Interactive Physical Zero-Knowledge Proof for Norinori

Jean-Guillaume Dumas[1][0000−0002−2591−172X], Pascal Lafourcade[2][0000−0002−4459−511X], Daiki Miyahara[3,5], Takaaki Mizuki[4], Tatsuya Sasaki[3], and Hideaki Sone[4]

[1] Univ. Grenoble Alpes, CNRS, Grenoble INP[⋆⋆], LJK, 38000 Grenoble, France
[2] LIMOS, University Clermont Auvergne, CNRS UMR 6158, France
[3] Graduate School of Information Sciences, Tohoku University, Japan
[4] Cyberscience Center, Tohoku University, Japan
[5] National Institute of Advanced Industrial Science and Technology, Japan

Abstract. Norinori is a logic game similar to Sudoku. In Norinori, a grid of cells has to be filled with either black or white cells so that the given areas contain exactly two black cells, and every black cell shares an edge with exactly one other black cell. We propose a secure interactive physical algorithm, relying only on cards, to realize a zero-knowledge proof of knowledge for Norinori. It allows a player to show that he or she knows a solution without revealing it. For this, we show in particular that it is possible to physically prove that a particular element is present in a list, without revealing any other value in the list, and without revealing the actual position of that element in the list.

Keywords: Zero-knowledge proofs · Card-based secure two-party protocols · Puzzle · Norinori

1 Introduction

Sudoku, introduced under this name in 1986 by the Japanese puzzle company Nikoli, and similar games such as Akari, Takuzu, Makaro, and Norinori have gained immense popularity in recent years. Many of them have been proved to be NP-complete [1, 5, 10, 11]. In 2007, Gradwohl, Naor, Pinkas, and Rothblum proposed the first physical zero-knowledge proof protocols for Sudoku [7]. Their protocol only use several cards and allow a prover to prove to a verifier that he knows a solutions of a Sudoku grid. More precisely, a zero-knowledge proof of knowledge is a protocol that allows a prover P to convince a verifier V that she knows the solution w of an instance of a computational problem. Such a protocol has the following properties:

Correctness. If P knows w, then P can convince V.

[⋆⋆] Institute of Engineering Univ. Grenoble Alpes

Extractability. If P does not know w, then P cannot convince V, except with some *small* probability (here will have *perfect* Extractability, that is the probability that V does not detect a wrong grid is zero). This implies the standard *soundness* property, which ensures that if there exists no solution of the puzzle, then the prover is not able to convince the verifier regardless of the prover's behavior.

Zero-Knowledge. V cannot obtain any information about w. To prove that a protocol satisfies the zero-knowledge property, it is sufficient to exhibit a probabilistic polynomial time algorithm $M(\mathcal{I})$, not knowing w, such that the outputs of the protocol and the outputs of $M(\mathcal{I})$ follow the same probability distribution.

Recently, a novel protocol for Sudoku using fewer cards and with no soundness error was then proposed [15]. Physical protocols for other games, such as Hanjie, Akari, Kakuro, KenKen, Takuzu, and Makaro, have been designed [2–4]. In this article, we propose the first interactive physical zero-knowledge proof protocol for *Norinori*.

Norinori: It is a pencil puzzle published in the famous puzzle magazine *Nikoli*. The puzzle instance is a rectangular grid of cells. The grid is partitioned into *rooms*, that is, areas surrounded by bold lines. The goal of the puzzle is to shade certain cells so that they become black, according to the following rules [13]:

1. *Room condition*: Each room must contain exactly two black cells.
2. *Pair condition*: The black cells come in pairs: each black cell must be adjacent to exactly one, and only one, other black cell.

In Figure 1, we give a simple example of a Norinori game. It is easy to verify that both constraints are satisfied in the solution on the right part of the figure. We note that in a solution the number of black cells is exactly twice the number of rooms.

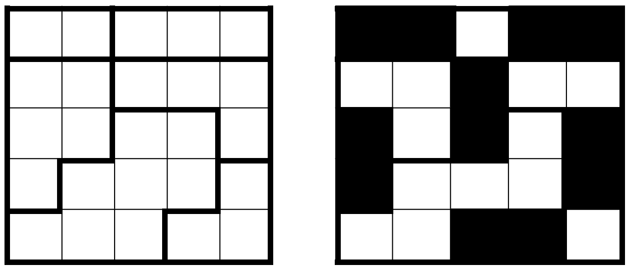


Fig. 1. Example of a Norinori grid and its solution.

Solving Norinori was shown to be NP-complete via a reduction from PLANAR 1-IN-3-SAT in [1]. Hence, it is possible to construct a cryptographic zero-knowledge proof of this game by using the generic cryptographic zero-knowledge proofs for all problems in NP given in [6]. However, this construction requires cryptographic primitives and is not very efficient.

Contributions: Our aim in this paper is to design an interactive physical protocol that uses only cards and envelopes for Nornori. This paper is not just another paper that proposed a physical zero-knowledge protocol for a Nikoli's game but we want to extend the physically verifiable set of functions that we are able to solve using only physical material. For instance, we know how to guarantee the presence of all numbers in some set without revealing their order [7], how to guarantee that two numbers are distinct without revealing their respective values [2], or how to prove that a number is the largest in a list, without revealing any value in the list [3]. In this paper, by providing a physical zero-knowledge proof for the Nikoli puzzle Norinori, we show in particular that it is possible to physically prove that a particular element is present in a list, without revealing any other value in the list, and without revealing the actual position of that element in the list.

