Playing President with Virtual Players: How to Play Multiple Cards of a Kind

Daiki Miyahara
The University of Electro-Communications, Japan
National Institute of Advanced Industrial Science and Technology, Japan

Pascal Lafourcade
Université Clermont Auvergne, CNRS, Clermont Auvergne INP, LIMOS, 63000 Clermont-Ferrand, France

Takaaki Mizuki
Cyberscience Center, Tohoku University, Japan
National Institute of Advanced Industrial Science and Technology, Japan

Kazumasa Shinagawa[1]
University of Tsukuba, Japan
Kyushu University, Japan
National Institute of Advanced Industrial Science and Technology, Japan

Abstract

President is a popular card game in which players may play one to four cards of the same rank. Since it is less enjoyable with too few human players, to address this, we study the player-simulation problem for President: realizing the moves of a virtual player while keeping its hand hidden. We propose a selection protocol, which selects multiple cards of the same rank uniformly at random from a hidden virtual player's hand, whose rank exceeds the latest played cards. Our construction reduces the task to secure sorting, so the overall efficiency is dominated by the underlying sorting protocol. To address this bottleneck, we design an efficient sorting protocol, which reduces the number of steps from $O(m \log m)$ to $O(m)$, compared to the existing sorting protocols.

2012 ACM Subject Classification Security and privacy → Cryptography

Keywords and phrases Card-Based Cryptography, Player Simulation Problem, President

Digital Object Identifier 10.4230/LIPIcs.FUN.2026.26

Funding This work was supported in part by ANR PROJECT PRIVA-SIQ ANR-23-CE39-0008, JSPS KAKENHI Grant Numbers JP21K17702, JP23H00479, and JP24K02938, JST CREST Grant Number JPMJCR22M1, JSPS Bilateral Joint Research Projects JPJSBP120253206, and JST K Program Grant Number JPMJKP24U2, Japan.

Acknowledgements We thank the anonymous reviewers, whose comments have helped us improve the presentation of the paper. We thank Hayato Shikata and Yuto Shinoda for helpful discussions on the sorting protocol presented in Section 3 at the early stages of this work.

1 Introduction

President is a card game in which each player aims to be the first to dispose of all cards in their hand. In each round, the first player can play one to four cards of the same rank; thereafter, each player chooses either "play" the same number of cards of a strictly higher rank or "pass." When all players pass consecutively, the last player who played a card wins the round and becomes the first player of the next round. For example, suppose that the

[1] Corresponding Author

following pairs are played in a round:

$$[3♣][3♥] \to [5♣][5♠] \to [9♥][9♦] \to [J♥][J♣] \to [K♣][K♦] \to [2♥][2♦],$$

where the order of rank is: $3 < 4 < \cdots < 10 < \mathrm{J} < \mathrm{Q} < \mathrm{K} < \mathrm{A} < 2$. Then the player who plays the pair of 2 wins the round and becomes the first player of the next round. At the end of the game, the player who goes out first (resp. second) becomes the role of president (resp. vice-president), whereas the player who goes out last (resp. second-to-last) becomes the role of scum (resp. vice-scum). At the beginning of the next game, the scum (resp. vice-scum) must give their two cards (resp. single card) of highest rank to the president (resp. vice-president), and the players who receive cards from the bottom positions always hand back an equal number of cards that they do not want to hold.

President can be played with as few as two players; however, to make the game enjoyable, the game typically requires at least four players. When the number of players is insufficient, one natural approach is to introduce virtual players. The simplest implementation is to reveal the entire hand of each virtual player and to have the human players execute the virtual player's moves according to a predetermined strategy. However, revealing the virtual player's hand diminishes the uncertainty of the game and hence reduces its enjoyment. Therefore, we need to solve the *player simulation problem* to realize the moves of a virtual player while keeping its hand hidden.

Existing studies have proposed *card-based cryptographic protocols* simulating virtual players for Old Maid [30] and UNO [26]. Specifically, the protocol for UNO [26] allows a virtual player to play a single card from its hand, without revealing any other cards. This protocol can be applied to President; however, its output is restricted to a single-card play, and hence it does not support multi-card plays such as pairs, three of a kind, or four of a kind. If we implement a virtual player using the existing protocol supporting only single-card plays, the virtual player would be forced to pass whenever a multi-card play is required. This significantly weakens the virtual player and removes the original fun of the game.

1.1 Contribution

We propose a selection protocol for $k \in \{1, 2, 3, 4\}$, which selects k cards of a kind uniformly at random from all k cards of a kind in a virtual player's hand, whose rank is greater than the latest played cards. For example, if a virtual player's hand is

$$[3♥][3♦][4♣][5♣][5♥][7♣][7♥][10♦][Q♦][K♣][K♠]$$

and the latest played cards are $[5♦][5♠]$, then the protocol for $k = 2$ selects either $[7♣][7♥]$ or $[K♣][K♠]$ uniformly at random.

In addition to a standard deck of playing cards for playing President, we use *helping cards* to construct a protocol. The number of helping cards used in our protocol is $\max(\mathsf{card}_{\mathsf{lot}}, \mathsf{card}_{\mathsf{sort}}) + 52 \cdot 3$, where $\mathsf{card}_{\mathsf{lot}}$ and $\mathsf{card}_{\mathsf{sort}}$ denote the numbers of helping cards in the *covert lottery protocol* [33] and a *sorting protocol*, respectively, where a sorting protocol rearranges a sequence of face-down cards representing $x_1, x_2, \ldots, x_m \in \{0, 1\}$ in descending order without revealing any information. The number of shuffles used in our protocol is $R \cdot (\mathsf{shuf}_{\mathsf{sort}} + 1) + \mathsf{shuf}_{\mathsf{lot}} + 2$, where R denotes the number of ranks greater than the latest played cards, and $\mathsf{shuf}_{\mathsf{lot}}, \mathsf{shuf}_{\mathsf{sort}}$ denote the numbers of shuffles in the covert lottery protocol and a sorting protocol, respectively. Thus, the dominant parameter of the efficiency of our selection protocol is the efficiency of the underlying sorting protocol.

Table 1 The efficiency of secure sorting protocols described in terms of the following: m is the number of inputs; ℓ is the input bit-length; c is the number of comparators of the underlying sorting networks, where $c = O(m \log m)$. The values for [15] here are under the assumption that m is a power of two. The number of shuffles for [9] is an expected value.

