Two Standard Decks of Playing Cards Are Sufficient for a ZKP for Sudoku

Suthee Ruangwises$^{(\boxtimes)}$

Department of Mathematical and Computing Science,
Tokyo Institute of Technology, Tokyo, Japan

Abstract. Sudoku is a logic puzzle with an objective to fill a number between 1 and 9 into each empty cell of a 9×9 grid such that every number appears exactly once in each row, each column, and each 3×3 block. In 2020, Sasaki et al. proposed a physical zero-knowledge proof (ZKP) protocol for Sudoku using 90 cards, which allows a prover to physically show that he/she knows a solution without revealing it. However, their protocol requires nine identical copies of some cards, which cannot be found in a standard deck of playing cards (with 52 different cards and two jokers). Therefore, nine identical decks are actually required in order to perform that protocol. In this paper, we propose a new ZKP protocol for Sudoku that can be performed using only two standard decks of playing cards. In general, we develop the first ZKP protocol for an $n \times n$ Sudoku that can be performed using a deck of all different cards.

Keywords: Zero-knowledge proof · Card-based cryptography · Sudoku · Puzzle

1 Introduction

Sudoku is one of the world's most popular logic puzzles. A Sudoku puzzle consists of a 9×9 grid divided into nine blocks of size 3×3. Some of the cells in the grid are already filled with numbers between 1 and 9. The player has to fill a number into each empty cell such that every number from 1 to 9 appears exactly once in each row, each column, and each 3×3 block [18] (see Fig. 1). There is also a generalized version of Sudoku where the grid has size $n \times n$ and is divided into n blocks of size $\sqrt{n} \times \sqrt{n}$, where n is a perfect square. The generalized Sudoku is known to be NP-complete [25].

1.1 Zero-Knowledge Proof

We want to construct a *zero-knowledge proof (ZKP)* for Sudoku, which allows a prover P to convince a verifier V that he/she knows a solution of the puzzle without revealing any information about it. Formally, a ZKP is an interactive

		7			1		8	
5	9			2		6	1	3
1		5	4		6		3	
3	6			1				
		6			5	4		
	5				2	3	7	
			3	7	4			1

6	2	7	5	3	1	9	8	4
8	1	3	6	4	9	2	5	7
5	9	4	8	2	7	6	1	3
1	7	5	4	9	6	8	3	2
3	6	9	2	1	8	7	4	5
2	4	8	7	5	3	1	9	6
7	3	6	1	8	5	4	2	9
4	5	1	9	6	2	3	7	8
9	8	2	3	7	4	5	6	1

Fig. 1. An example of a 9×9 Sudoku puzzle (left) and its solution (right)

proof between P and V where both of them are given a computational problem x, but only P knows a solution w. A ZKP with perfect completeness and perfect soundness must satisfy the following properties.

1. **Perfect Completeness:** If P knows w, then V always accepts.
2. **Perfect Soundness:** If P does not know w, then V always rejects.
3. **Zero-knowledge:** V does not obtain any information about w. Formally, there exists a probabilistic polynomial time algorithm S (called a *simulator*) that does not know w, and the outputs of S follow the same probability distribution as the outputs of the actual protocol.

The concept of a ZKP was first introduced by Goldwasser et al. [5]. Recently, many results have been focusing on constructing physical ZKPs using objects found in everyday life such as a deck of cards. These protocols have a benefit that they do not require electronic devices, and also have didactic values since they are easy to understand and verify the correctness, even for non-experts in cryptography.

2   Previous Protocols

The first ZKP protocols for Sudoku were developed by Gradwohl et al. [6] in 2009. However, each of their six proposed protocols either has a nonzero soundness error or requires special tools such as scratch-off cards. In 2020, Sasaki et al. [24] proposed the improved ZKP protocols for Sudoku that have perfect soundness without using special tools.

2.1   Uniqueness Verification Protocol

Before showing the protocol of Sasaki et al., we first explain the following subprotocol, which was also developed by the same authors [24]. This protocol allows the prover P to convince the verifier V that a sequence σ of n face-down cards

is a permutation of different cards $a_1, a_2, ..., a_n$ in some order, without revealing their orders. It also preserves the orders of the cards in σ (so that the sequence can be later used in other protocols).

