{\boxed{?}\boxed{?}\boxed{?}\boxed{?}}_{\heartsuit \clubsuit \heartsuit \heartsuit} & \text{ if } (x, y) = (0, 1), \\ \underbrace{\boxed{?}\boxed{?}\boxed{?}\boxed{?}}_{\heartsuit \heartsuit \clubsuit \heartsuit \clubsuit} & \text{ if } (x, y) = (1, 0), \\ \underbrace{\boxed{?}\boxed{?}\boxed{?}\boxed{?}}_{\heartsuit \heartsuit \clubsuit \clubsuit \heartsuit} & \text{ if } (x, y) = (1, 1). \end{aligned}$$

2. Apply a *random cut* to the sequence of five cards; a random cut, denoted by $\langle \cdot \rangle$, is a shuffling operation that cyclically shifts a sequence of cards at random without changing its order:

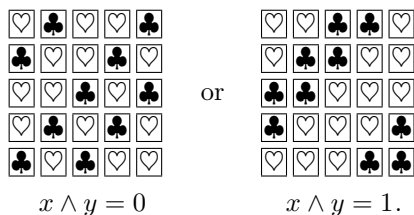
$$\langle \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?} \rangle \rightarrow \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}.$$

Thus, the resulting sequence becomes one of the following five cases with the equal probability (i.e., $1/5$), where we attach a number to each card for convenience sake:

$$\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array} \rightarrow \left\{ \begin{array}{l} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array}, \\ \begin{array}{ccccc} 2 & 3 & 4 & 5 & 1 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array}, \\ \begin{array}{ccccc} 3 & 4 & 5 & 1 & 2 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array}, \\ \begin{array}{ccccc} 4 & 5 & 1 & 2 & 3 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array}, \\ \begin{array}{ccccc} 5 & 1 & 2 & 3 & 4 \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array} \end{array} \right.$$

It is well-known that humans can securely implement a random cut easily so that nobody learns which case occurs [24].

3. Reveal all the five cards; then, we know the value of $x \wedge y$ depending on whether the three \heartsuit s are cyclically consecutive:



This is the five-card trick, whereby Alice and Bob can learn only the value of $x \wedge y$ without leaking any information about x and y more than necessary. As seen above, the five-card trick is extremely easy-to-understand: even lay-people can easily understand why this AND protocol works, and it is simple enough for non-experts such as high school students to execute without any difficulty.

1.2 Our Target

Since the invention of the five-card trick, many card-based protocols have been devised; refer to [9,14,23] for surveys. However, the five-card trick is considered to be still the most elegant, easy-to-understand, and practical protocol. The five-card trick provides a way of “secure dating” for two people [12] as well as it can be used for teaching the concept of secure multiparty computation to students [20,22].

While it is quite certain that the five-card trick is beautiful, many lay-people would consider that such a secure two-input AND computation may be somewhat useless because once Alice having a private bit of $x = 1$ knows that the result of the computation is $x \wedge y = 0$, she gets to know that Bob’s bit is $y = 0$, implying that information about the other private input is leaked. Of course, this is inherent, but it would be worthwhile to find more suitable functions for educational purpose.

In this study, as a promising function, we consider the three-input majority function $\text{maj} : \{0, 1\}^3 \rightarrow \{0, 1\}$ defined as

$$\text{maj}(a, b, c) = \begin{cases} 1 & \text{if } a + b + c \geq 2, \\ 0 & \text{if } a + b + c \leq 1. \end{cases} \quad (2)$$

That is, we solicit card-based protocols that securely evaluate $\text{maj}(a, b, c)$; we call them *three-input majority protocols*. By using such a three-input majority protocol, Alice, Bob, and Carol can know only whether or not there are two or more players having ‘Yes.’ We believe that the three-input majority function is more convincing than the two-input AND function when teaching the concept of secure multiparty computation.

1.3 The Existing Protocols

There are several existing three-input majority protocols. Here, we review the history.

In 2013, Nishida et al. [17] presented for the first time a three-input majority protocol using eight cards. Their protocol takes commitments to bits $a, b, c \in \{0, 1\}$ along with two additional cards and outputs a commitment to the value of $\text{maj}(a, b, c)$ using two shuffles:

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_a \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?}}_b \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?}}_c \clubsuit \heartsuit \rightarrow \dots \rightarrow \underbrace{\boxed{?} \boxed{?}}_{\text{maj}(a,b,c)} .$$

Such a protocol is called a *committed-format* protocol (because it outputs a commitment). It has been an open problem to reduce the number of required additional cards (to 0 or 1).

In 2017, Nakai et al. [16] showed that the three-input majority function can be securely evaluated with four cards by introducing “private operations.” Allowing private operations is a strong assumption⁴ that a player may manipulate the cards privately, say behind their back, while making sure that the other players do not see the movement (cf. private PEZ protocols [1,5]). The protocol proposed by Nakai et al. includes a private rearranging operation and a private turning operation.

In 2018, Watanabe et al. [25] showed that the same task can be conducted with only three cards. Their protocol uses an oral response to reverse the value depending on the input value. In other words, this protocol uses a private negation.

In 2020, Yasunaga [26] proposed a protocol using six cards based on simple private operations. This protocol needs to reconstruct the input commitment to a in the middle. Yasunaga [26] also provided a protocol that does not use private operation by making a copy of the input commitment to a in advance, resulting in an eight-card protocol.

⁴ Malicious behaviors during the private operations have been discussed in [2,11,18,19].

Table 1. The existing three-input majority protocols (without private operations) and our proposed protocols

	#Cards	#Shuf.	Committed format	Runtime
Nishida et al., 2013 [17]	8	2	✓	finite
Yasunaga, 2020 [26]	8	3		finite
Ours (Section 2)	6	2		finite
Ours (Appendix A)	6	8 (exp.)	✓	Las Vegas

This paper focuses on protocols that do not rely on any private operation; hence, in our setting, all the previous works are the protocol by Nishida et al. [17] and the second protocol by Yasunaga [26], as shown in Table 1. The open problem mentioned above has not been resolved yet; we will close it by proposing “card-minimal” protocols, as explained below.

