Card-Based Secure Sorting Protocol*

Rikuo Haga[1], Kodai Toyoda[2], Yuto Shinoda[2], Daiki Miyahara[3,5], Kazumasa Shinagawa[4], Yuichi Hayashi[1,5], and Takaaki Mizuki[2,5]

[1] Nara Institute of Science and Technology, Ikoma, Japan
[2] Tohoku University, Sendai, Japan
mizuki+lncs[atmark]tohoku.ac.jp
[3] The University of Electro-Communications, Chofu, Japan
[4] Ibaraki University, Hitachi, Japan
[5] National Institute of Advanced Industrial Science and Technology, Koto, Japan

Abstract. The research area of card-based cryptography, which relies on a deck of physical cards to perform cryptographic functionalities, has been growing in recent years, ranging from basic secure computations, such as secure AND and XOR evaluations, to more complex tasks, such as Yao's Millionaires' problem and zero-knowledge proof. In this paper, we propose a card-based "secure sorting" protocol; although sorting is probably the most fundamental problem in computer science, secure sorting has not been addressed in the field of card-based cryptography yet. Given a sequence of face-down cards representing a collection of keys with values (to be sorted), our proposed protocol sorts them without leaking any information. As imagined, secure sorting provides many applications; for instance, we show how to apply our protocol to implementing an auction. Since many algorithms for computational problems (say, graph algorithms) use sorting as subroutines, we expect that our secure sorting protocol will be useful when constructing card-based secure computations regarding computational problems.

1 Introduction

A *secure computation* allows players (holding individual private inputs) to obtain the output value of a predetermined function while keeping information about the individual inputs secret. Since Yao [42] proposed a secure computation solving the Millionaires' problem in 1982, various secure computation protocols have been proposed (refer to [3,6] for survey). While such cryptographic protocols are typically designed to be run on computers, there is another research direction where protocols should be run on daily physical tools (instead of computers).

* This paper appears in Proceedings of IWSEC 2022. 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/10.1007/978-3-031-15255-9_12. 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.

Because such physical cryptographic protocols are executed by human hands, they have the advantage that players do not need to trust a computer as a black box and that the correctness and security can be easily understood (cf. [11,14]).

1.1 Card-based Cryptography

Among physical cryptographic protocols, many *card-based protocols* using a deck of physical cards, such as $\boxed{\clubsuit}\boxed{\clubsuit}\cdots\boxed{\heartsuit}\boxed{\heartsuit}\cdots$, have been constructed since Den Boer [5] proposed the first card-based protocol, called the "five-card trick." Actually, the research area of *card-based cryptography* has been growing in recent years [22,23], ranging from basic secure computations, such as secure AND and XOR evaluations (e.g., [1,4,16,20,24,25,27,30,41]), to more complex tasks, such as Yao's Millionaires' problem [21,26,29], secure ranking [40], and zero-knowledge proof (e.g., [2,8,35–37]).

1.2 Secure Sorting with Cards

In this study, we consider the fact that *secure sorting* (e.g., [7,10]) has not been addressed in card-based cryptography yet although sorting is probably the most fundamental problem in computer science. Thus, we propose a *card-based secure sorting protocol* for the first time. Given a sequence of cards representing a collection of keys with values (to be sorted), our proposed protocol sorts them without leaking any information. We describe the problem and goal more concretely, as follows.

Given a sequence of n pairs

$$(1, x_1), (2, x_2), (3, x_3), \ldots, (n, x_n), \tag{1}$$

we want to sort them by taking the second elements $x_1, x_2, \ldots, x_n$ as keys: that is, we want to obtain a sorted sequence

$$(\sigma^{-1}(1), x_{\sigma^{-1}(1)}), (\sigma^{-1}(2), x_{\sigma^{-1}(2)}), (\sigma^{-1}(3), x_{\sigma^{-1}(3)}), \ldots, (\sigma^{-1}(n), x_{\sigma^{-1}(n)}) \tag{2}$$

such that a permutation $\sigma \in S_n$ satisfies the following:

$$x_{\sigma^{-1}(i)} \geq x_{\sigma^{-1}(i+1)} \text{ for every } i \in \{1, \ldots, n-1\}, \tag{3}$$

where S_n is the symmetric group of degree n.

In addition, we want to hide the individual values $x_1, x_2, \ldots, x_n$ themselves as well as the sorted sequence. As typically done in card-based cryptography, we use a pair of face-down cards $\boxed{?}\boxed{?}$ to commit a one-bit value according to the following encoding:

$$\boxed{\clubsuit}\boxed{\heartsuit} = 0, \quad \boxed{\heartsuit}\boxed{\clubsuit} = 1. \tag{4}$$

Thus, assuming that $x_1, x_2, \ldots, x_n$ are m-bit values for some positive integer m, i.e., $x_1, x_2, \ldots, x_n \in \{0,1\}^m$, each x_i is assumed to be committed to $2m$

face-down cards:

$$\begin{array}{cl} \boxed{?}\boxed{?} & \leftarrow x_i[1] \\ \boxed{?}\boxed{?} & \leftarrow x_i[2] \\ \vdots & \quad\vdots \\ \boxed{?}\boxed{?} & \leftarrow x_i[m], \end{array}$$

where $x[j]$, $1 \leq j \leq m$, means the j-th bit of an m-bit value $x \in \{0,1\}^m$ (throughout the paper). We call this a *commitment* to $x_i \in \{0,1\}^m$, denoted by

$$\begin{array}{c} \boxed{?}\boxed{?} \\ \boxed{?}\boxed{?} \\ \vdots \\ \underbrace{\boxed{?}\boxed{?}}_{x_i}. \end{array}$$

Since the first elements in the sequence (1) above serve indices, we prepare numbered cards $\boxed{1}\boxed{2}\cdots\boxed{n}$ (whose backs are also $\boxed{?}$) and place them along with n commitments to $x_1, x_2, \ldots, x_n \in \{0,1\}^m$, as follows:

$$\begin{array}{cccc} \boxed{1} & \boxed{2} & \ldots & \boxed{n} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \ldots & \boxed{?}\boxed{?} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \ldots & \boxed{?}\boxed{?} \\ \vdots & \vdots & & \vdots \\ \underbrace{\boxed{?}\boxed{?}}_{x_1} & \underbrace{\boxed{?}\boxed{?}}_{x_2} & \ldots & \underbrace{\boxed{?}\boxed{?}}_{x_n}. \end{array} \tag{5}$$

This should be the input to a secure sorting protocol.