Proposer	# Helping Cards	# Shuffles
[9]	$3m + \ell + 2$	$\ell(2m + 1 + (m + 1)\sum_{i=1}^{m} \frac{1}{i})$
[15]	$\ell + 20$	$5\ell c$
This work	$m + 2$	$2\ell m$

Another contribution of our work is to design an efficient sorting protocol. Table 1 summarizes a comparison between the existing sorting protocols [9, 15] and our sorting protocol in terms of the number of helping cards and the number of shuffles. By plugging our sorting protocol into our selection protocol, the resulting numbers of helping cards and shuffles are 160 and $6R + 3$, respectively, where R is the number of ranks greater than the latest played cards. Moreover, our selection protocol also supports the maximum or the minimum strategy, which selects the maximum or the minimum rank.

Our approach for President is broadly applicable to games in which a player may play multiple cards at once, and we expect it to serve as a milestone for future work on virtual-player simulation.

1.2 Related Work

This paper connects games with cryptography and falls within the line of research referred to as *fun with cryptography*.

A representative research theme in fun with cryptography is applying cryptographic techniques to puzzles. In particular, zero-knowledge proof protocols for pencil puzzles have been extensively studied for a decade. These are cryptographic protocols where a prover who knows a solution to a puzzle convinces a verifier that the prover indeed knows a solution, without revealing the solution itself. Protocols have been proposed for many puzzles, including Sudoku [8, 23, 24, 28, 34, 35], Kakuro [3, 19], Slitherlink [10, 17], Moon-or-Sun [10], Sumplete [12], Sukoro [27], Usowan [18], Zeiger [25] and so on. Another theme is to generate an instance of permutation puzzles such as the Rubik's Cube uniformly at random while keeping the generated instance hidden [29].

There are other research topics applying cryptography to games, to which our work also belongs. Typical topics include simulating virtual players [26, 31] and removing a game master [11, 13, 20]. In particular, research on simulating virtual players has so far been proposed only for Old Maid and UNO. Since this line of work is still in its early stage, even basic methods for simulating card plays have not been fully clarified.

Another important theme in fun with cryptography is to execute cryptographic protocols in an understandable and enjoyable manner. A representative area in this direction is card-based cryptography [1, 5, 21]. There has been research on representing differential privacy using card-based cryptography [7], as well as research on constructing private simultaneous messages using card-based cryptography [32]. Beyond card-based cryptography, many protocols have been proposed that implement cryptographic primitives using familiar physical tools, such as physical auction protocols [6] and physical ring signatures [2].

2 Preliminaries

We define the notations and introduce the existing protocols.

2.1 Notation

Card. President typically uses a standard deck of commonly available cards excluding jokers. Each card has a unique pair of rank in $\{\mathrm{A}, 2, \ldots, \mathrm{K}\}$ and suit in $\{\spadesuit, \diamondsuit, \clubsuit, \heartsuit\}$ and is denoted as follows:

$$\boxed{\mathrm{A}\spadesuit}\boxed{2\spadesuit}\boxed{3\spadesuit}\cdots\boxed{\mathrm{K}\spadesuit}\ \boxed{\mathrm{A}\diamondsuit}\boxed{2\diamondsuit}\boxed{3\diamondsuit}\cdots\boxed{\mathrm{K}\diamondsuit}\ \boxed{\mathrm{A}\clubsuit}\boxed{2\clubsuit}\boxed{3\clubsuit}\cdots\boxed{\mathrm{K}\clubsuit}\ \boxed{\mathrm{A}\heartsuit}\boxed{2\heartsuit}\boxed{3\heartsuit}\cdots\boxed{\mathrm{K}\heartsuit}.$$

Our protocol also uses a two-color deck of cards, $\boxed{\heartsuit}$ and $\boxed{\clubsuit}$. The backs of all cards are identical, denoted by $\boxed{?}$. A Boolean value is encoded with the order of two cards, i.e., $\boxed{\clubsuit}\boxed{\heartsuit} = 0$ and $\boxed{\heartsuit}\boxed{\clubsuit} = 1$. Such two cards encoding a bit $x \in \{0, 1\}$ is called a *commitment to* x and is denoted as follows:

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

Shuffling. To introduce a randomness, our protocol uses a *pile-scramble shuffle* [14]. Given m piles each consisting of the same number of cards, denoted by $(p_1, p_2, \ldots, p_m)$, it rearranges the order uniformly at random, i.e., for a random permutation r uniformly chosen from the symmetric group of degree m, denoted by S_m, the resulting order is $(p_{r(1)}, p_{r(2)}, \ldots, p_{r(m)})$:

$$\Big[\overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{p_1}\Big|\overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{p_2}\Big|\cdots\Big|\overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{p_m}\Big] \to \overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{p_{r(1)}}\ \overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{p_{r(2)}}\ \cdots\ \overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{p_{r(m)}},$$

where the application of a pile-scramble shuffle is denoted by $[\cdot|\cdots|\cdot]$. Note that the order of cards within each pile p_i unchanges. When $m = 2$, a pile-scramble shuffle is called a *random bisection cut* [22] in the literature.

2.2 Mizuki–Sone's AND Protocol

Given two commitments to $x, y \in \{0, 1\}$, the Mizuki–Sone's AND protocol [22] (MS-AND in short) produces a commitment to $x \wedge y$ as well as a commitment to $\overline{x} \wedge y$ without revealing anything:

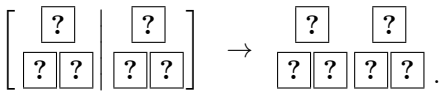
$$\underbrace{\boxed{?}\boxed{?}}_{x}\ \underbrace{\boxed{?}\boxed{?}}_{y}\ \boxed{\clubsuit}\boxed{\heartsuit} \to \cdots \to \underbrace{\boxed{?}\boxed{?}}_{x\wedge y}\ \underbrace{\boxed{?}\boxed{?}}_{\overline{x}\wedge y}\ \boxed{\clubsuit}\boxed{\heartsuit}.$$

We describe the procedure for MS-AND as follows.