1.4 Our Contribution

In this paper, we construct two protocols using six cards that improve upon the existing three-input majority protocol proposed by Nishida et al. [16], as shown in Table 1.

Our first protocol is the main contribution of this paper: it securely evaluates $\text{maj}(a, b, c)$ without any additional card:

$$\underbrace{\boxed{?}\boxed{?}}_a \underbrace{\boxed{?}\boxed{?}}_b \underbrace{\boxed{?}\boxed{?}}_c \rightarrow \dots \rightarrow \text{maj}(a, b, c).$$

Because any three-input majority protocol requires six cards due to three input commitments (in our setting) and our protocol needs no additional card, i.e., only three input commitments (consisting of six cards) suffice, our protocol is optimal in terms of the number of required cards under the encoding rules (1), i.e., it is *card-minimal*. Therefore, this is the first card-minimal protocol for the three-input majority function $\text{maj}(a, b, c)$.

As seen later, our protocol makes use of the five-card trick [6]. That is, by applying a simple shuffle, we reduce the three-input majority computation to evaluating the AND value. Thus, our card-minimal protocol uses only a simple shuffle along with an execution of the five-card trick. As the five-card trick is famous for its brevity, our protocol is also simple enough for lay-people to execute.

As the second protocol, we will also provide a committed-format version, which we will present in Appendix A:

$$\underbrace{\boxed{?}\boxed{?}}_a \underbrace{\boxed{?}\boxed{?}}_b \underbrace{\boxed{?}\boxed{?}}_c \rightarrow \dots \rightarrow \underbrace{\boxed{?}\boxed{?}}_{\text{maj}(a,b,c)}.$$

This protocol is obtained by amending our first protocol using the idea behind the AND protocol proposed by Abe et al. [3]. Although the runtime of the

protocol is Las Vegas, it is interesting to note that we can achieve the minimum number of cards without complex shuffling operations such as those required for the card-minimal AND protocol proposed by Koch et al. [10] and the one modified by Ruangwises and Itoh [21].

1.5 Outline

The outline of this paper is as follows. In Section 2, we present a simple and easy-to-understand three-input majority protocol which is card-minimal. In Section 3, we formally describe our protocol and give a formal proof for correctness and security. In Appendix A, we present a committed-format version of a card-minimal three-input majority protocol. We conclude this paper in Section 4.

2 Our Card-Minimal Majority Protocol

In this section, we design a three-input majority protocol without any additional card.

In our construction, we make use of the following simple fact on the three-input majority function $\text{maj}(a, b, c)$ [16]:

$$\text{maj}(a, b, c) = \begin{cases} b \wedge c & \text{if } a = 0, \\ b \vee c & \text{if } a = 1. \end{cases} \quad (3)$$

That is, we will reduce the computation of $\text{maj}(a, b, c)$ to the computation of $b \wedge c$ or $b \vee c$. As we can compute $b \wedge c$ using the five-card trick shown in Section 1.1, let us consider how to compute $b \vee c$ in a similar manner, i.e., we first propose the five-card “OR” protocol by modifying the five-card trick slightly. We also describe variants of the five-card trick and the five-card OR protocol by replacing the additional red card \heartsuit with a black card \clubsuit .

2.1 Variants of Five-Card Trick

In this subsection, based on the idea behind the five-card trick, we describe its four variants: the \heartsuit -based AND, \heartsuit -based OR, \clubsuit -based AND, and \clubsuit -based OR protocols.

The \heartsuit -based AND protocol is exactly the five-card trick itself because it uses \heartsuit as an additional card and evaluates the AND value. We can obtain the \heartsuit -based OR protocol by modifying the rearrangements in the five-card trick. As for the \clubsuit -based protocols, let the output be 0 if three \clubsuit s are cyclically consecutive; otherwise, let the output be 1. (Note that this encoding is the opposite case where \heartsuit is the additional card.) Then, we can have the \clubsuit -based AND and OR protocols, as seen below. For the sake of illustration, let us write commitments to x and y using x_0, x_1, y_0 and y_1 as

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{x_0 \ x_1 \ y_0 \ y_1},$$

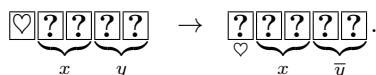
$$\underbrace{\hspace{1.5cm}}_x \quad \underbrace{\hspace{1.5cm}}_y$$

where x_0 and x_1 (y_0 and y_1) represent the two cards constituting the commitment to x (y). For example, if $x = 1$, x_0 is \heartsuit and x_1 is \clubsuit .

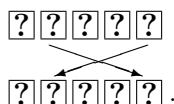
The \heartsuit -based AND protocol. This is exactly the same as the five-card trick presented in Section 1.1.

The \heartsuit -based OR protocol.

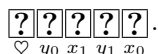
1. Arrange the five cards as follows:



2. Rearrange the order of the sequence as

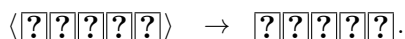


The resulting sequence of cards becomes



Note that the three \heartsuit s are cyclically consecutive if and only if $x \vee y = 1$.

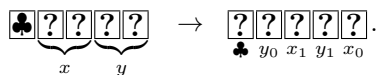
3. Apply a random cut to the sequence of five cards:



4. Reveal all the five cards. If the three red cards are cyclically consecutive $\heartsuit\heartsuit\heartsuit$, we have $x \vee y = 1$; otherwise, we have $x \vee y = 0$.

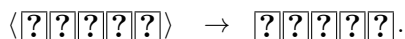
The \clubsuit -based AND protocol.

1. Perform Steps 1 and 2 of the \heartsuit -based OR protocol where the additional card is \clubsuit instead of \heartsuit . The resulting sequence of cards becomes



Note that the three \clubsuit s are *not* cyclically consecutive if and only if $x \wedge y = 1$.

2. Apply a random cut to the sequence of five cards:



3. Reveal all the five cards. If the three black cards are cyclically consecutive $\clubsuit\clubsuit\clubsuit$, $x \wedge y = 0$; otherwise, $x \wedge y = 1$.

The ♣-based OR protocol.