Given an input arrangement (5), after turning over the n numbered cards (on the first row), a secure sorting protocol should output the following arrangement without leaking any information about the input:

$$\begin{array}{cccc} \underset{\sigma^{-1}(1)}{\boxed{?}} & \underset{\sigma^{-1}(2)}{\boxed{?}} & \ldots & \underset{\sigma^{-1}(n)}{\boxed{?}} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \ldots & \boxed{?}\boxed{?} \\ \vdots & \vdots & & \vdots \\ \underbrace{\boxed{?}\boxed{?}}_{x_{\sigma^{-1}(1)}} & \underbrace{\boxed{?}\boxed{?}}_{x_{\sigma^{-1}(2)}} & \ldots & \underbrace{\boxed{?}\boxed{?}}_{x_{\sigma^{-1}(n)}}, \end{array} \tag{6}$$

such that the permutation $\sigma \in S_n$ satisfies the condition (3) above. Here,

$$\underset{i}{\boxed{?}}$$

(appearing on the first row in the arrangement (6)) for i, $1 \leq i \leq n$, represents a face-down numbered card whose face is $\boxed{i}$. Thus, the arrangement (6) serves a hidden form of the sorted sequence (2).

1.3 Contribution

In this paper, we present a concrete construction of a card-based secure sorting protocol. In other words, we construct a protocol that performs secure sorting using a physical deck of cards. Specifically, given an input arrangement as shown in (5) together with some additional cards, our protocol transforms it into an output arrangement as shown in (6) via a series of actions such as shuffling and revealing cards.

Actually, our protocol performs a *stable sort*, meaning that the resulting permutation σ satisfies the following property in addition to the condition (3):

$$\text{for every } i \in \{1, \ldots, n-1\}, \text{ if } x_{\sigma^{-1}(i)} = x_{\sigma^{-1}(i+1)}, \ \sigma^{-1}(i) < \sigma^{-1}(i+1).$$

That is, our protocol preserves the original order if two input commitments have the same value. For example, if the input sequence is $(1, 10), (2, 11), (3, 11), (4, 10)$, the output will be $(2, 11), (3, 11), (1, 10), (4, 10)$ because the order of $(1, 10)$ and $(4, 10)$ as well as the order of $(2, 11)$ and $(3, 11)$ should be kept.

Beyond just the purpose of sorting, our protocol has many applications. For example, consider an auction (sealed bid), and we would like to ensure that the information on the prices other than the successful bidder's one is not leaked to anyone. This can be achieved by our proposed secure sorting protocol (as will be seen in Section 4.1). In addition, a wide range of functions can be implemented by our protocol, from basic secure computations such as the multi-input AND computation and majority decision, to the Millionaires' problem and secret lottery protocol [39]. That is, our protocol serves a generic protocol in a sense.

1.4 Related work

As mentioned above, in the field of card-based cryptography, basic operations such as logical computation [1, 5, 9, 13, 17–19, 25] and applied computation protocols covering a wide range of applications have been proposed. The applied computation protocols include: a millionaire protocol [21, 26, 29] that reveals who is richer between two players while keeping their money information secret, a ranking protocol [40] that outputs only the ranking information while keeping the money information of multiple players secret, and a card-based covert lottery protocol [39], which determines the first and second moves of a game based on two players' secret preferences. In addition, there are applications to zero-knowledge proofs, which prove the existence of a solution to a puzzle problem without divulging any information about the solution [31–34, 37].

2 Preliminaries

In this section, we explain a deck of cards and shuffling operations which will be used in our proposed protocol.

2.1 Deck of Cards

As already seen, we use black cards $\boxed{\clubsuit}\boxed{\clubsuit}\cdots$, red cards $\boxed{\heartsuit}\boxed{\heartsuit}\cdots$, and numbered cards $\boxed{1}\boxed{2}\cdots\boxed{n}$. In addition, our protocol uses white cards $\boxed{}\boxed{}\cdots$ and marker cards $\boxed{\star}\boxed{\star}\cdots$. We assume that the sizes of all these cards are the same, and their backs, denoted by $\boxed{?}$, are identical. That is, they are indistinguishable except for designs on their fronts.

2.2 Pile-scramble Shuffle

A *pile-scramble* shuffle [12] is a shuffling action that completely randomizes the order of multiple piles consisting of the same number of cards. For a positive integer d, applying a pile-scramble shuffle to a sequence of d piles $(\mathsf{pile}_1, \mathsf{pile}_2, \ldots, \mathsf{pile}_d)$ results in $(\mathsf{pile}_{\pi^{-1}(1)}, \mathsf{pile}_{\pi^{-1}(2)}, \ldots, \mathsf{pile}_{\pi^{-1}(d)})$, where $\pi \in S_d$ is a uniformly distributed random permutation:

$$\begin{array}{cccc} 1 & 2 & & d \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \vdots & \vdots & & \vdots \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{array} \rightarrow \left[\begin{array}{c|c|c|c} \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \vdots & \vdots & & \vdots \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{array}\right] \rightarrow \begin{array}{cccc} \pi^{-1}(1) & \pi^{-1}(2) & \ldots & \pi^{-1}(d) \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \vdots & \vdots & & \vdots \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{array}.$$

Note that no one can know which permutation was applied.