Let $x_1, x_2, ..., x_n$ be another set of n different cards. P performs the following steps.
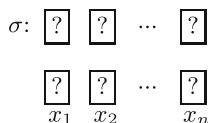
Fig. 2. A $2 \times n$ matrix constructed in Step 1

1. Publicly place face-down cards $x_1, x_2, ..., x_n$ below the face-down sequence σ in this order from left to right to form a $2 \times n$ matrix of cards (see Fig. 2).
2. Rearrange all columns of the matrix by a uniformly random permutation. This can be performed in real world by putting both cards in each column into an envelope and scrambling all envelopes together.
3. Turn over all cards in the top row. V verifies that the sequence is a permutation of $a_1, a_2, ..., a_n$. Otherwise, V rejects.
4. Turn over all face-up cards. Rearrange all columns of the matrix by a uniformly random permutation.
5. Turn over all cards in the bottom row. Rearrange the columns such that the cards in the bottom rows are $x_1, x_2, ..., x_n$ in this order from left to right. The sequence in the top row now returns to its original state.

2.2   Protocol of Sasaki et al.

Sasaki et al. [24] proposed three protocols to verify a solution of an $n \times n$ Sudoku puzzle. Here we will show only the first protocol, which is the one using the least number of cards.

Each card used in this protocol has a positive number on the front side $\boxed{1}$, $\boxed{2}$, ...; all cards have identical back sides $\boxed{?}$. On each cell already having a number j, P publicly places a face-down $\boxed{j}$. On each empty cell having a number j is P's solution, P secretly places a face-down $\boxed{j}$.

P then applies the uniqueness verification protocol to verify that every row, column, and block contains a permutation of $\boxed{1}$, $\boxed{2}$, ..., $\boxed{n}$.

In total, this protocol uses $n^2 + n$ cards: n identical copies of $\boxed{1}$, $\boxed{2}$, ..., $\boxed{n}$ (to encode the numbers in the grid), and another set of n different cards (to use in the uniqueness verification protocol). For a standard 9×9 puzzle, the protocol uses 90 cards, which is less than the number of cards in two standard decks; however, it requires nine identical copies of $\boxed{1}$, $\boxed{2}$, ..., $\boxed{9}$. As a standard deck consists of 54 different cards (including two different jokers), nine identical decks

are actually required in order to perform this protocol, which are too many to be practical. Another choice is to use a different kind of deck (e.g. cards from board games) that includes several identical copies of some cards, but these decks are more difficult to find in everyday life.

Considering a drawback of this protocol, we aim to develop a more practical ZKP protocol for a 9×9 Sudoku that can be performed using only two standard decks of playing cards.

2.3   Related Work

After the discovery of the physical ZKP protocols for Sudoku, physical ZKP protocols for other popular logic puzzles have been proposed as well, including Nonogram [3], Akari [1], Takuzu [1,14], Kakuro [1,15], KenKen [1], Makaro [2], Norinori [4], Slitherlink [12], Juosan [14], Numberlink [21], Suguru [20], Ripple Effect [22], Nurikabe [19], Hitori [19], and Bridges [23].

Besides verifying solutions of logic puzzles, card-based protocols have also been extensively studied in secure multi-party computation, a setting where multiple parties want to jointly compute a function of their secret inputs without revealing the input of any party. The vast majority of work in this area, however, also uses identical copies of ♣ and ♡ in the protocols. The only exceptions are [9,11,16,17] which introduced AND, XOR, and copy protocols using a standard deck, and [13] which introduced a Yao's millionaire protocol using a standard deck. In [11], the authors also posed an open problem to develop ZKP protocols for logic puzzles using a standard deck.

Pratically, a standard deck of playing cards consists of 54 different cards (including two different jokers). Theoretically, it is also a challenging problem to develop a protocol that uses a deck of all different cards, so we also study the setting where the deck consists of 1, 2, ... where each card can have an arbitrarily large number on it.

3   Our Contribution

In this paper, we propose a new ZKP protocol for a generalized $n \times n$ Sudoku puzzle with perfect completeness and soundness using a set of all different cards.

There are two slightly different methods to implement our protocol. The first one uses $n^2 + n\sqrt{n} + n + \sqrt{n}$ cards and $4n\sqrt{n}$ shuffles. The second one uses $n^2 + 2n + 3\sqrt{n}$ cards and at most $2n^2(\sqrt{n} - 1) + 2$ shuffles (see Table 1).