1. Perform Step 1 of the ♥-based AND protocol (namely, the five-card trick) where the additional card is ♣ instead of ♥. The resulting sequence of cards becomes

$$\begin{array}{c} \boxed{\clubsuit} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \\ \underbrace{\hspace{1.5cm}}_x \quad \underbrace{\hspace{1.5cm}}_y \end{array} \rightarrow \begin{array}{c} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \\ \underbrace{\hspace{1.5cm}}_x \quad \underbrace{\hspace{1.5cm}}_{\bar{y}} \end{array}$$

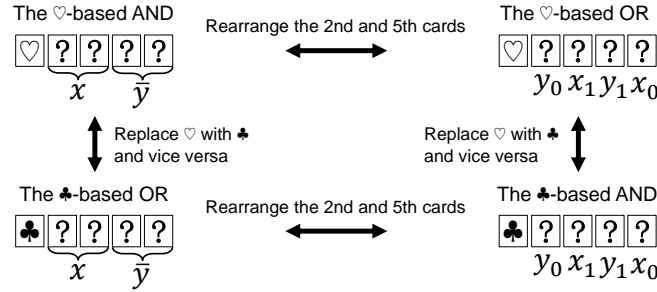
Note that the three ♣s are *not* cyclically consecutive if and only if $x \vee y = 1$.

2. Apply a random cut to the sequence of five cards:

$$\langle \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \rangle \rightarrow \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}$$

3. Reveal all the five cards. If the three black cards are cyclically consecutive $\boxed{\clubsuit} \boxed{\clubsuit} \boxed{\clubsuit}$, we have $x \vee y = 0$; otherwise, we have $x \vee y = 1$.

These four protocols satisfy the following relationship:



2.2 Idea

Here, we explain the idea behind our card-minimal three-input majority protocol. Consider an initial state

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_a \quad \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_b \quad \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{\bar{c}}, \tag{4}$$

where we have a commitment to \bar{c} by applying the NOT computation to a commitment to c . Let us apply a random cut to the second through sixth cards in this sequence; since the ♥-based AND will be applied (to the commitments to b and c) if $a = 0$ and the ♣-based OR will be applied if $a = 1$, we can derive the value of $\text{maj}(a, b, c)$ because of the fact (3). However, depending on whether the output is ♥-based or ♣-based, the value of a will be leaked.

To resolve this issue, using the aforementioned relationship between the four protocols, we “randomize” the state so that if $a = 0$, either the ♥-based AND or the ♣-based AND is applied with the equal probability, as follows. (Automatically, if $a = 1$, either the ♣-based OR or the ♥-based OR is applied in the same way.)

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_a \quad \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_b \quad \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{\bar{c}} \rightarrow \begin{cases} \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_a & \text{with prob. } \frac{1}{2}, \\ \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{\bar{a}} \quad \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{c_0 \ b_1 \ c_1 \ b_0} & \text{with prob. } \frac{1}{2}. \end{cases} \tag{5}$$

It is easy to see that if we apply a random cut to the second through sixth cards in the above randomized state in (5), then either the ♡- or ♣-based AND is applied when $a = 0$ (with the equal probability) and either the ♡- or ♣-based OR is applied when $a = 1$; hence, the value of a cannot be leaked while the value of $\text{maj}(a, b, c)$ can be derived.

To realize the randomization in (5), we introduce a practical shuffle, called a *random bisection cut*, invented by Mizuki and Sone [15]. It bisects a sequence of cards and swaps the two halves at random (denoted by $[\cdot|\cdot]$) as follows:

$$[\overset{1}{\boxed{?|?}} | \overset{2}{\boxed{?|?}}] \rightarrow \begin{cases} \overset{1}{\boxed{?|?}} \overset{2}{\boxed{?|?}} & \text{with prob. } \frac{1}{2}, \\ \overset{2}{\boxed{?|?}} \overset{1}{\boxed{?|?}} & \text{with prob. } \frac{1}{2}. \end{cases}$$

We will apply a random bisection cut to the commitment to a , the third card, and the sixth card, as seen later.

A random bisection cut can be securely implemented using familiar tools; a few implementations (that can be conducted publicly) were shown in [24]. If the backs of cards are asymmetric, a random bisection cut can be reduced to applying a random cut without using any auxiliary tools [24].

2.3 Description

Here, we present the complete description of our three-input majority protocol. This protocol starts with six cards of commitments to input bits a , b , and c .

1. Given commitments to a , b , and c , take the negation of the commitment to c :

$$\underbrace{\boxed{?|?}}_a \underbrace{\boxed{?|?}}_b \underbrace{\boxed{?|?}}_c \rightarrow \underbrace{\boxed{?|?}}_a \underbrace{\boxed{?|?}}_b \underbrace{\boxed{?|?}}_{\bar{c}}.$$

2. Apply a random bisection cut to the commitment to a , the third card, and the sixth card, and return the four cards, as follows:

$$\begin{aligned} & \overset{1}{\boxed{?|?}} \overset{2}{\boxed{?|?}} \overset{3}{\boxed{?|?}} \overset{6}{\boxed{?|?}} \rightarrow [\overset{1}{\boxed{?|?}} \overset{3}{\boxed{?|?}} | \overset{2}{\boxed{?|?}} \overset{6}{\boxed{?|?}}] \overset{4}{\boxed{?|?}} \\ \rightarrow & \overset{1}{\boxed{?|?}} \overset{2}{\boxed{?|?}} \overset{3}{\boxed{?|?}} \overset{4}{\boxed{?|?}} \overset{5}{\boxed{?|?}} \rightarrow \overset{1}{\boxed{?|?}} \overset{3}{\boxed{?|?}} \overset{2}{\boxed{?|?}} \overset{4}{\boxed{?|?}}. \end{aligned}$$

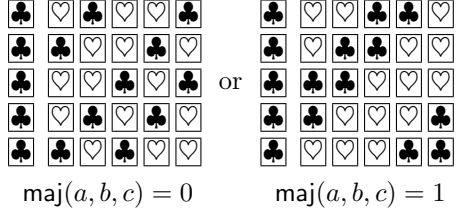
Note that the state of the resulting sequence becomes what is in (5).