Implementation methods for a pile-scramble shuffle have been discussed in the literature, e.g., [12, 40]. A typical implementation is to use envelopes; each pile of cards is fixed by using envelopes, and then players jointly shuffle them by hands (until the players are all satisfied).

2.3 Pile-shifting Shuffle

Another shuffling action is a *pile-shifting* shuffle, which cyclically and randomly shifts the order of piles consisting of the same number of cards [28, 38]. For a positive integer d, by applying a pile-shifting shuffle to d piles $(\mathsf{pile}_1, \mathsf{pile}_2, \ldots, \mathsf{pile}_d)$, we obtain $(\mathsf{pile}_{1+(r\%d)}, \mathsf{pile}_{1+(1+r\%d)}, \ldots, \mathsf{pile}_{1+(d+r\%d)})$, where $r \in \{0, 1, \ldots, d-1\}$ is a random number, and % denotes the remainder.

Similar to the pile-scramble shuffle explained in Section 2.2, a pile-shifting shuffle can be implemented by using envelopes.

2.4 Koch–Walzer Sort Protocol

In 2022, Koch and Walzer [15] proposed the "coupled sorting sub-protocol" that sorts multiple piles of cards according to the order of given numbered cards. We note that the distribution of the numbers written on the numbered cards is known in their protocol; in contrast, our protocol sorts multiple commitments to multi-bit values whose distribution is unknown; below is the more specific explanation.

Let us apply their idea to the input arrangement (5); turn over all the numbered cards, apply a pile-scramble shuffle to the arrangement, reveal all the commitments, and sort the whole piles according to the order of the revealed values:

$$\left[\begin{array}{c|c|c|c} \boxed{1} & \boxed{2} & & \boxed{n} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \cdots & \boxed{?}\boxed{?} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \cdots & \boxed{?}\boxed{?} \\ \vdots & \vdots & & \vdots \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \cdots & \boxed{?}\boxed{?} \end{array}\right] \rightarrow \begin{array}{ccc} \boxed{?} & \boxed{?} & & \boxed{?} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \cdots & \boxed{?}\boxed{?} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \cdots & \boxed{?}\boxed{?} \\ \vdots & \vdots & & \vdots \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \cdots & \boxed{?}\boxed{?} \end{array} \rightarrow$$

$$\begin{array}{cccc} \boxed{?} & \boxed{?} & & \boxed{?} \\ \boxed{\heartsuit}\boxed{\clubsuit} & \boxed{\heartsuit}\boxed{\clubsuit} & \cdots & \boxed{\clubsuit}\boxed{\heartsuit} \\ \boxed{\clubsuit}\boxed{\heartsuit} & \boxed{\heartsuit}\boxed{\clubsuit} & \cdots & \boxed{\clubsuit}\boxed{\heartsuit} \\ \vdots & \vdots & & \vdots \\ \boxed{\clubsuit}\boxed{\heartsuit} & \boxed{\clubsuit}\boxed{\heartsuit} & \cdots & \boxed{\heartsuit}\boxed{\clubsuit} \end{array} \rightarrow \begin{array}{cccc} \boxed{?} & \boxed{?} & & \boxed{?} \\ \boxed{\heartsuit}\boxed{\clubsuit} & \boxed{\heartsuit}\boxed{\clubsuit} & \cdots & \boxed{\clubsuit}\boxed{\heartsuit} \\ \boxed{\heartsuit}\boxed{\clubsuit} & \boxed{\clubsuit}\boxed{\heartsuit} & \cdots & \boxed{\clubsuit}\boxed{\heartsuit} \\ \vdots & \vdots & & \vdots \\ \boxed{\clubsuit}\boxed{\heartsuit} & \boxed{\clubsuit}\boxed{\heartsuit} & \cdots & \boxed{\heartsuit}\boxed{\clubsuit} \end{array}.$$

Then, we obtain sorted indices

$$\underset{\sigma^{-1}(1)}{\boxed{?}}\;\underset{\sigma^{-1}(2)}{\boxed{?}}\;\cdots\;\underset{\sigma^{-1}(n)}{\boxed{?}},$$

but the distribution of the key values $x_1, x_2, \ldots, x_n$ are leaked. It is non-trivial to sort the arrangement (5) without leaking any information; we will construct a protocol to overcome this difficulty.

3 Our Proposed Secure Sorting Protocol

In this section, we construct a card-based secure sorting protocol. In Section 3.1, we illustrate an overall flow of our protocol. In Section 3.2, by showing a working example, we present the idea of how to securely sort commitments. In Section 3.3, we give the complete description of our protocol. In Section 3.4, we prove the security of our protocol.