In particular, for a standard 9×9 Sudoku puzzle, our protocol (with the second method of implementation) uses 108 cards and can be performed using two standard decks of playing cards, regardless of whether the two decks are the same or different types (see Table 2).

Theoretically, this work is an important step in card-based cryptography as it is the first ZKP protocol for any logic puzzle that can be performed using a deck of all different cards, an open problem posed in [11].

Table 1. The number of required cards and shuffles for an $n \times n$ Sudoku

Protocol	Standard Deck?	#Cards	#Shuffles
Sasaki et al. [24]	No	$n^2 + n$	$5n$
Ours (Sect. 5.1)	Yes	$n^2 + n\sqrt{n} + n + \sqrt{n}$	$4n\sqrt{n}$
Ours (Sect. 5.2)	Yes	$n^2 + 2n + 3\sqrt{n}$	$2n^2(\sqrt{n} - 1)$ for even n $2n^2(\sqrt{n} - 1) + 2$ for odd $n > 9$

Table 2. The number of required cards and shuffles for a 9×9 Sudoku

Protocol	Standard Deck?	#Cards	#Shuffles
Sasaki et al. [24]	No	90	45
Ours (Sect. 5.1)	Yes	120	108
Ours (Sect. 5.2)	Yes	108	322

4 Preliminaries

At first, we assume that all cards used in our protocols have different front sides and identical back sides (although we will later show that some pairs of cards can have identical front sides or different back sides, and our protocol still works correctly).

4.1 Marked Matrix

Suppose we have a $k \times \ell$ matrix of face-down cards (we call these cards *encoding cards*). Let Row i denote an i-th topmost row and let Column j denote a j-th leftmost column. To the left of Column 1, publicly place face-down cards $p_1, p_2, ..., p_k$ in this order from top to bottom; this new column is called Column 0. Analogously, above Row 1, publicly place face-down cards $q_1, q_2, ..., q_\ell$ in this order from left to right; this new row is called Row 0.

We call this structure a $k \times \ell$ *marked matrix* (see Fig. 3), and we call the cards in Row 0 and Column 0 *marking cards*.

4.2 Shuffle Operations

Suppose we have a $k \times \ell$ marked matrix. For a set $S \subseteq \{1, 2, ..., k\}$, an operation `row_shuf`$(S)$ rearranges the rows in S (including marking cards in Column 0) by a uniformly random permutation. For example, `row_shuf`$(\{1, 3, 4\})$ rearranges Row 1, Row 3, and Row 4 of the matrix by a uniformly random permutation. This can be performed in real world by putting all cards in each row in S into an envelope and scrambling all envelopes together.

Analogously, for a set $S \subseteq \{1, 2, ..., \ell\}$, an operation `col_shuf`$(S)$ rearranges the columns in S (including marking cards in Row 0) by a uniformly random permutation.

Fig. 3. An example of a 4×5 marked matrix

4.3   Rearrangement Protocol

After applying some shuffle operations to a marked matrix, a *rearrangement protocol* reverts the matrix back to its original state. Slightly different variants of this protocol with the same idea has been used in previous work [2,7,8,21,22,24].

Suppose we have a $k \times \ell$ marked matrix M with marking cards $p_1, p_2, ..., p_k$ in Column 0 and $q_1, q_2, ..., q_\ell$ in Row 0. We perform the following steps.

1. Apply `row_shuffle`$(\{1, 2, ..., k\})$ and `col_shuffle`$(\{1, 2, ..., \ell\})$ to M.
2. Turn over all marking cards in Column 0 and Row 0. Rearrange the rows of M such that the marking cards in Column 0 are $p_1, p_2, ..., p_k$ in this order from top to bottom. Rearrange the columns of M such that the marking cards in Row 0 are $q_1, q_2, ..., q_\ell$ in this order from left to right.

4.4   Standard Deck Chosen Cut Protocol

Given a $k \times \ell$ marked matrix M, this protocol allows the prover P to choose a card located at Row i and Column j of M he/she wants without revealing i or j. It was modified from an original chosen cut protocol of Koch and Walzer [10] (which uses identical copies of ♣ and ♡) so that it can be performed using a standard deck. P performs the following steps.