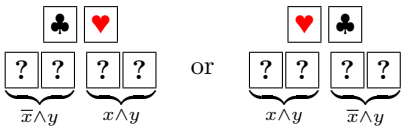
1. Place the two input commitments to x, y along with a commitment to 0 as follows:

$$\begin{array}{c}\underbrace{\boxed{?}\boxed{?}}_{x}\\ \underbrace{\boxed{?}\boxed{?}}_{y}\underbrace{\boxed{?}\boxed{?}}_{0}\end{array}.$$

2. Apply a pile-scramble shuffle as follows:

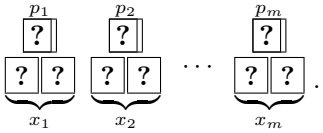


3. Reveal the above two cards and obtain the output as follows:



2.3 Covert Lottery Protocol

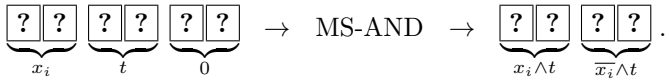
Shinoda et al. [33] presented a *covert lottery protocol* that selects a pile uniformly at random from all piles with a commitment to 1 while keeping the commitments hidden. Although the original covert lottery protocol selects a random pile among all piles if there is no commitment to 1, Ruangwises and Shinagawa [26] modified the protocol in such a way that no pile is selected if there is no commitment to 1. Hereafter, we consider the modified version and refer to it simply as the covert lottery protocol. It takes m commitments to $x_1, \ldots, x_m \in \{0, 1\}$ associated with m piles of the same number of cards $(p_1, p_2, \ldots, p_m)$ as input:



From these, we want to select a pile p_i uniformly at random among all piles p_i with $x_i = 1$. If there is no such pile, i.e., $x_i = 0$ for every i, $1 \leq i \leq m$, the protocol outputs $\perp$. For example, when $(p_1, \ldots, p_5)$ and $(x_1, \ldots, x_5) = (0, 1, 1, 0, 1)$, the protocol outputs a pile p_i uniformly at random from the set $\{p_2, p_3, p_5\}$.

The covert lottery protocol proceeds as follows.

1. Prepare a commitment to $t \coloneqq 1$ called a *token*.
2. Repeat the following for $i = 1, 2, \ldots, m$.
   a. Obtain commitments to $x_i \wedge t$ and $\overline{x_i} \wedge t$ using MS-AND [22] from the commitments to x_i and t (along with a commitment to 0):

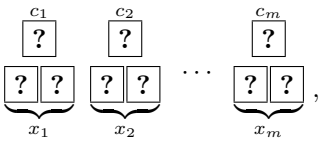


   b. Place the commitment to $y_i \coloneqq x_i \wedge t$ below p_i, and update the token by replacing it with the commitment to $\overline{x_i} \wedge t$.
3. After this loop, we have m piles, each consisting of p_i and y_i, where $y_i = x_i \wedge \bigwedge_{j<i} \overline{x_j}$. In particular, for any index i' with $y_{i'} = 1$, we have $x_{i'} = 1$, $x_j = 0$ for all $j < i'$, and $y_k = 0$ for all $k \neq i'$. Hence i' is the leftmost index such that $x_{i'} = 1$.
4. Apply a pile-scramble shuffle to the m piles.
5. Reveal the commitments to all y_i to find at most one commitment to 1. Select the pile above that commitment. If there is no commitment to 1, then output $\perp$.

The number of helping cards is four: the token used in Step 1 requires two cards  , and MS-AND in Step 2 requires two additional cards  .

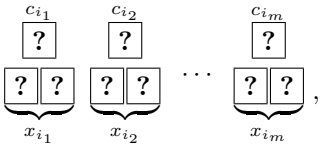
3 Sorting Protocol for Commitments

This section provides a sorting protocol for m commitments to $x_1, \ldots, x_m \in \{0, 1\}$ associated with some cards: Given



(1)

we want to have



such that $x_{i_j} \geq x_{i_{j+1}}$ for every j, $1 \leq j \leq m-1$.

3.1 Idea

Given a sequence of ♣s and ♥s of length m, namely $(s_1, \ldots, s_m) \in \{♣, ♥\}^m$, the following algorithm sorts it in descending order where ♣ < ♥.[2]

1. Set $i := 1$.
2. If the last element in the current sequence is ♣, then move it just before the i-th element; if the last element is ♥, then move it to the first position.
3. Set $i := i + 1$. If $i \leq m$, return to Step 2.

For example, a sequence ♥♣♥♣♥ is rearranged as:

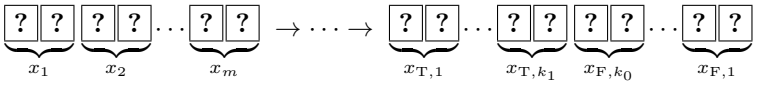
$$\overset{1\ 2\ 3\ 4\ 5}{♥♣♥♣♥} \to \overset{5\ 1\ 2\ 3\ 4}{♥♥♣♥♣} \to \overset{5\ 4\ 1\ 2\ 3}{♥♣♥♣♥} \to \overset{3\ 5\ 4\ 1\ 2}{♥♥♣♥♣} \to \overset{3\ 5\ 4\ 2\ 1}{♥♥♣♣♥} \to \overset{1\ 3\ 5\ 4\ 2}{♥♥♥♣♣}.$$

More precisely, let k_0 and k_1 be the numbers of ♣s and ♥s in $(s_1, \ldots, s_m)$, respectively, let $s_{♣,i}$ for each $i \in \{1, \ldots, k_0\}$ be the i-th element (counting from the left) among those having ♣, and let $s_{♥,i}$ for each $i \in \{1, \ldots, k_1\}$ be the i-th element among those having ♥; then, the above algorithm transforms $(s_1, \ldots, s_m)$ into

$$(s_{♥,1}, s_{♥,2}, \ldots, s_{♥,k_1}, s_{♣,k_0}, s_{♣,k_0-1}, \ldots, s_{♣,1}).$$

3.2 Basic Protocol

Assume m commitments to $(x_1, \ldots, x_m) \in \{0, 1\}^m$. Let k_0 and k_1 be the numbers of 0s and 1s in $(x_1, \ldots, x_m)$, respectively, let $x_{\mathrm{F},i}$ for each $i \in \{1, \ldots, k_0\}$ correspond to the i-th commitment (counting from the left) among those having 0 (False), and let $s_{\mathrm{T},i}$ for each $i \in \{1, \ldots, k_1\}$ correspond to the i-th commitment among those having 1 (True). We here provide a basic protocol for sorting the m commitments:

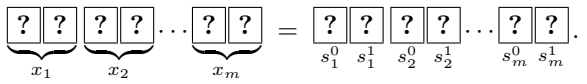


(2)

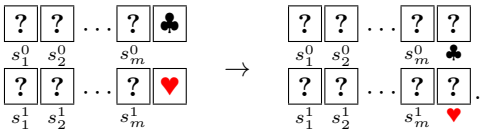
[2] This problem is sometimes called the "binary array sorting," and one can easily confirm that the PARTITION subroutine of QUICKSORT (see, e.g., [4]) solves it, although the algorithm here is different. Our algorithm is somewhat similar to the "two-way insertion" sorting (e.g., [16]).

That is, our basic protocol partitions the commitments such that the order of those having 1 is kept stable while the order of those having 0 is reversed.