Table 1. Overall flow of our protocol

	(a)			(b)			(c)		
	1	2	3	1	3	2	1	2	3
1st-bit →	♡♣	♣♡	♡♣	♡♣	♡♣	♣♡	♡♣	♣♡	♡♣
2nd-bit →	♡♣	♡♣	♣♡	♡♣	♣♡	♡♣	♡♣	♡♣	♣♡
3rd-bit →	♣♡	♡♣	♡♣	♣♡	♡♣	♡♣	♣♡	♡♣	♡♣
4th-bit →	♡♣	♣♡	♡♣	♡♣	♡♣	♣♡	♡♣	♣♡	♡♣
				(d)			(e)		
				2	3	1	3	1	2
				♣♡	♡♣	♡♣	♡♣	♡♣	♣♡
				♡♣	♣♡	♡♣	♣♡	♡♣	♡♣
				♡♣	♡♣	♣♡	♡♣	♣♡	♡♣
				♣♡	♡♣	♡♣	♡♣	♡♣	♣♡

3.1 Overall Flow

Take the sequence

$$(1, 1011), (2, 0110), (3, 1101)$$

as a working example (to be sorted). As mentioned in Section 1.2, we use commitments (consisting of face-down cards) to represent such an input: That is, we now have

$$\begin{array}{ccc} \boxed{1} & \boxed{2} & \boxed{3} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \boxed{?}\boxed{?} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \boxed{?}\boxed{?} \\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \boxed{?}\boxed{?} \\ \underbrace{\boxed{?}\boxed{?}}_{1011} & \underbrace{\boxed{?}\boxed{?}}_{0110} & \underbrace{\boxed{?}\boxed{?}}_{1101} \end{array} \tag{7}$$

as an input arrangement, whose front sides satisfy (a) in Table 1.

In our protocol, we sort the commitments (together with the numbered cards on the first row) bit by bit in a stable manner. Thus, we first apply a stable sort based on the first bit, i.e., the least significant bit; then, the resulting sequence is

$$(1, 1011), (3, 1101), (2, 0110),$$

which corresponds to (b) in Table 1. Next, we apply a stable sort based on the second bit, resulting in

$$(1, 1011), (2, 0110), (3, 1101),$$

which corresponds to (c) in Table 1. In the same manner, we have

$$(2, 0110), (3, 1101), (1, 1011)$$

corresponding to (d) and then

$$(3, 1101), (1, 1011), (2, 0110)$$

corresponding to (e).

In this way, we sort the input arrangement. In the next subsection, we show how to transform the arrangement without leaking any information about the input.

3.2 How to Securely Sort

Assume that we want to perform a stable sort based on the first bits, given the arrangement (7) above. We here use additional white cards $\boxed{}\boxed{}\boxed{}$ and numbered cards $\boxed{1}\boxed{1}\boxed{2}\boxed{2}\boxed{3}\boxed{3}$.

First, place the three white cards as follows, and turn over the cards on the first row:

$$\begin{matrix}\boxed{1}\boxed{} & \boxed{2}\boxed{} & \boxed{3}\boxed{}\\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \boxed{?}\boxed{?}\\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \boxed{?}\boxed{?}\\ \boxed{?}\boxed{?} & \boxed{?}\boxed{?} & \boxed{?}\boxed{?}\\ \underbrace{\boxed{?}\boxed{?}}_{1011} & \underbrace{\boxed{?}\boxed{?}}_{0110} & \underbrace{\boxed{?}\boxed{?}}_{1101}\end{matrix} \rightarrow \begin{matrix}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\\ \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\\ \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\\ \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\\ \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\,.\end{matrix}$$

Next, after turning over the additional numbered cards $\boxed{1}\boxed{1}\boxed{2}\boxed{2}\boxed{3}\boxed{3}$, we apply a pile-scramble shuffle as follows:

$$\underset{1}{\boxed{?}}\underset{1}{\boxed{?}}\underset{2}{\boxed{?}}\underset{2}{\boxed{?}}\underset{3}{\boxed{?}}\underset{3}{\boxed{?}} \rightarrow \left[\,\boxed{?}\boxed{?}\,\middle|\,\boxed{?}\boxed{?}\,\middle|\,\boxed{?}\boxed{?}\,\right] \rightarrow \underset{r_1}{\boxed{?}}\underset{r_1}{\boxed{?}}\underset{r_2}{\boxed{?}}\underset{r_2}{\boxed{?}}\underset{r_3}{\boxed{?}}\underset{r_3}{\boxed{?}},$$

where (r_1, r_2, r_3) is a random rearrangement of $(1, 2, 3)$ generated by the pile-scramble shuffle. Then, place these six cards above the arrangement, as follows:

$$\begin{array}{rcccccc}
 & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
 & r_1 & r_1 & r_2 & r_2 & r_3 & r_3 \\
 & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
\text{1st-bit} \rightarrow & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
 & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
 & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
 & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \, .
\end{array}$$

We say that a column is *white* if its second cards is $\boxed{}$; thus, in this case, the second, forth, and sixth columns are white.

Remember that we want to perform a stable sort based on the first bits; however, we cannot open the cards corresponding to the first bits (i.e., the cards on the third row), of course. Therefore, we apply a pile-scramble shuffle to each commitment (together with the four cards above it):

$$\left[\begin{array}{c|c}
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?}
\end{array}\right]
\left[\begin{array}{c|c}
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?}
\end{array}\right]
\left[\begin{array}{c|c}
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?}
\end{array}\right].$$

As known from the encoding rule (4), now, revealing the cards on the third row does not leak any information (because each bit value was negated with a probability of exactly 1/2); therefore, reveal those cards:

$$\begin{array}{cccccc}
\boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
\boxed{\heartsuit} & \boxed{\clubsuit} & \boxed{\heartsuit} & \boxed{\clubsuit} & \boxed{\clubsuit} & \boxed{\heartsuit} \\
\boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \, .
\end{array}$$

Then, we perform a stable sort according to the revealed values (based on $\heartsuit > \clubsuit$) while keeping the order of cards inside each column unchanged:

$$\begin{array}{cccccc}
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{\heartsuit}&\boxed{\heartsuit}&\boxed{\heartsuit}&\boxed{\clubsuit}&\boxed{\clubsuit}&\boxed{\clubsuit}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\,.
\end{array}$$