1. Secretly stack a face-down card x_1 on a card located at Row i and Column j.
2. On each of the remaining $k\ell - 1$ cards in the matrix, secretly stack each of face-down cards $x_2, x_3, ..., x_{k\ell}$ in a uniformly random order. The cards $x_1, x_2, ..., x_{k\ell}$ are called *helper cards*.
3. Apply `row_shuffle`$(\{1, 2, ..., k\})$ and `col_shuffle`$(\{1, 2, ..., \ell\})$ to M.
4. Turn over all helper cards. Locate the position of x_1. The encoding card in that stack is the one originally located at Row i and Column j as desired.
5. Remove all helper cards. Apply the rearrangement protocol to revert M to its original state.

This protocol will be implicitly used in our main protocol, with Step 3 being replaced by equivalent operations.

5   Main Protocol

For simplicity, we will show a protocol for a standard 9×9 Sudoku puzzle. Our protocol can be straightforwardly generalized to an $n \times n$ puzzle.

We use the following cards in our protocol.

– encoding cards $a_j, b_j, c_j, d_j, e_j, f_j, g_j, h_j, i_j$ $(j = 1, 2, ..., 9)$
– marking cards p_j $(j = 1, 2, 3)$ and q_j $(j = 1, 2, ..., 9)$
– helper cards x_j, y_j, z_j $(j = 1, 2, ..., 9)$

Suppose the grid is divided into blocks $A, B, ..., I$ (see Fig. 4). We use a card a_j $(j = 1, 2, ..., 9)$ to encode a number j in Block A. Analogously, we use cards $b_j, c_j, ..., i_j$ $(j = 1, 2, ..., 9)$ to encode numbers j in blocks $B, C, ..., I$, respectively.
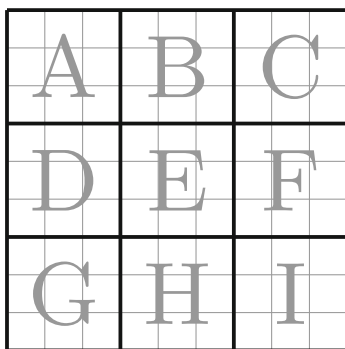

Fig. 4. Blocks A, B, C, D, E, F, G, H, and I in the grid

On each cell already having a number, P publicly places a face-down corresponding card (e.g. places a card b_3 on a cell with a number 3 in Block B). On each empty cell, P secretly places a face-down corresponding card according to his/her solution.

Apply the uniqueness verification protocol in Sect. 2.1 to verify that Block A consists of cards $a_1, a_2, ..., a_9$ in some order. Do the same for Blocks $B, C, ..., I$. Now V is convinced that every number from 1 to 9 appears exactly once in each block.

Next, we will show two methods to verify that every number from 1 to 9 appears exactly once in each row and column.

5.1   Method A

First, P performs the following steps to verify that a number 1 appears exactly once in each of the three topmost rows.

1. Take the cards from the three topmost rows to form a 3×9 matrix and publicly place marking cards p_1, p_2, p_3 in Column 0 and $q_1, q_2, ..., q_9$ in Row 0 to create a marked matrix M.

2. Secretly stack face-down cards x_1, y_1, and z_1 on a_1, b_1, and c_1, respectively.
3. On each of the remaining 8 cards in Block A, secretly stack each of face-down cards $x_2, x_3, ..., x_9$ in a uniformly random order. Do the same for cards $y_2, y_3, ..., y_9$ in Block B and $z_2, z_3, ..., z_9$ in Block C.
4. Apply `row_shuffle`($\{1,2,3\}$), `col_shuffle`($\{1,2,3\}$), `col_shuffle`($\{4,5,6\}$), and `col_shuffle`($\{7,8,9\}$) to M.
5. Turn over all helper cards. Locate the positions of x_1, y_1, and z_1. Turn over the encoding cards in these three stacks to show that they are a_1, b_1, and c_1, respectively, and that they are all located at different rows. Otherwise, V rejects.
6. Remove all helper cards and turn all encoding cards face-down. Apply the rearrangement protocol in Sect. 4.3 to revert M to its original state.

Note that Steps 2 to 6 are equivalent to applying the standard deck chosen cut protocol in Sect. 4.4 to Blocks A, B, and C, simultaneously. These steps ensure that the three 1s in Blocks A, B, and C are all located at different rows. Since it has already been shown that each block contains exactly one 1, this implies there is exactly one 1 in each of the three topmost rows.