3. Apply a random cut to the second to sixth cards as follows:

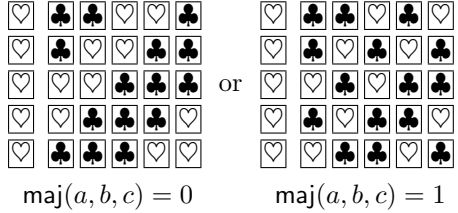
$$\overset{1}{\boxed{?}} \langle \overset{2}{\boxed{?|?|?|?|?}} \rangle \rightarrow \overset{1}{\boxed{?|?|?|?|?}}.$$

4. Reveal the first card.
 - (a) If ♣ appears, the result is ♡-based. Reveal all the remaining five cards. If the three red cards are cyclically consecutive $\heartsuit\heartsuit\heartsuit$, we have $\text{maj}(a, b, c) =$

1; otherwise, we have $\text{maj}(a, b, c) = 0$.



(b) If \heartsuit appears, the result is \clubsuit -based. Reveal all the remaining five cards. If the three black cards are cyclically consecutive $\clubsuit\clubsuit\clubsuit$, we have $\text{maj}(a, b, c) = 0$; otherwise, we have $\text{maj}(a, b, c) = 1$.



Thus, this protocol does not need any additional card, and uses only two shuffles. It is easy-to-understand as well as easy-to-implement. We will provide a committed-format version as well in Appendix A.

3 Formal Treatment

In this section, we give a description of our majority protocol (presented in Section 2) in a formal way based on the computation model of card-based cryptography [13]. We also prove the correctness and security of our protocol by using the KWH-tree invented by Koch et al. [10].

3.1 Operations in Card-Based Cryptography

In card-based cryptography, there are three main operations performed on a sequence of cards, namely, permuting, turning, and shuffling. Below, we assume a sequence of n cards.

Permute. This is denoted by (perm, π) where π is a permutation applied to the sequence of cards as follows:

$$\overset{1}{\boxed{?}} \overset{2}{\boxed{?}} \dots \overset{n}{\boxed{?}} \xrightarrow{(\text{perm}, \pi)} \overset{\pi^{-1}(1)}{\boxed{?}} \overset{\pi^{-1}(2)}{\boxed{?}} \dots \overset{\pi^{-1}(n)}{\boxed{?}} .$$

Turn. This is denoted by (turn, T) where T is a set of indexes, indicating that the t -th card is turned over for every $t \in T$ as follows:

$$\overset{1}{\boxed{?}} \overset{2}{\boxed{?}} \dots \overset{t}{\boxed{?}} \dots \overset{n}{\boxed{?}} \xrightarrow{(\text{turn}, T)} \overset{1}{\boxed{?}} \overset{2}{\boxed{?}} \dots \overset{t \in T}{\clubsuit} \dots \overset{n}{\boxed{?}} .$$

Algorithm 1 Our majority protocol

input set:

$$\left\{ \left(\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit} \right), \right. \\ \left. \left(\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \right. \\ \left. \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right) \right\}$$

1. (perm, (5 6))
 2. (shuf, {id, (1 2)(3 6)})
 3. (shuf, RC_{2,3,4,5,6})
 4. (turn, {1})
 5. **if** visible sequence = (♥, ?, ?, ?, ?, ?) **then**
 6. (result, 2, 3, 4, 5, 6)
 7. **else if** visible sequence = (♣, ?, ?, ?, ?, ?) **then**
 8. (result, 2, 3, 4, 5, 6)
-

Shuffle. This is denoted by (shuf, Π , \mathcal{F}) where Π is a permutation set and \mathcal{F} is a probability distribution on Π , indicating that $\pi \in \Pi$ is drawn according to \mathcal{F} and is applied to the sequence of cards as follows:

$$\frac{1}{?} \frac{2}{?} \dots \frac{n}{?} \xrightarrow{(\text{shuf}, \Pi, \mathcal{F})} \frac{\pi^{-1}(1)}{?} \frac{\pi^{-1}(2)}{?} \dots \frac{\pi^{-1}(n)}{?} .$$

We note that nobody knows which permutation in Π was applied. If the probability distribution \mathcal{F} is uniform, we may omit it.

3.2 Pseudocode

A pseudocode for our majority protocol is depicted in Algorithm 1, where we define

$$\text{RC}_{2,3,4,5,6} := \{\text{id}, (2\ 3\ 4\ 5\ 6), (2\ 3\ 4\ 5\ 6)^2, (2\ 3\ 4\ 5\ 6)^3, (2\ 3\ 4\ 5\ 6)^4\},$$

and (result, i, j, k, l, m) specifies output positions. The shuffle (shuf, RC_{2,3,4,5,6}) means that a random cut is applied to the second through sixth cards. The shuffle (shuf, {id, (1 2)(3 6)}) in Algorithm 1 represents the application of a random bisection cut in Step 2 shown in Section 2.3.

3.3 Correctness and Security

In this subsection, we verify the correctness and security of our non-committed-format majority protocol. A three-input majority protocol is said to be *correct* if, given input commitments to a, b and c , it always evaluates the value of $\text{maj}(a, b, c)$

correctly. We say that such a protocol is *secure* if it leaks no information beyond the value of $\text{maj}(a, b, c)$ for any run of the protocol.

To verify that our majority protocol is correct and secure, we make use of the *KWH-tree*, which is an excellent tool developed by Koch, Walzer, and Härtel [10]. That is, if one is able to write a KWH-tree satisfying some properties for a protocol, then it automatically implies that the protocol is correct and secure; see [8,9,10,14] for the details.