Next, using the existing technique [40] (as the details will be explained in Step 5 of our protocol presented in Sect. 3.3), we take out all the white columns:

$$\begin{array}{ccccccc}
\boxed{?}&\boxed{?}&\boxed{?}&\qquad&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&&\boxed{}&\boxed{}&\boxed{}\\
\boxed{?}&\boxed{?}&\boxed{?}&&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&&\boxed{?}&\boxed{?}&\boxed{?}\,.
\end{array}$$

Finally, reveal all the cards on the first row, and move each white column so that the commitment is restored. Note that the cards in the first row are shuffled so that no information is leaked when they are turned over:

$$\begin{array}{cccccc}
\boxed{3}&\boxed{3}&\boxed{1}&\boxed{1}&\boxed{2}&\boxed{2}\\
\boxed{?}&\boxed{}&\boxed{?}&\boxed{}&\boxed{?}&\boxed{}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\\
\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}&\boxed{?}\,.
\end{array}$$

3.3 Description of Our Protocol

In this subsection, we give the complete description of our secure sorting protocol.

Given an arrangement as shown in Eq. (5) along with additional n white cards $\boxed{}\boxed{}\cdots\boxed{}$ and $2n$ numbered cards $\boxed{1}\boxed{1}\boxed{2}\boxed{2}\cdots\boxed{n}\boxed{n}$, our protocol proceeds as follows.

1. To the input arrangement, add the n white cards as below, and turn over all the cards on the first row:

$$\begin{array}{c}
\boxed{1}\boxed{}\,\boxed{2}\boxed{}\ldots\boxed{n}\boxed{} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\vdots \qquad \vdots \qquad\quad \vdots \\
\underbrace{\boxed{?}\boxed{?}}_{x_1}\,\underbrace{\boxed{?}\boxed{?}}_{x_2}\ldots\underbrace{\boxed{?}\boxed{?}}_{x_n}
\end{array}
\rightarrow
\begin{array}{c}
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\vdots \qquad \vdots \qquad\quad \vdots \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?}\,.
\end{array}
\tag{8}$$

Set $j := 1$.

2. Turn over the additional $2n$ numbered cards $\boxed{1}\boxed{1}\boxed{2}\boxed{2}\ \ldots\boxed{n}\boxed{n}$, and apply a pile-scramble shuffle as follows:

$$\underset{1}{\boxed{?}}\underset{1}{\boxed{?}}\,\underset{2}{\boxed{?}}\underset{2}{\boxed{?}}\cdots\underset{n}{\boxed{?}}\underset{n}{\boxed{?}} \rightarrow \left[\boxed{?}\boxed{?}\middle|\boxed{?}\boxed{?}\middle|\cdots\middle|\boxed{?}\boxed{?}\right] \rightarrow$$

$$\underset{r_1}{\boxed{?}}\underset{r_1}{\boxed{?}}\,\underset{r_2}{\boxed{?}}\underset{r_2}{\boxed{?}}\cdots\underset{r_n}{\boxed{?}}\underset{r_n}{\boxed{?}},$$

where $(r_1, r_2, \ldots, r_n)$ is a random rearrangement of $(1, 2, \ldots, n)$ generated by the pile-scramble shuffle. Then, place these $2n$ cards above the arrangement (8), as follows:

$$\begin{array}{c}
\underset{r_1}{\boxed{?}}\underset{r_1}{\boxed{?}}\,\underset{r_2}{\boxed{?}}\underset{r_2}{\boxed{?}}\ldots\underset{r_n}{\boxed{?}}\underset{r_n}{\boxed{?}} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\vdots \qquad \vdots \qquad\quad \vdots \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?}\,.
\end{array}$$

3. Apply a pile-scramble shuffle to the $(2i-1)$-th and $2i$-th columns for every i, $1 \leq i \leq n$:

$$\left[\begin{array}{c|c}
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\vdots & \vdots \\
\boxed{?} & \boxed{?}
\end{array}\right]
\left[\begin{array}{c|c}
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\vdots & \vdots \\
\boxed{?} & \boxed{?}
\end{array}\right]
\cdots
\left[\begin{array}{c|c}
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\boxed{?} & \boxed{?} \\
\vdots & \vdots \\
\boxed{?} & \boxed{?}
\end{array}\right]
\rightarrow
\begin{array}{c}
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?} \\
\vdots \qquad \vdots \qquad\quad \vdots \\
\boxed{?}\boxed{?}\,\boxed{?}\boxed{?}\ldots\boxed{?}\boxed{?}\,.
\end{array}$$

4. Reveal the cards corresponding to the j-th bits and perform a stable sort as follows:

$$\begin{array}{rccc}
 & \boxed{?}\boxed{?}\cdots\boxed{?}\boxed{?} & & \boxed{?}\cdots\boxed{?}\boxed{?}\cdots\boxed{?} \\
 & \boxed{?}\boxed{?}\cdots\boxed{?}\boxed{?} & & \boxed{?}\cdots\boxed{?}\boxed{?}\cdots\boxed{?} \\
 & \vdots\quad\vdots\qquad\vdots\quad\vdots & & \vdots\qquad\vdots\quad\vdots\qquad\vdots \\
j\text{-th bit} & \boxed{\heartsuit}\boxed{\clubsuit}\cdots\boxed{\clubsuit}\boxed{\heartsuit} & \rightarrow & \boxed{\heartsuit}\cdots\boxed{\heartsuit}\boxed{\clubsuit}\cdots\boxed{\clubsuit} \\
 & \vdots\quad\vdots\qquad\vdots\quad\vdots & & \vdots\qquad\vdots\quad\vdots\qquad\vdots \\
 & \boxed{?}\boxed{?}\cdots\boxed{?}\boxed{?} & & \boxed{?}\cdots\boxed{?}\boxed{?}\cdots\boxed{?}\,.
\end{array}$$