P performs these steps analogously for numbers $2, 3, ..., 9$. Now V is convinced that every number appears exactly once in each of the three topmost rows.

P then does the same for Blocks D, E, and F and for Blocks G, H, and I to verify the rest of the rows. The verification for columns works analogously (P takes the cards from Blocks A, D, and G, from Blocks B, E, and H, and from Blocks C, F, and I, and just transposes the matrix).

This method uses 81 encoding cards, 12 marking cards, and 27 helper cards, resulting in the total of 120 cards, slightly more than the number of cards in two standard decks, and uses 342 shuffles.[1] We aim to further reduce the number of required cards as a trade-off between the numbers of cards and shuffles.

5.2   Method B

In Method A, we verify that the three 1s in Blocks A, B, and C are all located at different rows by verifying these three blocks at the same time, which requires a lot of marking and helper cards. Instead, we can first verify that the two 1s in Blocks A and B are located at different rows, then do the same for Blocks A and C, and for Blocks B and C. This leads to the same conclusion that the three 1s in Blocks A, B, and C are all located at different rows. The formal steps for verifying Blocks A and B are shown below.

1. Take the cards from blocks A and B to form a 3×6 matrix and publicly place marking cards p_1, p_2, p_3 in Column 0 and $q_1, q_2, ..., q_6$ in Row 0 to create a marked matrix M.
2. Secretly stack face-down cards x_1 and y_1 on a_1 and b_1, respectively.

---

[1] The number of shuffles can be reduced to 108 after optimization. See the full version.

3. On each of the remaining 8 cards in Block A, secretly stack each of face-down cards $x_2, x_3, ..., x_9$ in a uniformly random order. Do the same for cards $y_2, y_3, ..., y_9$ in Block B.
4. Apply `row_shuffle`($\{1, 2, 3\}$), `col_shuffle`($\{1, 2, 3\}$), and `col_shuffle` ($\{4, 5, 6\}$) to M.
5. Turn over all helper cards. Locate the positions of x_1 and y_1. Turn over the encoding cards in both stacks to show that they are a_1 and b_1, respectively, and that they are located at different rows. Otherwise, V rejects.
6. Remove all helper cards and turn all encoding cards face-down. Apply the rearrangement protocol in Sect. 4.3 to revert M to its original state.

We say that two cards are from the same set if they are denoted by the same letter with different indices (e.g. d_2 and d_5 are from the same set). Notice that in both methods, cards from different sets never get mixed together. Therefore, cards from different sets can have identical front sides or different back sides (or even different sizes) and our protocol still works correctly. The only requirement is that all cards from the same set must have different front sides and identical back sides.

This method uses 81 encoding cards, nine marking cards, and 18 helper cards, resulting in the total of 108 cards, which is exactly the number of cards from two standard decks (including jokers), and uses 828 shuffles.[2] We can, for example, use 54 cards from the first deck in the sets $a_j, b_j, ..., f_j$ and 54 cards from the second deck in the remaining sets. The protocol works correctly regardless of whether the two decks are identical or different, since it allows cards from different sets to have identical front sides (in case of identical decks) or different back sides or sizes (in case of different decks). Note that in some decks, the two jokers are identical; in that case, we just need to make sure that the two jokers are in different sets.

5.3 Generalization

Our protocol can be straightforwardly generalized to an $n \times n$ puzzle.

Method A uses n^2 encoding cards, $n + \sqrt{n}$ marking cards, and $n\sqrt{n}$ helper cards, resulting in the total of $n^2 + n\sqrt{n} + n + \sqrt{n}$ cards. It uses $4n\sqrt{n}$ shuffles (after the optimization).

Method B uses n^2 encoding cards, $3\sqrt{n}$ marking cards, and $2n$ helper cards, resulting in the total of $n^2 + 2n + 3\sqrt{n}$ cards. It uses at most $2n^2(\sqrt{n} - 1) + 2$ shuffles (after the optimization).

6 Proof of Correctness and Security

We will prove the perfect completeness, perfect soundness, and zero-knowledge properties of our protocol.

---

[2] The number of shuffles can be reduced to 322 after optimization. See the full version.