We describe the KWH-tree of our non-committed-format majority protocol in Figure 1, following Chapter 7 of [9]. The first box in Figure 1 corresponds to an initial sequence, consisting of three input commitments; X_{111} , X_{110} , X_{101} , X_{100} , X_{011} , X_{010} , X_{001} , and X_{000} represent the probabilities of $(a, b, c) = (1, 1, 1)$, $(a, b, c) = (1, 1, 0)$, $(a, b, c) = (1, 0, 1)$, $(a, b, c) = (1, 0, 0)$, $(a, b, c) = (0, 1, 1)$, $(a, b, c) = (0, 1, 0)$, $(a, b, c) = (0, 0, 1)$, and $(a, b, c) = (0, 0, 0)$, respectively. In the bottom boxes, we write X_1 instead of $X_{111} + X_{110} + X_{101} + X_{011}$ and write X_0 instead of $X_{100} + X_{010} + X_{001} + X_{000}$. A polynomial annotating a card sequence in a state, such as $1/2X_{111}$, represents the conditional probability that the current sequence is the one next to the polynomial, given the visible sequence trace observed so far on the table. From the two boxes at the bottom, one can see that $(\text{turn}, \{2, 3, 4, 5, 6\})$ reveals the value of $\text{maj}(a, b, c)$ definitively. Furthermore, in each box, the sum of all polynomials is equal to $X_{111} + X_{110} + X_{101} + X_{100} + X_{011} + X_{010} + X_{001} + X_{000}$, implying that no information about a, b and c leaks, i.e., the inputs and visible sequence trace are stochastically independent (before $(\text{turn}, \{2, 3, 4, 5, 6\})$ is applied finally).

Thus, the KWH-tree in Figure 1 guarantees that our proposed non-committed-format majority protocol is correct and secure.

4 Conclusion

In this paper, we constructed a three-input majority protocol using only six cards without depending on private operations. Therefore, this is the first card-minimal protocol for the three-input majority function $\text{maj}(a, b, c)$. We also show that we can obtain a committed-format majority protocol with the minimum number of cards.

The former protocol is so simple that lay-people can easily execute it; see the pseudocode presented in Section 3.2 again to recall that the protocol is quite simple. Thus, we believe that our three-input majority protocol is no less practical and beautiful than the five-card trick. We even think that our majority protocol is better than the five-card trick in a sense: lay-people will be able to learn the concept of logical AND and OR operations and their relationship through our majority protocol (because our protocol is based on the nice property that $\text{maj}(a, b, c)$ can be expressed simply using $b \wedge c$ and $b \vee c$, i.e., $\text{maj}(a, b, c)$ is equal to one of the four variants of the five-card trick according to the value of c) while the five-card trick is just based on the fact that the three red cards are consecutive only when $x = y = 1$.

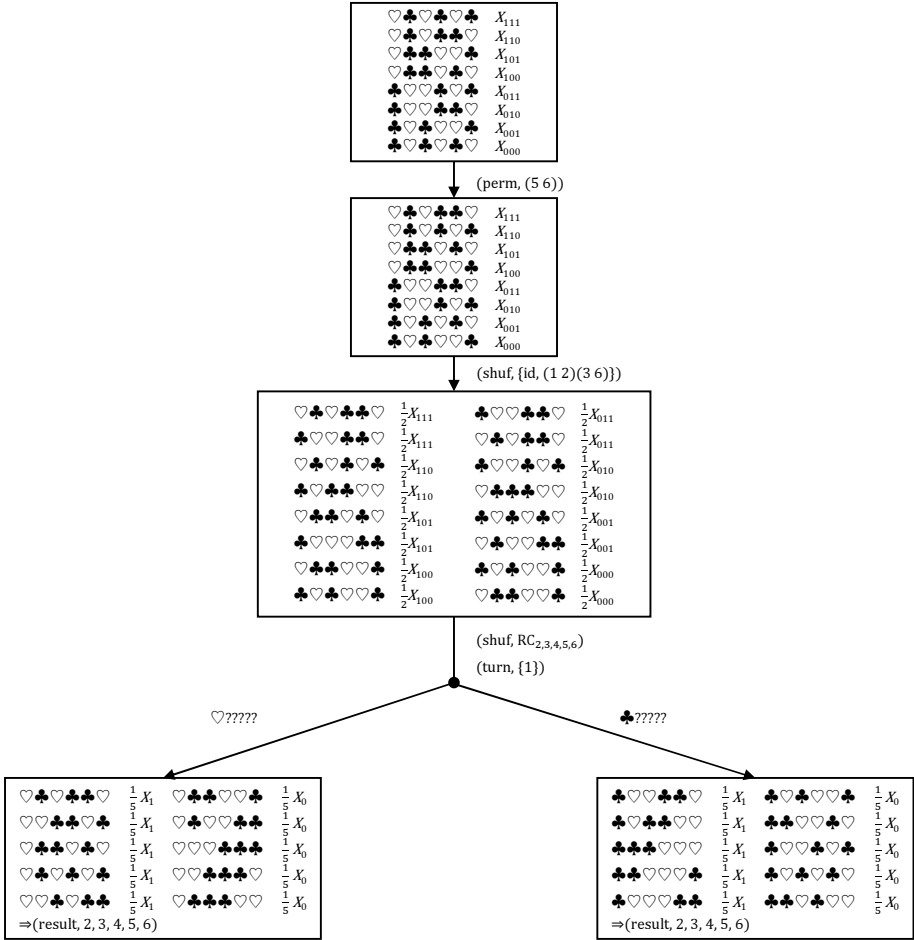


Fig. 1. The KWH-tree of our non-committed-format majority protocol

Our committed-format three-input majority protocol shown in Appendix A is not finite-runtime. Since there is no committed-format finite-runtime protocol other than the eight-card protocol proposed by Nishida et al. [17], it is an interesting open problem to determine whether we can have a committed-format finite-runtime three-input majority protocol using less than eight cards. It is also interesting to investigate relationship between card-based general majority function protocols and Turing world computations (cf. [7]).

Acknowledgements

We thank the anonymous referees, whose comments have helped us improve the presentation of the paper. We would like to thank Hideaki Sone for his

cooperation in preparing a Japanese draft version at an earlier stage of this work. This work was supported in part by JSPS KAKENHI Grant Numbers JP19J21153 and JP21K11881.

A Transformation to Committed Format

In this section, we show how to transform our non-committed-format protocol proposed in Section 2 to a committed-format one. Subsequently, we give the description of the derived committed-format majority protocol.