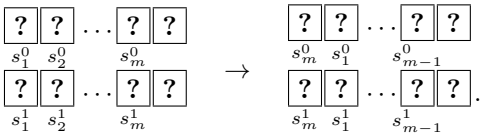
Given m commitments along with two helping cards $\clubsuit\heartsuit$, the protocol proceeds as follows, where $s_i^0, s_i^1 \in \{\clubsuit, \heartsuit\}$ for every i, $1 \le i \le m$, denote the first and second cards of the i-th commitment, respectively, i.e.,



1. Rearranging the cards to make two sequences of face-down cards as follows:

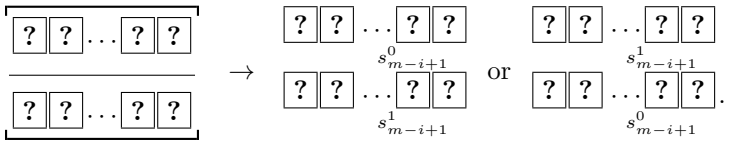


2. For each sequence, move the m-th card to the first position:



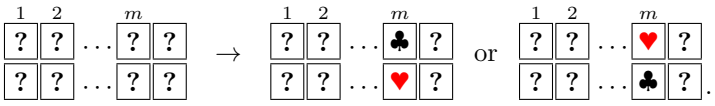
Set $i := 2$.

3. Apply a pile-scramble shuffle where the top and bottom sequences are regarded as piles:



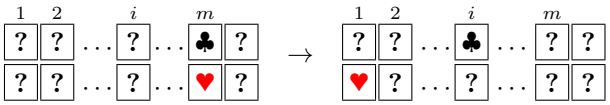
Note that the m-th cards are either s_{m-i+1}^0, s_{m-i+1}^1 or s_{m-i+1}^1, s_{m-i+1}^0 (from top to bottom) equally likely.

4. Reveal the m-th card of each sequence:

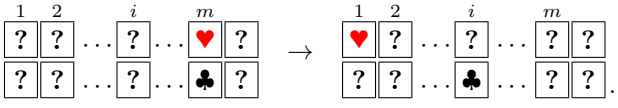


This means to reveal x_{m-i+1} or $\overline{x}_{m-i+1}$, each of which occurs with a probability of $1/2$ (and hence, no information leaks).

5. As in the algorithm explained in Section 3.1, move the revealed $\clubsuit$ just before the i-th card, and move the revealed $\heartsuit$ to the first position:

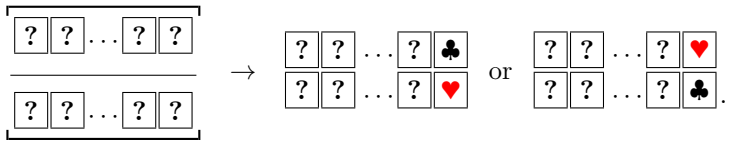


or



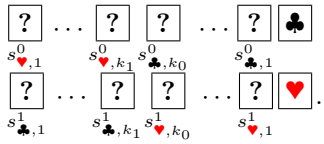
6. Turn over the two face-up cards. Set $i := i + 1$. If $i \le m$, return to Step 3.

7. Note that each of the top and bottom sequences (excluding the last cards) has been sorted now. Apply a pile-scramble shuffle and reveal the rightmost cards:

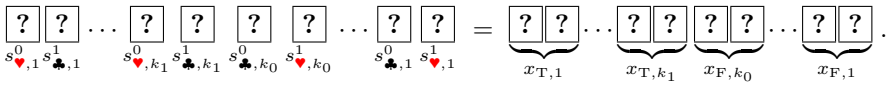


If the revealed card in the top sequence is ♥, swap the top and bottom sequences. Then, in either case, the top sequence (excluding the last card) is $(s^0_{\heartsuit,1}, \ldots, s^0_{\heartsuit,k_1}, s^0_{\clubsuit,k_0}, \ldots, s^0_{\clubsuit,1})$ and the bottom sequence is $(s^1_{\heartsuit,1}, \ldots, s^1_{\heartsuit,k_0}, s^1_{\clubsuit,k_1}, \ldots, s^1_{\clubsuit,1})$, where $s^0_{\clubsuit,1}, s^0_{\heartsuit,1}, \ldots$ and $s^1_{\clubsuit,1}, s^1_{\heartsuit,1}, \ldots$ are defined in a similar manner to Section 3.1.

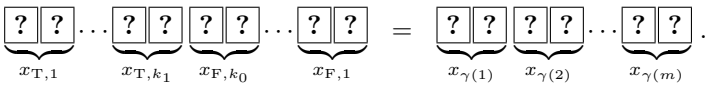
8. Reverse the order of the first m cards in the bottom sequence; then, we have



9. Rearrange the cards (excluding the rightmost ones) as follows:

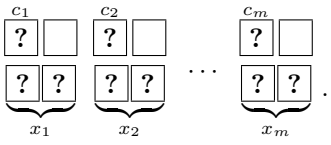


Thus, this protocol uses m shuffles to establish the sorting transition in Eq. (2). Hereinafter, by $\gamma_{(x_1,\ldots,x_m)} \in S_m$ or simply by $\gamma \in S_m$, we denote the permutation corresponding to the sorting:

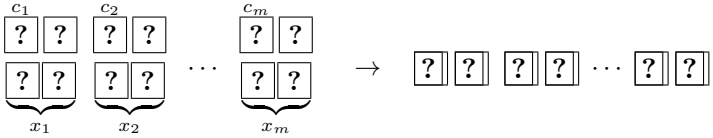


3.3 Sorting Protocols

Next, let us extend the basic protocol so that a sequence shown in Eq. (1) can be sorted. Given a sequence shown in Eq. (1), we use m helping □s[3] as "padding cards," as follows:

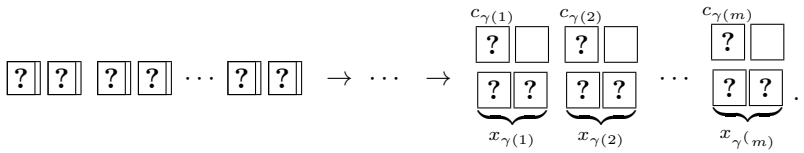


After turning over the padding cards, we move the top sequence below the bottom sequence so that we have $2m$ two-card piles:



[3] Instead of □s, the padding cards could be ♣s.

We apply the procedure of the basic protocol presented in Section 3.2 to the above $2m$ piles (where we consider a two-card pile as a single card) along with two helping cards, and turn over the padding cards; then, we obtain:



Thus, this protocol uses $m+2$ helping cards and m shuffles. While only one card is attached to each commitment in Eq. (1), the number of attached cards to each commitment can be arbitrary. Furthermore, if the number of attached cards to each commitment is an even number, then we do not need any padding cards, and hence, the protocol can be executed with only two helping cards.