5. Take out all the white columns, as follows:
   (a) Turn all the face-up cards face down, and place to the left a new column consisting of marker cards, as follows:

$$\begin{array}{cc}
\boxed{\star} & \boxed{?}\cdots\boxed{?}\boxed{?}\cdots\boxed{?} \\
\boxed{\star} & \boxed{?}\cdots\boxed{?}\boxed{?}\cdots\boxed{?} \\
\boxed{\star} & \boxed{?}\cdots\boxed{?}\boxed{?}\cdots\boxed{?} \\
\vdots & \vdots\qquad\vdots\quad\vdots\qquad\vdots \\
\boxed{\star} & \boxed{?}\cdots\boxed{?}\boxed{?}\cdots\boxed{?}\,.
\end{array}$$

   (b) Turn all the face-up cards face-down, and apply a pile-shifting shuffle to the whole columns. Then, turn over the card on the second row of the first column; if it is a white card $\boxed{}$, then the column is white and hence, remove it. If the total number of removed white columns reaches n, proceed to the next step. Otherwise, return to the beginning of this step.

6. Reveal all the cards on the first row; then, ignore the marker column. Restore each commitment by placing the white columns at the appropriate positions:

$$\begin{array}{cccc}
\boxed{2}\boxed{4}\cdots\boxed{3} & \boxed{3}\boxed{5}\cdots\boxed{1} & & \boxed{2}\boxed{2}\boxed{4}\boxed{4}\cdots\boxed{3}\boxed{3} \\
\boxed{?}\boxed{?}\cdots\boxed{?} & \boxed{?}\boxed{?}\cdots\boxed{?} & & \boxed{?}\boxed{?}\boxed{?}\boxed{?}\cdots\boxed{?}\boxed{?} \\
\boxed{?}\boxed{?}\cdots\boxed{?} & \boxed{?}\boxed{?}\cdots\boxed{?} & \rightarrow & \boxed{?}\boxed{?}\boxed{?}\boxed{?}\cdots\boxed{?}\boxed{?} \\
\vdots\quad\vdots\qquad\vdots & \vdots\quad\vdots\qquad\vdots & & \vdots\quad\vdots\quad\vdots\quad\vdots\qquad\vdots\quad\vdots \\
\boxed{?}\boxed{?}\cdots\boxed{?} & \boxed{?}\boxed{?}\cdots\boxed{?} & & \boxed{?}\boxed{?}\boxed{?}\boxed{?}\cdots\boxed{?}\boxed{?}\,.
\end{array}$$

7. Remove the cards in the first row.
8. Set $j := j + 1$. If $j < m$, return to Step 2.

3.4 Security

Information about inputs and outputs is generally leaked when revealing cards in card-based protocols; thus, we focus on Steps 4, 5, and 6 of our protocol. In Step 4, we reveal the n bit-values on the j-th bit. Since each bit-value is randomized by the shuffle in Step 3, no information about the values is leaked. In Step 5(b), because a pile-shifting shuffle is applied before the second card is revealed, the revealed card does not leak information. Similarly, in Step 6, no information leaks. In conclusion, the proposed protocol is information-theoretically secure.

3.5 Optimization

Because Steps 5(b) has a repetition, our protocol is a Las-Vegas protocol. We note that our protocol can be converted to a finite-runtime protocol by applying the existing technique used in the secure ranking protocols [40].

Remember that in Step 5(b), we extract the n white columns from the $2n+1$ ones. The above-mentioned technique enables us to achieve the same task in finite runtime using two pile-scramble shuffles, n pile-shifting shuffles, and n^2 additional cards.

4 Applications of Card-based Secure Sorting

In this section, we show how to apply our secure sorting protocol proposed in Section 3 to achieving an auction and secure computation of threshold functions. Recall that the proposed protocol outputs an arrangement shown in Eq. (6).

4.1 Auction

Let auction be the functionality of auction. Since auction only needs to output the maximum bid price and its bidder, it can be written as follows using a permutation σ corresponding to a stable sort:

$$\mathsf{auction}(x_1, \ldots, x_n) = (x_{\sigma^{-1}(1)}, \sigma^{-1}(1)).$$

That is, auction can be realized by revealing the first commitment and its numbered card after executing our secure sorting protocol; the former indicates the price and the latter indicates the winner. In the case where there is a player who bids the same price as the winning bid price, the player can confirm by turning the second price and the numbered card. Therefore, ties can also be detected. Alternatively, using the existing XOR and OR protocols (e.g., [25]), the players can determine whether the second commitment has the same value as the first one without revealing its value.

4.2 Secure Threshold Function Evaluation

We define a threshold function thr_n^t that outputs 1 if and only if the sum of n bits $x_1, \ldots, x_n \in \{0, 1\}$ is greater than or equal to $t \in \{1, \ldots, n\}$:

$$\mathsf{thr}_n^t(x_1, \ldots, x_n) \coloneqq \begin{cases} 1 & \text{if } \sum_{i=1}^n x_i \geq t, \\ 0 & \text{otherwise.} \end{cases}$$

Using the permutation σ corresponding to a stable sort, we have the following:

$$\mathsf{thr}_n^t(x_1, \ldots, x_n) = x_{\sigma^{-1}(t)}.$$

Therefore, thr_n^t can be realized by turning over the t-th commitment after executing the proposed protocol with $m = 1$.

5 Conclusion

In this paper, we proposed a card-based secure sorting protocol. The protocol itself is useful as well as it can provide various applications. The protocol is based on the representation of each player's value as a binary string, and sorts the values bit by bit from the least significant bit. As examples of the application of our protocol, we showed how to implement an auction and secure computations of threshold functions. This protocol can also be applied to a computation similar to a ranking computation [40] and to a covert lottery protocol [39] (although we omit the details).