Lemma 1 (Perfect Completeness). *If P knows a solution of the Sudoku puzzle, then V always accepts.*

Proof. Suppose P knows a solution and places cards on the grid accordingly. Every number from 1 to 9 will appear exactly once in each row, each column, and each block. Hence, the uniqueness verification protocol will pass for every block. Also, the same numbers from different blocks are always located at different rows and columns, so both Methods A and B will pass. Therefore, V always accepts. □

Lemma 2 (Perfect Soundness). *If P does not know a solution of the Sudoku puzzle, then V always rejects.*

Proof. Suppose P does not know a solution. There will be a number that appears at least twice in the same row, column, or block. If it appears twice in a block, the uniqueness verification protocol for that block will fail. If it appears twice in different blocks in the same row (resp. column), Method A will fail when verifying the three blocks containing that row (resp. column); also, method B will fail when verifying the two blocks where these two numbers appear. Therefore, V always rejects. □

Lemma 3 (Zero-Knowledge). *During the verification, V learns nothing about P's solution.*

Proof. It is sufficient to show that all distributions of cards that are turned face-up can be simulated by a simulator S that does not know P's solution.

– In Steps 3 and 5 of the uniqueness verification protocol in Sect. 2.1, the orders of the n cards are uniformly distributed among all $n!$ permutations. Hence, it can be simulated by S.
– In Step 2 of the rearrangement protocol in Sect. 4.3, the orders of $p_1, p_2, ..., p_k$ and $q_1, q_2, ..., q_\ell$ are uniformly distributed among all $k!$ permutations and $\ell!$ permutations, respectively. Hence, it can be simulated by S.
– In Step 5 of Method A in Sect. 5.1, the rows where x_1, y_1, and z_1 are located are uniformly distributed among all $3! = 6$ permutations of the first three rows; the columns where they are located are uniformly distributed among all $3^3 = 27$ combinations of three columns from Blocks A, B, and C. Also, the orders of $x_2, x_3, ..., x_9$ are uniformly distributed among all $8!$ permutations of the remaining cards in Block A; the same goes for $y_2, y_3, ..., y_9$ in Block B and $z_2, z_3, ..., z_9$ in Block C. Hence, it can be simulated by S.
– In Step 5 of Method B in Sect. 5.2, the rows where x_1 and y_1 are located are uniformly distributed among all $\frac{3!}{1!} = 6$ 2-permutations of the first three rows; the columns where they are located are uniformly distributed among all $3^2 = 9$ combinations of two columns from Blocks A and B. Also, the orders of $x_2, x_3, ..., x_9$ are uniformly distributed among all $8!$ permutations of the remaining cards in Block A; the same goes for $y_2, y_3, ..., y_9$ in Block B. Hence, it can be simulated by S. □

7   Future Work

We developed the first ZKP protocol for Sudoku, and also the first one for any logic puzzle, that uses a deck of all different cards. Our protocol for a standard 9×9 Sudoku can be performed using two standard decks of playing cards. However, the drawback of our protocol is that it uses a large number of shuffles, which makes it impractical. A possible future work is to develop an equivalent protocol for Sudoku that uses asymptotically less number of shuffles. Other challenging future work includes developing ZKP protocols for other logic puzzles (e.g. Kakuro, Numberlink) that uses a deck of all different cards.