Note that if we apply the above protocol twice, we obtain the *stable sorting* because the order of those having 0 (False) will be reversed:

$$(x_{\mathrm{T},1} \cdots x_{\mathrm{T},k_1}, x_{\mathrm{F},k_0} \ldots x_{\mathrm{F},1}) \quad \to \quad (x_{\mathrm{T},1} \cdots x_{\mathrm{T},k_1}, x_{\mathrm{F},1} \ldots x_{\mathrm{F},k_0}).$$

Therefore, we can have a stable-sort version protocol with $m+2$ helping cards and $2m$ shuffles.

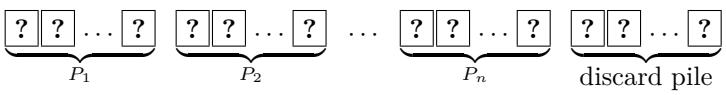
Applying the stable-sort version protocol, we can construct a sorting protocol for multi-bit (i.e., ℓ-bit) inputs: Given m ℓ-bit inputs, each of which consists of ℓ commitments, along with attached cards, we can sort the sequence digit by digit based on radix sort. Such a protocol requires $m+2$ helping cards and $2\ell m$ shuffles.

4 Selection Protocol for Multiple Cards of a Kind

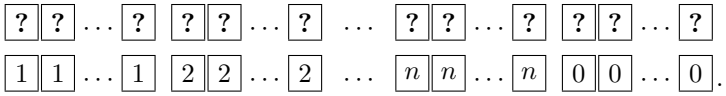
Let $P_1, P_2, \ldots, P_n$ be n players, each either real or virtual. We now want to simulate a virtual player P's action, where P is one of $P_1, P_2, \ldots, P_n$. We propose a selection protocol for k cards of a kind for $k \in \{1,2,3,4\}$, which selects k cards of a kind in P's hand, whose rank is greater or equal to a given rank r. Our protocol admits three strategies: rand, min, and max. When the strategy is rand, the protocol selects k cards of a kind uniformly at random from all k cards of a kind in P's hand. Similarly, when the strategy is min (resp. max), it selects k cards of a kind of lowest (resp. highest) rank from all k cards of a kind in P's hand.

Suppose a rank $r \in \{3,4,\ldots,\mathrm{K},\mathrm{A},2\}$, the number of cards $k \in \{1,2,3,4\}$, and a strategy $s \in \{\mathrm{rand},\mathrm{min},\mathrm{max}\}$ are given. Our selection protocol for (r,k,s) proceeds as follows.

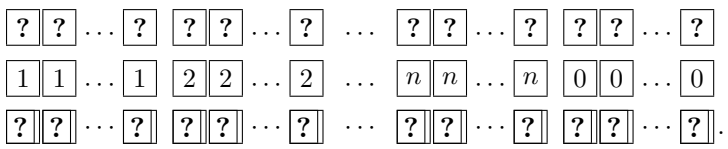
1. Place all cards in a horizontal line as follows:



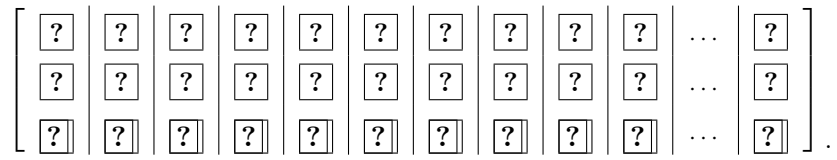
   Here, the virtual player P is one of $P_1, P_2, \ldots, P_n$. It does not matter whether the other players are real or virtual.
2. Place $\boxed{i}$ on the bottom of the P_i's cards $(1 \le i \le n)$ and $\boxed{0}$ on the bottom of the discard pile as follows:



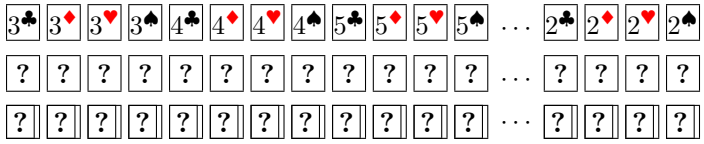
3. Place a commitment to 1 (♥♣) on the bottom of P's cards and a commitment to 0 (♣♥) on the bottom of the other cards as follows:



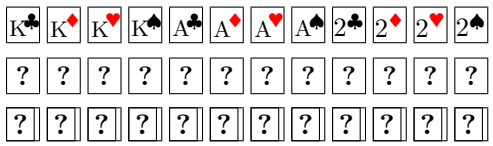
4. Turn all cards face-down and apply a pile-scramble shuffle as follows:



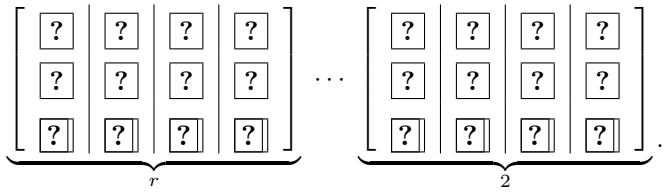
5. Open the topmost row and sort the piles as follows:



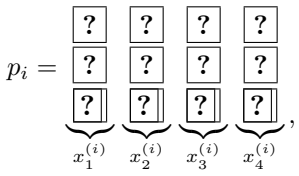
Put aside columns of smaller rank than r. The following shows the case of $r = \mathrm{K}$:



6. Turn all cards face-down and apply a pile-scramble shuffle to each four piles of the same number as follows:



7. For each $i \in \{r, \ldots, \mathrm{A}, 2\}$, apply our sorting protocol (presented in Section 3.3) to the four piles of $\boxed{i}$ according to the commitments in the bottom row. Now we have a sequence of piles $(p_r, \ldots, p_{\mathrm{A}}, p_2)$. Let $x_j^{(i)} \in \{0, 1\}$ be the value of the j-th commitment (counting from the left) of p_i as follows:



where $x_1^{(i)} \geq x_2^{(i)} \geq x_3^{(i)} \geq x_4^{(i)}$. We call the commitment to $x_k^{(i)}$ the *leading commitment* of p_i. We note that the value of $x_k^{(i)}$ determines whether P has at least k cards of rank i or not. Let q_i be the pile obtained from p_i by removing its leading commitment.