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 JP21K11881 and JP19H01104.

References

1. Abe, Y., Hayashi, Y., Mizuki, T., Sone, H.: Five-card AND computations in committed format using only uniform cyclic shuffles. New Gener. Comput. **39**, 97–114 (2021), https://doi.org/10.1007/s00354-020-00110-2
2. Bultel, X., Dreier, J., Dumas, J.G., Lafourcade, P.: Physical zero-knowledge proofs for Akari, Takuzu, Kakuro and KenKen. In: Demaine, E.D., Grandoni, F. (eds.) Fun with Algorithms. LIPIcs, vol. 49, pp. 8:1–8:20. Schloss Dagstuhl, Dagstuhl, Germany (2016), https://doi.org/10.4230/LIPIcs.FUN.2016.8
3. Cramer, R., Damgård, I.B., et al.: Secure multiparty computation and secret sharing. Cambridge University Press (2015), https://ir.cwi.nl/pub/23529

4. Crépeau, C., Kilian, J.: Discreet solitary games. In: Stinson, D.R. (ed.) Advances in Cryptology—CRYPTO' 93. LNCS, vol. 773, pp. 319–330. Springer, Berlin, Heidelberg (1994), https://doi.org/10.1007/3-540-48329-2_27
5. 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. LNCS, vol. 434, pp. 208–217. Springer, Berlin, Heidelberg (1990), https://doi.org/10.1007/3-540-46885-4_23
6. Evans, D., Kolesnikov, V., Rosulek, M.: A pragmatic introduction to secure multi-party computation. Foundations and Trends in Privacy and Security **2**(2–3), 70–246 (2018), https://doi.org/10.1561/3300000019
7. Goodrich, M.T.: Randomized shellsort: A simple data-oblivious sorting algorithm. J. ACM **58**(6), 1–26 (2011), https://doi.org/10.1145/2049697.2049701
8. Gradwohl, R., Naor, M., Pinkas, B., Rothblum, G.N.: Cryptographic and physical zero-knowledge proof systems for solutions of Sudoku puzzles. Theory of Computing Systems **44**(2), 245–268 (2009), https://doi.org/10.1007/s00224-008-9119-9
9. Haga, R., Hayashi, Y., Miyahara, D., Mizuki, T.: Card-minimal protocols for three-input functions with standard playing cards. In: Progress in Cryptology—AFRICACRYPT 2022. LNCS, Springer, Cham (2022), to appear
10. Hamada, K., Kikuchi, R., Ikarashi, D., Chida, K., Takahashi, K.: Practically efficient multi-party sorting protocols from comparison sort algorithms. In: Kwon, T., Lee, M.K., Kwon, D. (eds.) Information Security and Cryptology – ICISC 2012. LNCS, vol. 7839, pp. 202–216. Springer, Berlin, Heidelberg (2013), https://doi.org/10.1007/978-3-642-37682-5_15
11. Hanaoka, G.: Towards user-friendly cryptography. In: Phan, R.C.W., Yung, M. (eds.) Paradigms in Cryptology–Mycrypt 2016. Malicious and Exploratory Cryptology. LNCS, vol. 10311, pp. 481–484. Springer, Cham (2017), https://doi.org/10.1007/978-3-319-61273-7_24
12. Ishikawa, R., Chida, E., Mizuki, T.: Efficient card-based protocols for generating a hidden random permutation without fixed points. In: Calude, C.S., Dinneen, M.J. (eds.) Unconventional Computation and Natural Computation. LNCS, vol. 9252, pp. 215–226. Springer, Cham (2015), https://doi.org/10.1007/978-3-319-21819-9_16
13. Isuzugawa, R., Toyoda, K., Sasaki, Y., Miyahara, D., Mizuki, T.: A card-minimal three-input AND protocol using two shuffles. In: Chen, C., Hon, W., Hung, L., Lee, C. (eds.) Computing and Combinatorics. LNCS, vol. 13025, pp. 668–679. Springer, Cham (2021), https://doi.org/10.1007/978-3-030-89543-3_55
14. Koch, A.: The landscape of security from physical assumptions. In: 2021 IEEE Information Theory Workshop (ITW). pp. 1–6. IEEE, Los Alamitos, CA, USA (2021), https://doi.org/10.1109/ITW48936.2021.9611501
15. Koch, A., Walzer, S.: Private function evaluation with cards. New Gener. Comput. **40**, 115–147 (2022), https://doi.org/10.1007/s00354-021-00149-9
16. 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
17. Koyama, H., Miyahara, D., Mizuki, T., Sone, H.: A secure three-input AND protocol with a standard deck of minimal cards. In: Santhanam, R., Musatov, D. (eds.) Computer Science – Theory and Applications. LNCS, vol. 12730, pp. 242–256. Springer, Cham (2021), https://doi.org/10.1007/978-3-030-79416-3_14