A.1 How to Transform

The transformation is inspired by the five-card Las Vegas AND protocol proposed by Abe et al. [3]. Briefly, their protocol realizes to output a commitment to $x \wedge y$, by adding further manipulations to the five-card trick. Since the output in our proposed protocol is derived by using (the four variants of) the five-card trick, it is possible to obtain a commitment to $\text{maj}(a, b, c)$ in a similar way to their protocol.

Here is the idea behind their protocol [3].

1. Perform Steps 1 to 3 of the five-card trick shown in Section 1.1:

$$\underbrace{\heartsuit \ ? \ ? \ ? \ ?}_a \rightarrow \dots \rightarrow \boxed{? \ ? \ ? \ ? \ ?}.$$

2. Turn over the center card; suppose that \clubsuit appears:

$$\boxed{? \ ? \ ? \ ? \ ?} \rightarrow \boxed{? \ ? \ \clubsuit \ ? \ ?}.$$

At this time, the resulting sequence of cards is one of the following four cases:

- (i) $\heartsuit \ \heartsuit \ \clubsuit \ \heartsuit \ \clubsuit$ if $a \wedge b = 0$;
- (ii) $\clubsuit \ \heartsuit \ \clubsuit \ \heartsuit \ \heartsuit$ if $a \wedge b = 0$;
- (iii) $\heartsuit \ \heartsuit \ \clubsuit \ \clubsuit \ \heartsuit$ if $a \wedge b = 1$;
- (iv) $\heartsuit \ \clubsuit \ \clubsuit \ \heartsuit \ \heartsuit$ if $a \wedge b = 1$.

3. Turn the center card face down. For the sake of illustration, let us represent the sequence as follows:

$$\underbrace{\boxed{? \ ? \ ? \ ? \ ?}}_x \underbrace{\boxed{? \ ? \ ? \ ? \ ?}}_y.$$

Observe that, in the cases (ii) and (iv), if we let the first and second cards be a commitment to $x \in \{0, 1\}$ and the third and fourth ones be a commitment to $y \in \{0, 1\}$, we have $x \oplus y = a \wedge b$. Therefore, by applying the committed-format XOR protocol [15] to them, one can obtain a commitment to $x \oplus y = a \wedge b$:

$$\underbrace{\boxed{? \ ? \ ? \ ? \ ?}}_x \underbrace{\boxed{? \ ? \ ? \ ? \ ?}}_y \rightarrow \underbrace{\boxed{\clubsuit \ \heartsuit \ ? \ ? \ ?}}_{a \wedge b} \text{ or } \underbrace{\boxed{\heartsuit \ \clubsuit \ ? \ ? \ ?}}_{\overline{a \wedge b}}.$$

Note that, even if it is the case (i) or (iii), one can still continue to execute the protocol without leaking information, as seen below.

In the next subsection, we present the description of our committed-format protocol using this idea.

A.2 Description

The following is our committed-format three-input majority protocol.

1. Perform Steps 1 to 3 of our non-committed-format protocol presented in Section 2.3:

$$\underbrace{\boxed{?}\boxed{?}\boxed{?}}_a \underbrace{\boxed{?}\boxed{?}\boxed{?}}_b \underbrace{\boxed{?}\boxed{?}}_c \rightarrow \dots \rightarrow \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}.$$

2. Reveal the first card. Assume that it is black, i.e., the result will be ♣-based:

$$\overbrace{\boxed{?}}^{\text{reveal}} \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?} \rightarrow \clubsuit\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}.$$

(In the case where a red card is shown, it works by interchanging the black cards and the red cards.)

3. Reveal the fourth card. If \heartsuit appears, turn it over and apply a random cut to the second through sixth cards; then, return to this step. If \clubsuit appears, turn it over and go to the next step.
4. Apply the XOR protocol [15] to the second through fifth cards as follows.
 - (a) Rearrange the order of the sequence as

$$\begin{array}{c} \clubsuit\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?} \\ \swarrow \searrow \\ \clubsuit\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?} \end{array}$$

- (b) Apply a random bisection cut to the second through fifth cards:

$$\clubsuit \left[\boxed{?}\boxed{?} \mid \boxed{?}\boxed{?} \right] \boxed{?} \rightarrow \clubsuit\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}.$$

- (c) Rearrange the order of the sequence as

$$\begin{array}{c} \clubsuit\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?} \\ \swarrow \searrow \\ \clubsuit\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?} \end{array}$$

5. Reveal the second and third cards.
 - (a) If $\clubsuit\heartsuit$ or $\heartsuit\clubsuit$ appears, we obtain a commitment to $\text{maj}(a, b, c)$ as follows:

$$\underbrace{\clubsuit\clubsuit\heartsuit}_{\text{maj}(a,b,c)} \underbrace{\boxed{?}\boxed{?}}_{\text{maj}(a,b,c)} \boxed{?} \quad \text{or} \quad \underbrace{\clubsuit\heartsuit\clubsuit}_{\text{maj}(a,b,c)} \underbrace{\boxed{?}\boxed{?}}_{\text{maj}(a,b,c)} \boxed{?}.$$

In the latter case, by swapping the left and right cards, we obtain a commitment to $\text{maj}(a, b, c)$.

(b) If $\heartsuit\heartsuit$ appears, then turn them over:

$$\clubsuit\heartsuit\heartsuit??? \rightarrow \clubsuit?????,$$

and rearrange the order of the sequence as

$$\begin{array}{c} \clubsuit\ ?\ ?\ ?\ ?\ ? \\ \swarrow \quad \searrow \quad \searrow \\ \clubsuit\ ?\ ?\ ?\ ?\ ? \end{array}.$$

Then, apply a random cut to the second through sixth cards and return to Step 3.

$$\clubsuit \langle ?\ ?\ ?\ ?\ ? \rangle \rightarrow \clubsuit\ ?\ ?\ ?\ ?\ ?.$$

Let us find the number of required shuffles for this committed-format protocol. The AND protocol proposed by Abe et al. takes the average of seven shuffles to terminate [3]. Since we apply a random bisection cut first in our protocol, it terminates with the expected number of eight shuffles in total. (It should be noted that a recent technique presented in [4] will reduce the number of shuffles further.)