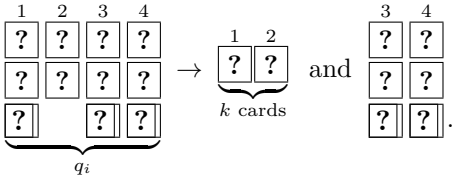
8. Execute the following procedures according to the strategy s:

Case min: Open the leading commitment of p_i in ascending order of rank, i.e., $i = r, \ldots, \mathrm{A}, 2$, until the commitment to 1 is opened. If p_i is the first pile having 1, go to Step 9. If all opened values are 0, then announce that "there is no k cards of a kind in the P's hand" and go to Step 10.

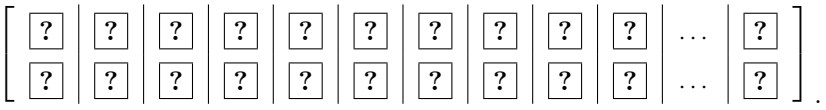
Case max: Open the leading commitment of p_i in descending order of rank, i.e., $i = 2, \mathrm{A}, \ldots, r$, until the commitment to 1 is opened. If p_i is the first pile having 1, go to Step 9. If all opened values are 0, then announce that "there is no k cards of a kind in the P's hand" and go to Step 10.

Case rand: Apply a covert lottery protocol for $(q_r, \ldots, q_{\mathrm{A}}, q_2)$ with their leading commitments. If it outputs q_i, then go to Step 9. Otherwise, go to Step 10.

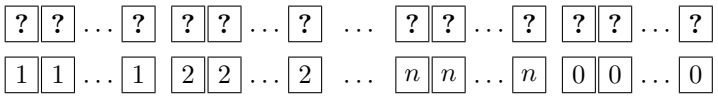
9. Determine $\{1, \ldots, k\}$-th cards in the topmost row of q_i as the "k cards of a kind" and remove $\{1, \ldots, k\}$-th columns of q_i as follows:



10. Return the columns put aside in Step 5, and remove the bottom row from all piles and apply a pile-scramble shuffle as follows:



11. Open the bottom row and sort the piles as follows:



Return the cards to P_i's hand and discard pile, respectively. Output the k cards if Step 9 is executed, and return $\perp$ otherwise.

The number of helping cards is $\mathsf{card}_{\mathsf{select}} = \max(\mathsf{card}_{\mathsf{sort}}, \mathsf{card}_{\mathsf{lot}}) + 52 \cdot 3$, where $\mathsf{card}_{\mathsf{sort}}$ and $\mathsf{card}_{\mathsf{lot}}$ are the numbers of helping cards of our sorting protocol in Step 7 and of the covert lottery protocol in Step 8. Since our sorting protocol requires 2 helping cards and the covert lottery protocol requires 4 helping cards, we have $\mathsf{card}_{\mathsf{select}} = 160$. The number of shuffles of the protocol $\mathsf{shuf}_{\mathsf{select}}$ is given as follows:

$$\mathsf{shuf}_{\mathsf{select}} = \begin{cases} R \cdot (\mathsf{shuf}_{\mathsf{sort}} + 1) + 2 & \text{if } s \in \{\min, \max\}, \\ R \cdot (\mathsf{shuf}_{\mathsf{sort}} + 1) + \mathsf{shuf}_{\mathsf{lot}} + 2 & \text{if } s = \mathrm{rand}, \end{cases}$$

where $R := |\{r, r+1, \ldots, A, 2\}|$ is the number of ranks greater than or equal to r, and $\mathsf{shuf}_{\mathsf{sort}}, \mathsf{shuf}_{\mathsf{lot}}$ are the numbers of shuffles of our sorting protocol in Step 7 and of the covert lottery protocol in Step 8, respectively. Since $\mathsf{shuf}_{\mathsf{sort}} = 4$ and $\mathsf{shuf}_{\mathsf{lot}} = R + 1$, we have $\mathsf{shuf}_{\mathsf{select}} = 5R + 2$ if $s \in \{\min, \max\}$ and $6R + 3$ if $s = \mathrm{rand}$.

5 Playing President with Virtual Players

We show how to play President with virtual players using our card-based protocols described in the preceding sections. In the following, we describe how to simulate the virtual player P's action. For the simulation of P, we need to choose the strategy of P. In particular, we need to choose three parameters (s_1, s_2, s_3) each from the following list:

- $s_1 \in \{\min, \mathrm{rand}\}$ is the parameter for selecting cards to scum and (vice-)scum.
- $s_2 \in \{\min, \max, \mathrm{rand}\}$ is the parameter for playing cards in a round.
- $s_3 \in \{\min, \max, \mathrm{rand}\} \times \mathbb{D}$ is the parameter for the first play, where $\mathbb{D}$ is the whole set of probability distribution over $\{1, 2, 3, 4\}$.

These parameters may differ for each virtual player. Even for the same virtual player, the parameters may change depending on the game's progress.

After dealing cards. The cards are dealt in the usual way. After dealing cards, each virtual player P's action is simulated as follows.

- If P is scum (resp. vice-scum), then select the two (resp. one) highest cards from P's hand using the selection protocol for $(3, 1, \max)$ twice (resp. once) and transfer them to president (resp. vice-president).
- If P is president (resp. vice-president), then select two (resp. one) cards according to the parameter s_1 as follows:
  - **Case min:** Execute the selection protocol for $(3, 1, \min)$, and pass the cards to scum (resp. vice-scum).
  - **Case rand:** Choose the cards uniformly at random from P's hand, and pass them to scum (resp. vice-scum).

The play in a round. The P's play in a round is simulated as follows.

- Suppose that P is not the first player in the round and that k cards of rank r have just been played before P's turn. If P's hand is less than k cards, then P's turn is passed. Otherwise, execute the selection protocol for $(r + 1, k, s_2)$, and play the cards (or pass if it outputs $\perp$).
- Suppose that P is the first player in this round. Then P's action is simulated according to $s_3 = (s'_3, \mathcal{D})$ as follows: First, choose $k \in \{1, 2, 3, 4\}$ according to the distribution $\mathcal{D}$. Then, execute the selection protocol for $(3, k, s'_3)$ cards of a kind. If it outputs k cards of a kind, play them. Otherwise, update $k \leftarrow k - 1$ and then execute the selection protocol for $(3, k, s'_3)$ until the cards are found[4].

A major difference between a human player and a virtual player P is that, while a human player may be able to play but still choose to pass strategically, P cannot behave in this way. Consequently, if P passes for k cards of a kind of rank r, the information that "P does not hold k cards of a kind of rank greater than r" is inevitably revealed. One possible mitigation is to allow the virtual player P to pass probabilistically even when it is able to play. In particular, it suffices to set the commitments placed under P's cards in Step 3 of the selection protocol to be 1 with probability $1 - \epsilon$ and 0 with probability ϵ. Then, even if P passes in a given situation, it does not mean that P will always pass under the same situation in the future, thereby the uncertainty of the game is not compromised.