18. Koyama, H., Toyoda, K., Miyahara, D., Mizuki, T.: New card-based copy protocols using only random cuts. In: ASIA Public-Key Cryptography Workshop. p. 13–22. APKC '21, ACM, New York, NY, USA (2021), https://doi.org/10.1145/3457338.3458297
19. Kuzuma, T., Toyoda, K., Miyahara, D., Mizuki, T.: Card-based single-shuffle protocols for secure multiple-input AND and XOR computations. In: ASIA Public-Key Cryptography. pp. 51–58. ACM, NY (2022), https://doi.org/10.1145/3494105.3526236
20. Manabe, Y., Ono, H.: Card-based cryptographic protocols with malicious players using private operations. New Gener. Comput. **40**, 67–93 (2022), https://doi.org/10.1007/s00354-021-00148-w
21. Miyahara, D., Hayashi, Y., Mizuki, T., Sone, H.: Practical card-based implementations of Yao's millionaire protocol. Theor. Comput. Sci. **803**, 207–221 (2020), https://doi.org/10.1016/j.tcs.2019.11.005
22. Mizuki, T.: Preface: Special issue on card-based cryptography. New Gener. Comput. **39**, 1–2 (2021), https://doi.org/10.1007/s00354-021-00127-1
23. Mizuki, T.: Preface: Special issue on card-based cryptography 2. New Gener. Comput. **40**, 47–48 (2022), https://doi.org/10.1007/s00354-022-00170-6
24. 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
25. 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
26. Nakai, T., Misawa, Y., Tokushige, Y., Iwamoto, M., Ohta, K.: How to solve millionaires' problem with two kinds of cards. New Gener. Comput. **39**, 73–96 (2021), https://doi.org/10.1007/s00354-020-00118-8
27. Niemi, V., Renvall, A.: Secure multiparty computations without computers. Theor. Comput. Sci. **191**(1–2), 173–183 (1998), https://doi.org/10.1016/S0304-3975(97)00107-2
28. Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: Pile-shifting scramble for card-based protocols. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **101**(9), 1494–1502 (2018), https://doi.org/10.1587/transfun.E101.A.1494
29. Ono, H., Manabe, Y.: Efficient card-based cryptographic protocols for the millionaires' problem using private input operations. In: Asia Joint Conference on Information Security (AsiaJCIS). pp. 23–28 (2018), https://doi.org/10.1109/AsiaJCIS.2018.00013
30. Ono, H., Manabe, Y.: Card-based cryptographic logical computations using private operations. New Gener. Comput. **39**, 19–40 (2021), https://doi.org/10.1007/s00354-020-00113-z
31. Robert, L., Miyahara, D., Lafourcade, P., Mizuki, T.: Physical zero-knowledge proof for Suguru puzzle. In: Devismes, S., Mittal, N. (eds.) Stabilization, Safety, and Security of Distributed Systems. LNCS, vol. 12514, pp. 235–247. Springer, Cham (2020), https://doi.org/10.1007/978-3-030-64348-5_19
32. Robert, L., Miyahara, D., Lafourcade, P., Mizuki, T.: Interactive physical ZKP for connectivity: Applications to Nurikabe and Hitori. In: De Mol, L., Weiermann, A., Manea, F., Fernández-Duque, D. (eds.) Connecting with Computability. LNCS, vol. 12813, pp. 373–384. Springer, Cham (2021), https://doi.org/10.1007/978-3-030-80049-9_37

33. Robert, L., Miyahara, D., Lafourcade, P., Libralesso, L., Mizuki, T.: Physical zero-knowledge proof and NP-completeness proof of Suguru puzzle. Information and Computation p. 104858 (2021). https://doi.org/https://doi.org/10.1016/j.ic.2021.104858, https://www.sciencedirect.com/science/article/pii/S0890540121001905, in press
34. Robert, L., Miyahara, D., Lafourcade, P., Mizuki, T.: Card-based ZKP for connectivity: Applications to Nurikabe, Hitori, and Heyawake. New Gener. Comput. pp. 1–23 (2022), https://doi.org/10.1007/s00354-022-00155-5, in press
35. Ruangwises, S.: Two standard decks of playing cards are sufficient for a ZKP for Sudoku. New Gener. Comput. **40**, 49–65 (2022), https://doi.org/10.1007/s00354-021-00146-y
36. Ruangwises, S., Itoh, T.: Physical zero-knowledge proof for Ripple Effect. Theor. Comput. Sci. **895**, 115–123 (2021), https://doi.org/10.1016/j.tcs.2021.09.034
37. Sasaki, T., Miyahara, D., Mizuki, T., Sone, H.: Efficient card-based zero-knowledge proof for Sudoku. Theor. Comput. Sci. **839**, 135–142 (2020), https://doi.org/10.1016/j.tcs.2020.05.036
38. Shinagawa, K., Mizuki, T., Schuldt, J., Nuida, K., Kanayama, N., Nishide, T., Hanaoka, G., Okamoto, E.: Card-based protocols using regular polygon cards. IEICE Trans. Fundam. **E100.A**(9), 1900–1909 (2017), https://doi.org/10.1587/transfun.E100.A.1900
39. Shinoda, Y., Miyahara, D., Shinagawa, K., Mizuki, T., Sone, H.: Card-based covert lottery. In: Maimuţ, D., Oprina, A., Sauveron, D. (eds.) Innovative Security Solutions for Information Technology and Communications. LNCS, vol. 12596, pp. 257–270. Springer, Cham (2021), https://doi.org/10.1007/978-3-030-69255-1_17
40. Takashima, K., Abe, Y., Sasaki, T., Miyahara, D., Shinagawa, K., Mizuki, T., Sone, H.: Card-based protocols for secure ranking computations. Theor. Comput. Sci. **845**, 122–135 (2020), https://doi.org/10.1016/j.tcs.2020.09.008
41. Toyoda, K., Miyahara, D., Mizuki, T., Sone, H.: Six-card finite-runtime XOR protocol with only random cut. In: ACM Workshop on ASIA Public-Key Cryptography. pp. 2–8. APKC '20, ACM, New York (2020), https://doi.org/10.1145/3384940.3388961
42. Yao, A.C.: Protocols for secure computations. In: Foundations of Computer Science. pp. 160–164. IEEE Computer Society, Washington, DC, USA (1982), https://doi.org/10.1109/SFCS.1982.88