A.3 Pseudocode

A pseudocode for our committed-format majority protocol is depicted in Algorithm 2.

A.4 Correctness and Security

To verify the correctness and security of our proposed committed-format majority protocol, we describe its KWH-tree in Figure 2; it guarantees that our protocol is correct and secure.

References

1. Abe, Y., Iwamoto, M., Ohta, K.: Efficient private PEZ protocols for symmetric functions. In: Hofheinz, D., Rosen, A. (eds.) *Theory of Cryptography. Lecture Notes in Computer Science*, vol. 11891, pp. 372–392. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-36030-6_15
2. Abe, Y., Iwamoto, M., Ohta, K.: How to detect malicious behaviors in a card-based majority voting protocol with three inputs. In: *2020 International Symposium on Information Theory and Its Applications (ISITA)*. pp. 377–381 (2020), <https://doi.org/10.34385/proc.65.C01-9>
3. Abe, Y., Hayashi, Y., Mizuki, T., Sone, H.: Five-card AND protocol in committed format using only practical shuffles. In: *5th ACM on ASIA Public-Key Cryptography Workshop*. pp. 3–8. APKC '18, Association for Computing Machinery, New York, NY, USA (2018), <https://doi.org/10.1145/3197507.3197510>

4. Abe, Y., Hayashi, Y., Mizuki, T., Sone, H.: Five-card AND computations in committed format using only uniform cyclic shuffles. *New Generation Computing* **39**(1), 97–114 (2021), <https://doi.org/10.1007/s00354-020-00110-2>
5. Balogh, J., Csirik, J.A., Ishai, Y., Kushilevitz, E.: Private computation using a PEZ dispenser. *Theoretical Computer Science* **306**(1), 69–84 (2003), [https://doi.org/10.1016/S0304-3975\(03\)00210-X](https://doi.org/10.1016/S0304-3975(03)00210-X)
6. Den Boer, B.: More efficient match-making and satisfiability the five card trick. In: Quisquater, J.J., Vandewalle, J. (eds.) *Advances in Cryptology—EUROCRYPT '89*. Lecture Notes in Computer Science, vol. 434, pp. 208–217. Springer, Berlin, Heidelberg (1990), https://doi.org/10.1007/3-540-46885-4_23
7. Dvořák, P., Koucký, M.: Barrington plays cards: The complexity of card-based protocols. In: Bläser, M., Monmege, B. (eds.) *Theoretical Aspects of Computer Science*. Leibniz International Proceedings in Informatics, vol. 187, pp. 26:1–26:17. Schloss Dagstuhl, Dagstuhl (2021), <https://doi.org/10.4230/LIPIcs.STACS.2021.26>
8. 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*. Lecture Notes in Computer Science, vol. 10626, pp. 126–155. Springer, Cham (2017), https://doi.org/10.1007/978-3-319-70700-6_5
9. Koch, A.: *Cryptographic Protocols from Physical Assumptions*. Ph.D. thesis, Karlsruhe Institute of Technology (2019), <https://doi.org/10.5445/IR/1000097756>
10. 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*. Lecture Notes in Computer Science, vol. 9452, pp. 783–807. Springer, Berlin, Heidelberg (2015), https://doi.org/10.1007/978-3-662-48797-6_32
11. Manabe, Y., Ono, H.: Secure card-based cryptographic protocols using private operations against malicious players. In: Maimut, D., Oprina, A.G., Sauveron, D. (eds.) *Innovative Security Solutions for Information Technology and Communications*. pp. 55–70. Springer, Cham (2021), https://doi.org/10.1007/978-3-030-69255-1_5
12. Marcedone, A., Wen, Z., Shi, E.: Secure dating with four or fewer cards. *Cryptology ePrint Archive*, Report 2015/1031 (2015), <https://eprint.iacr.org/2015/1031>
13. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. *International Journal of Information Security* **13**(1), 15–23 (2014), <https://doi.org/10.1007/s10207-013-0219-4>
14. Mizuki, T., Shizuya, H.: Computational model of card-based cryptographic protocols and its applications. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* **E100.A**(1), 3–11 (2017), <https://doi.org/10.1587/transfun.E100.A.3>
15. 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*. Lecture Notes in Computer Science, vol. 5598, pp. 358–369. Springer, Berlin, Heidelberg (2009), https://doi.org/10.1007/978-3-642-02270-8_36
16. 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*. Lecture Notes in Computer Science, vol. 10681, pp. 153–165. Springer, Cham (2017), https://doi.org/10.1007/978-3-319-72089-0_9

17. 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*. Lecture Notes in Computer Science, vol. 8273, pp. 193–204. Springer, Berlin, Heidelberg (2013), https://doi.org/10.1007/978-3-642-45008-2_16
18. Ono, H., Manabe, Y.: Card-based cryptographic protocols with the minimum number of cards using private operations. In: Zincir-Heywood, N., Bonfante, G., Debbabi, M., Garcia-Alfaro, J. (eds.) *Foundations and Practice of Security*. Lecture Notes in Computer Science, vol. 11358, pp. 193–207. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-18419-3_13
19. Ono, H., Manabe, Y.: Card-based cryptographic logical computations using private operations. *New Generation Computing* **39**, 19–40 (2021), <https://doi.org/10.1007/s00354-020-00113-z>
20. Pass, R., Shelat, A.: *A course in cryptography* (2010), www.cs.cornell.edu/~rafael/
21. Ruangwises, S., Itoh, T.: AND protocols using only uniform shuffles. In: van Bevern, R., Kucherov, G. (eds.) *Computer Science—Theory and Applications*. Lecture Notes in Computer Science, vol. 11532, pp. 349–358. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-19955-5_30
22. Salomaa, A.: *Public-Key Cryptography*. Texts in Theoretical Computer Science. An EATCS Series, Springer, Berlin Heidelberg (2013)
23. Shinagawa, K.: *On the Construction of Easy to Perform Card-Based Protocols*. Ph.D. thesis, Tokyo Institute of Technology (2020)
24. Ueda, I., Miyahara, D., Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: Secure implementations of a random bisection cut. *International Journal of Information Security* **19**(4), 445–452 (2020), <https://doi.org/10.1007/s10207-019-00463-w>
25. 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 (ISITA)*. pp. 218–222 (2018), <https://doi.org/10.23919/ISITA.2018.8664324>
26. Yasunaga, K.: Practical card-based protocol for three-input majority. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* **E103.A**(11), 1296–1298 (2020), <https://doi.org/10.1587/transfun.2020EAL2025>