[4] In this case, the fact that P does not possess k cards of a kind is leaked, although this information can be obfuscated using the mitigation mentioned below.

6 Conclusion

In this paper, we studied the player-simulation problem for President. In particular, we proposed a selection protocol for k cards of a kind, which enables to simulate multi-card plays. Our construction relies on secure sorting, and hence the overall efficiency is dominated by the underlying sorting protocol. Motivated by this, we proposed an efficient sorting protocol.

President has many variants worldwide, e.g., "Daifugo" in Japan or also "Asshole" in the U.S., and it also admits various local rules; thus, it remains an important direction for future work to study the player-simulation problem under a wide range of rule sets. Moreover, to make player-simulation protocols practically playable and enjoyable, it is necessary to further reduce the numbers of shuffles and cards. Another key challenge is to enhance the strategic behavior of virtual players, or more fundamentally, to identify what kinds of strategies make the game enjoyable for human players. Addressing this last question may require user studies for cognitive-psychological investigations.

References

1. Bert Den Boer. More efficient match-making and satisfiability *The Five Card Trick*. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology – EUROCRYPT' 89*, volume 434 of *LNCS*, pages 208–217, Heidelberg, 1990. Springer. URL: https://doi.org/10.1007/3-540-46885-4_23.
2. Xavier Bultel. Physical ring signature. In Andrei Z. Broder and Tami Tamir, editors, *Fun with Algorithms*, volume 291 of *LIPIcs*, pages 7:1–7:18. Schloss Dagstuhl, 2024.
3. Xavier Bultel, Jannik Dreier, Jean-Guillaume Dumas, and Pascal Lafourcade. Physical zero-knowledge proofs for Akari, Takuzu, Kakuro and KenKen. In Erik D. Demaine and Fabrizio Grandoni, editors, *Fun with Algorithms*, volume 49 of *LIPIcs*, pages 8:1–8:20, Dagstuhl, Germany, 2016. Schloss Dagstuhl. URL: https://doi.org/10.4230/LIPIcs.FUN.2016.8.
4. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 3rd edition, 2009.
5. Claude Crépeau and Joe Kilian. Discreet solitary games. In Douglas R. Stinson, editor, *Advances in Cryptology—CRYPTO' 93*, volume 773 of *LNCS*, pages 319–330, Berlin, Heidelberg, 1994. Springer. URL: https://doi.org/10.1007/3-540-48329-2_27.
6. Jannik Dreier, Hugo Jonker, and Pascal Lafourcade. Secure auctions without cryptography. In Alfredo Ferro, Fabrizio Luccio, and Peter Widmayer, editors, *Fun with Algorithms*, volume 8496 of *LNCS*, pages 158–170. Springer, 2014.
7. Reo Eriguchi, Kazumasa Shinagawa, and Takao Murakami. Card-based cryptography meets differential privacy. In Andrei Z. Broder and Tami Tamir, editors, *Fun with Algorithms*, volume 291 of *LIPIcs*, pages 12:1–12:20, Dagstuhl, Germany, 2024. Schloss Dagstuhl. URL: https://doi.org/10.4230/LIPIcs.FUN.2024.12.
8. Ronen Gradwohl, Moni Naor, Benny Pinkas, and Guy N. Rothblum. Cryptographic and physical zero-knowledge proof systems for solutions of Sudoku puzzles. In Pierluigi Crescenzi, Giuseppe Prencipe, and Geppino Pucci, editors, *Fun with Algorithms*, volume 4475 of *LNCS*, pages 166–182, Berlin, Heidelberg, 2007. Springer. URL: https://doi.org/10.1007/978-3-540-72914-3_16.
9. Rikuo Haga, Kodai Toyoda, Yuto Shinoda, Daiki Miyahara, Kazumasa Shinagawa, Yuichi Hayashi, and Takaaki Mizuki. Card-based secure sorting protocol. In Chen-Mou Cheng and Mitsuaki Akiyama, editors, *Advances in Information and Computer Security*, volume 13504 of *LNCS*, pages 224–240, Cham, 2022. Springer. URL: https://doi.org/10.1007/978-3-031-15255-9_12.
10. Samuel Hand, Alexander Koch, Pascal Lafourcade, Daiki Miyahara, and Léo Robert. Efficient card-based ZKP for single loop condition and its application to Moon-or-Sun. *New Gener. Comput.*, 42:449–477, 2024. URL: https://doi.org/10.1007/s00354-024-00274-1.

11 Yuji Hashimoto, Kazumasa Shinagawa, Koji Nuida, Masaki Inamura, and Goichiro Hanaoka. Secure grouping protocol using a deck of cards. In Junji Shikata, editor, *Information Theoretic Security*, volume 10681 of *LNCS*, pages 135–152, Cham, 2017. Springer. URL: https://doi.org/10.1007/978-3-319-72089-0_8.

12 Kyosuke Hatsugai, Suthee Ruangwises, Kyoichi Asano, and Yoshiki Abe. NP-completeness and physical zero-knowledge proofs for Sumplete, a puzzle generated by ChatGPT. *New Gener. Comput.*, 42:429–448, 2024. URL: https://doi.org/10.1007/s00354-024-00267-0.

13 Shota Ikeda and Kazumasa Shinagawa. How to play Mastermind without game master. In Min Li, Mingji Xia, and Peng Zhang, editors, *Theory and Applications of Models of Computation*, volume 16084 of *LNCS*, pages 109–120, Singapore, 2026. Springer. URL: https://doi.org/10.1007/978-981-95-4839-2_9.

14 Rie Ishikawa, Eikoh Chida, and Takaaki Mizuki. Efficient card-based protocols for generating a hidden random permutation without fixed points. In Cristian S. Calude and Michael J. Dinneen, editors, *Unconventional Computation and Natural Computation*, volume 9252 of *LNCS*, pages 215–226, Cham, 2015. Springer. URL: https://doi.org/10.1007/978-3-319-21819-9_16.

15 Kota Kato, Takeshi Nakai, and Koutarou Suzuki. Card-based secure sorting protocols based on the sorting networks. In *Advanced Informatics: Concept, Theory and Application (ICAICTA)*, pages 1–6, NY, 2024. IEEE. URL: https://doi.org/10.1109/ICAICTA63815.2024.10763066.

16 Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching*. Addison-Wesley Professional, Reading, Massachusetts, 2nd edition, 1998.