References

1. Bultel, X., Dreier, J., Dumas, J.G., Lafourcade, P.: Physical zero-knowledge proofs for akari, takuzu, kakuro and kenken. In: Proceedings of the 8th International Conference on Fun with Algorithms (FUN), pp. 8:1–8:20 (2016)
2. Bultel, X., et al.: Physical zero-knowledge proof for makaro. In: Izumi, T., Kuznetsov, P. (eds.) SSS 2018. LNCS, vol. 11201, pp. 111–125. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03232-6_8
3. Chien, Y.-F., Hon, W.-K.: Cryptographic and physical zero-knowledge proof: from sudoku to nonogram. In: Boldi, P., Gargano, L. (eds.) FUN 2010. LNCS, vol. 6099, pp. 102–112. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13122-6_12
4. Dumas, J.-G., Lafourcade, P., Miyahara, D., Mizuki, T., Sasaki, T., Sone, H.: Interactive physical zero-knowledge proof for norinori. In: Du, D.-Z., Duan, Z., Tian, C. (eds.) COCOON 2019. LNCS, vol. 11653, pp. 166–177. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26176-4_14
5. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM J. Comput. **18**(1), 186–208 (1989)
6. Gradwohl, R., Naor, M., Pinkas, B., Rothblum, G.N.: Cryptographic and physical zero-knowledge proof systems for solutions of sudoku puzzles. Theory Comput. Syst. **44**(2), 245–268 (2009)
7. Hashimoto, Y., Shinagawa, K., Nuida, K., Inamura, M., Hanaoka, G.: Secure grouping protocol using a deck of cards. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **101.A**(9), 1512–1524 (2018)
8. Ibaraki, T., Manabe, Y.: A more efficient card-based protocol for generating a random permutation without fixed points. In: Proceedings of the 3rd International Conference on Mathematics and Computers in Sciences and Industry (MCSI), pp. 252–257 (2016)
9. Koch, A., Schrempp, M., Kirsten, M.: Card-based cryptography meets formal verification. New Gener. Comput. **39**(1), 115–158 (2021)
10. Koch, A., Walzer, S.: Foundations for actively secure card-based cryptography. In: Proceedings of the 10th International Conference on Fun with Algorithms (FUN), pp. 17:1–17:23 (2020)
11. Koyama, H., Miyahara, D., Mizuki, T., Sone, H.: A secure three-input and protocol with a standard deck of minimal cards. In: Proceedings of the 16th International Computer Science Symposium in Russia (CSR), pp. 242–256 (2021)

12. Lafourcade, P., Miyahara, D., Mizuki, T., Sasaki, T., Sone, H.: A physical ZKP for slitherlink: how to perform physical topology-preserving computation. In: Heng, S.-H., Lopez, J. (eds.) ISPEC 2019. LNCS, vol. 11879, pp. 135–151. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34339-2_8

13. Miyahara, D., Hayashi, Y., Mizuki, T., Sone, H.: Practical card-based implementations of Yao's millionaire protocol. Theor. Comput. Sci. **803**, 207–221 (2020)

14. Miyahara, D., et al.: Card-based ZKP protocols for takuzu and juosan. In: Proceedings of the 10th International Conference on Fun with Algorithms (FUN), pp. 20:1–20:21 (2020)

15. Miyahara, D., Sasaki, T., Mizuki, T., Sone, H.: Card-based physical zero-knowledge proof for kakuro. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E102.A**(9): 1072–1078 (2019)

16. Mizuki, T.: Efficient and secure multiparty computations using a standard deck of playing cards. In: Proceedings of the 15th International Conference on Cryptology and Network Security (CANS), pp. 484–499 (2016)

17. Niemi, V., Renvall, A.: Solitaire zero-knowledge. Fundam. Inform. **38**(1,2), 181–188 (1999)

18. Nikoli: Sudoku. https://www.nikoli.co.jp/en/puzzles/sudoku.html

19. Robert, L., Miyahara, D., Lafourcade, P., Mizuki, T.: Interactive physical ZKP for connectivity: applications to nurikabe and hitori. In: Proceedings of the 17th Conference on Computability in Europe (CiE), pp. 373–384 (2021)

20. Robert, L., Miyahara, D., Lafourcade, P., Mizuki, T.: Physical zero-knowledge proof for suguru puzzle. In: Devismes, S., Mittal, N. (eds.) SSS 2020. LNCS, vol. 12514, pp. 235–247. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64348-5_19

21. Ruangwises, S., Itoh, T.: Physical zero-knowledge proof for numberlink puzzle and k vertex-disjoint paths problem. New Gener. Comput. **39**(1), 3–17 (2021)

22. Ruangwises, S., Itoh, T.: Physical zero-knowledge proof for ripple effect. In: Proceedings of the 15th International Conference and Workshops on Algorithms and Computation (WALCOM), pp. 296–307 (2021)

23. Ruangwises, S., Itoh, T.: Physical ZKP for connected spanning subgraph: applications to bridges puzzle and other problems. In: Proceedings of the 19th International Conference on Unconventional Computation and Natural Computation (UCNC) (2021, in press)

24. Sasaki, T., Miyahara, D., Mizuki, T., Sone, H.: Efficient card-based zero-knowledge proof for Sudoku. Theor. Comput. Sci. **839**, 135–142 (2020)

25. Yato, T., Seta, T.: Complexity and completeness of finding another solution and its application to puzzles. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **86.A**(5), 1052–1060 (2003)