Algorithm 2 Our committed-format majority protocol

input set:

$$\left\{ \begin{array}{l} \left(\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\clubsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\clubsuit} \right), \\ \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \\ \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right), \left(\frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit}, \frac{?}{\heartsuit} \right) \end{array} \right\}$$

1. (perm, (5 6))
2. (shuf, {id, (1 2)(3 6)})
3. (shuf, RC_{2,3,4,5,6})
4. (turn, {1})
5. **if** visible sequence = (♥, ?, ?, ?, ?, ?) **then**
6. (turn, {4})
7. **if** visible sequence = (♥, ?, ?, ♥, ?, ?) **then**
8. (turn, {4})
9. (shuf, RC_{2,3,4,5,6})
10. **go to 6**
11. **else if** visible sequence = (♥, ?, ?, ♣, ?, ?) **then**
12. (turn, {4})
13. (shuf, {id, (2 3)(4 5)})
14. (turn, {2, 3})
15. **if** visible sequence = (♥, ♥, ♥, ?, ?, ?) **then**
16. (turn, {2, 3})
17. (perm, (3 4 5 6))
18. (shuf, RC_{2,3,4,5,6})
19. **go to 6**
20. **else if** visible sequence = (♥, ♥, ♣, ?, ?, ?) **then**
21. (result, 4, 5)
22. **else if** visible sequence = (♥, ♣, ♥, ?, ?, ?) **then**
23. (result, 5, 4)
24. **else if** visible sequence = (♣, ?, ?, ?, ?, ?) **then**
25. (turn, {4})
26. **if** visible sequence = (♣, ?, ?, ♣, ?, ?) **then**
27. (turn, {4})
28. (shuf, RC_{2,3,4,5,6})
29. **go to 25**
30. **else if** visible sequence = (♣, ?, ?, ♥, ?, ?) **then**
31. (turn, {4})
32. (shuf, {id, (2 3)(4 5)})
33. (turn, {2, 3})
34. **if** visible sequence = (♣, ♣, ♣, ?, ?, ?) **then**
35. (turn, {2, 3})
36. (perm, (3 4 5 6))
37. (shuf, RC_{2,3,4,5,6})
38. **go to 25**
39. **else if** visible sequence = (♣, ♣, ♥, ?, ?, ?) **then**
40. (result, 4, 5)
41. **else if** visible sequence = (♣, ♥, ♣, ?, ?, ?) **then**
42. (result, 5, 4)

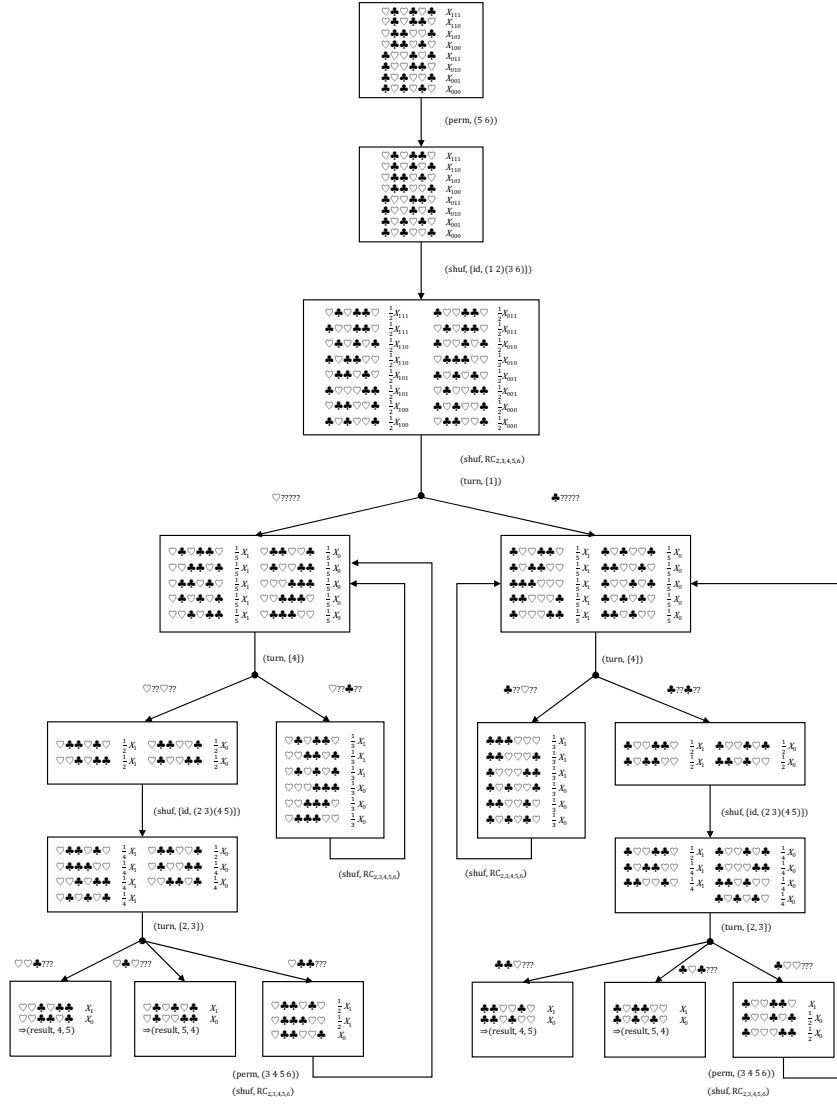


Fig. 2. The KWH-tree of our committed-format majority protocol.