17 Pascal Lafourcade, Daiki Miyahara, Takaaki Mizuki, Léo Robert, Tatsuya Sasaki, and Hideaki Sone. How to construct physical zero-knowledge proofs for puzzles with a "single loop" condition. *Theor. Comput. Sci.*, 888:41–55, 2021. URL: https://doi.org/10.1016/j.tcs.2021.07.019.

18 Daiki Miyahara, Léo Robert, Pascal Lafourcade, and Takaaki Mizuki. ZKP protocols for Usowan, Herugolf, and Five Cells. *Tsinghua Science and Technology*, 29(6):1651–1666, 2024. URL: https://doi.org/10.26599/TST.2023.9010153.

19 Daiki Miyahara, Tatsuya Sasaki, Takaaki Mizuki, and Hideaki Sone. Card-based physical zero-knowledge proof for Kakuro. *IEICE Trans. Fundam.*, 102(9):1072–1078, 2019. URL: https://doi.org/10.1587/transfun.E102.A.1072.

20 Takaaki Mizuki, Tomoki Kuzuma, Tomoya Hirano, Ririn Oshima, and Momofuku Yasuda. Gakmoro: An application of physical secure computation to card game. In *Unconventional Computation and Natural Computation*, volume 16364 of *LNCS*, pages 344–360, Cham, 2025. Springer. URL: https://doi.org/10.1007/978-3-032-15641-9_23.

21 Takaaki Mizuki and Hiroki Shizuya. A formalization of card-based cryptographic protocols via abstract machine. *Int. J. Inf. Secur.*, 13(1):15–23, 2014. URL: https://doi.org/10.1007/s10207-013-0219-4.

22 Takaaki Mizuki and Hideaki Sone. Six-card secure AND and four-card secure XOR. In Xiaotie Deng, John E. Hopcroft, and Jinyun Xue, editors, *Frontiers in Algorithmics*, volume 5598 of *LNCS*, pages 358–369, Berlin, Heidelberg, 2009. Springer. URL: https://doi.org/10.1007/978-3-642-02270-8_36.

23 Tomoki Ono, Suthee Ruangwises, Yoshiki Abe, Kyosuke Hatsugai, and Mitsugu Iwamoto. Single-shuffle physical zero-knowledge proof for sudoku using interactive inputs. In Keita Emura and Hiraku Morita, editors, *ACM ASIA Public-Key Cryptography Workshop*, New York, 2025. ACM.

24 Suthee Ruangwises. Two standard decks of playing cards are sufficient for a ZKP for Sudoku. *New Gener. Comput.*, 40:49–65, 2022. URL: https://doi.org/10.1007/s00354-021-00146-y.

25 Suthee Ruangwises. NP-completeness and physical zero-knowledge proofs for Zeiger. In Shin ichi Nakano and Mingyu Xiao, editors, *WALCOM: Algorithms and Computation*, volume 15411 of *LNCS*, pages 312–325, Singapore, 2025. Springer. URL: https://doi.org/10.1007/978-981-96-2845-2_20.

26 Suthee Ruangwises and Kazumasa Shinagawa. Simulating virtual players for UNO without computers. In *Unconventional Computation and Natural Computation*, LNCS, Cham, 2025. Springer. To Appear.

27 Shun Sasaki and Kazumasa Shinagawa. Physical zero-knowledge proof for Sukoro. *New Gener. Comput.*, 42:381–398, 2024. URL: https://doi.org/10.1007/s00354-024-00271-4.

28 Tatsuya Sasaki, Takaaki Mizuki, and Hideaki Sone. Card-based zero-knowledge proof for Sudoku. In Hiro Ito, Stefano Leonardi, Linda Pagli, and Giuseppe Prencipe, editors, *Fun with Algorithms*, volume 100 of *LIPIcs*, pages 29:1–29:10, Dagstuhl, Germany, 2018. Schloss Dagstuhl. URL: https://doi.org/10.4230/LIPIcs.FUN.2018.29.

29 Kazumasa Shinagawa, Kazuki Kanai, Kengo Miyamoto, and Koji Nuida. How to covertly and uniformly scramble the 15 puzzle and rubik's cube. In Andrei Z. Broder and Tami Tamir, editors, *Fun with Algorithms*, volume 291 of *LIPIcs*, pages 30:1–30:15, Dagstuhl, Germany, 2024. Schloss Dagstuhl. URL: https://doi.org/10.4230/LIPIcs.FUN.2024.30.

30 Kazumasa Shinagawa, Daiki Miyahara, and Takaaki Mizuki. How to play Old Maid with virtual players. *Theory of Computing Systems*, 69(1):13, 2025. URL: https://doi.org/10.1007/s00224-024-10203-w.

31 Kazumasa Shinagawa, Daiki Miyahara, and Takaaki Mizuki. How to play old maid with virtual players. In Bo Li, Minming Li, and Xiaoming Sun, editors, *Frontiers of Algorithmics*, volume 14752 of *LNCS*, pages 53–65, Singapore, 2025. Springer. URL: https://doi.org/10.1007/978-981-97-7752-5_4.

32 Kazumasa Shinagawa and Koji Nuida. Card-based protocols imply PSM protocols. In Olaf Beyersdorff, Michał Pilipczuk, Elaine Pimentel, and Nguyên Kim Thàng, editors, *Theoretical Aspects of Computer Science*, volume 327 of *LIPIcs*, pages 72:1–72:18, Dagstuhl, 2025. Schloss Dagstuhl. URL: https://doi.org/10.4230/LIPIcs.STACS.2025.72.

33 Yuto Shinoda, Daiki Miyahara, Kazumasa Shinagawa, Takaaki Mizuki, and Hideaki Sone. Card-based covert lottery. In Diana Maimut, Andrei-George Oprina, and Damien Sauveron, editors, *Innovative Security Solutions for Information Technology and Communications*, volume 12596 of *LNCS*, pages 257–270, Cham, 2021. Springer. URL: https://doi.org/10.1007/978-3-030-69255-1_17.

34 Kodai Tanaka and Takaaki Mizuki. Two UNO decks efficiently perform zero-knowledge proof for Sudoku. In Henning Fernau and Klaus Jansen, editors, *Fundamentals of Computation Theory*, volume 14292 of *LNCS*, pages 406–420, Cham, 2023. Springer. URL: https://doi.org/10.1007/978-3-031-43587-4_29.

35 Kodai Tanaka, Shun Sasaki, Kazumasa Shinagawa, and Takaaki Mizuki. Only two shuffles perform card-based zero-knowledge proof for Sudoku of any size. In *2025 Symposium on Simplicity in Algorithms (SOSA)*, pages 94–107. SIAM, 2025. URL: https://doi.org/10.1137/1.9781611978315.7.