Outline: In the next section, we introduce some notations and explain simple physical subprotocols that we use in our protocol for Norinori. In Section 3, we construct our zero-knowledge protocol for Norinori, before giving a security analysis in Section 4 and concluding in Section 5.

2 Preliminaries

In this section, we present some notations and introduce shuffling operations and a subprotocol that will be used later.

2.1 Physical Objects

The physical cards used in this paper are given in Table 1. We assume that the back sides of all cards are identical[6], and the face sides of all the cards of each type (such as $\boxed{}, \boxed{\clubsuit}, \boxed{\checkmark}, \boxed{\mathbf{S}}, \boxed{1}, \ldots$) are also identical. We use the notation $(c_1, c_2, \ldots, c_k)$ to represent a sequence of k face-down cards $\boxed{?}\,\boxed{?}\,\ldots\,\boxed{?}$. We also consider a pile of cards $\boxed{?}\|$ consisting of ℓ face-down cards and express it as a vector $\boldsymbol{p}$. Moreover, a sequence of k piles $(\boxed{?}\|, \boxed{?}\|, \ldots, \boxed{?}\|)$ is expressed as a k-tuple of vectors $(\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_k)$.

2.2 Pile-Scramble Shuffle

Pile-Scramble Shuffle [8] is a shuffle operation for piles: for a sequence of k piles $(\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_k)$, applying a Pile-Scramble Shuffle results in $(\boldsymbol{p}_{\pi^{-1}(1)}, \boldsymbol{p}_{\pi^{-1}(2)}, \ldots, \boldsymbol{p}_{\pi^{-1}(k)})$, where $\pi \in S_k$ is a uniformly distributed random permutation, and S_k is the symmetric group of degree k. A Pile-Scramble Shuffle can be implemented by physical cases, for instance by using a big box where we place all the cases that contain the pile of cards and we blindly shuffle them.

[6] It means that the cards face down are indistinguishable from each other.

Table 1. Names of cards.

Face side	Back side	Name
$\boxed{}$	$\boxed{?}$	White card
$\boxed{\clubsuit}$	$\boxed{?}$	Black card
$\boxed{\checkmark}$	$\boxed{?}$	Maker card
$\boxed{\mathbf{S}}$	$\boxed{?}$	Starting card
$\boxed{1}\,\boxed{2}\,\boxed{3}\,\boxed{4}$	$\boxed{?}$	Number card

2.3 Pile-Shifting Shuffle

Pile-Shifting Shuffle (which is also called Pile-Shifting Scramble) [14] cyclically shuffles piles of cards. That is, given a sequence of k piles, each of which consists of the same number of face-down cards, denoted by $(\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_k)$, applying a Pile-Shifting Shuffle results in $(\boldsymbol{p}_{r+1}, \boldsymbol{p}_{r+2}, \ldots, \boldsymbol{p}_{r+k})$, where r is uniformly randomly generated from $\mathbb{Z}/k\mathbb{Z}$ and is unknown to everyone. To implement a Pile-Shifting Scramble, we use physical cases that can store a pile of cards, such as boxes or envelopes. One possible implementation is to place the different pile of cards in cases that are linked together in a circle in order to form a cycle. Then we just have to physically shuffle the cases, for instance by turning the circle.

2.4 Card Choosing Protocol

In this subsection, we describe the Card Choosing Protocol, which is immediately obtained by borrowing the idea behind the Chosen Cut [12]. This protocol is used as a subprotocol in our construction in Section 3.

Given a sequence of k face-down cards $(c_1, c_2, \ldots, c_k)$, the Card Choosing Protocol enables the prover P to choose a designated card secretly. More precisely, for some i such that $1 \leq i \leq k$, P can choose the i-th card without revealing any information about i to the verifier V. The protocol proceeds as follows:

1. P holds $k-1$ white cards and one black card. Then, P puts them with their faces down below the cards $(c_1, c_2, \ldots, c_k)$, such that only the i-th card is the black card:

$$\begin{array}{cccccccc} \boxed{?} & \boxed{?} & \cdots & \boxed{?} & \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ c_1 & c_2 & & c_{i-1} & c_i & c_{i+1} & & c_k \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} & \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{} & \boxed{} & & \boxed{} & \boxed{\clubsuit} & \boxed{} & & \boxed{} \end{array}$$

2. Regarding cards in the same column as a pile, apply a Pile-Shifting Shuffle to the sequence of piles (which is denoted by $\langle \cdot | \ldots | \cdot \rangle$):

$$\left\langle \begin{array}{c} \boxed{?} \\ c_1 \\ \boxed{?} \end{array} \middle| \begin{array}{c}\boxed{?}\\ c_2\\ \boxed{?}\end{array} \middle| \cdots \middle| \begin{array}{c}\boxed{?}\\ c_k \\ \boxed{?}\end{array} \right\rangle \rightarrow \begin{array}{cccc} \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ c_{r+1} & c_{r+2} & & c_{r+k} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{array},$$

where $r \in \mathbb{Z}/k\mathbb{Z}$ is a uniformly distributed random value.
3. Reveal all the cards in the second row. Then, one black card appears, and the card above the revealed black card is the i-th card:

$$\begin{array}{cccccccc} \boxed{?}&\boxed{?}&\cdots&\boxed{?}&\boxed{?}&\boxed{?}&\cdots&\boxed{?}\\ &&&&c_i&&&\\ \boxed{}&\boxed{}&\cdots&\boxed{}&\boxed{\clubsuit}&\boxed{}&\cdots&\boxed{}\end{array}$$

Thus, P can show the designated card to V.

Because all the opened cards are shuffled in Step 2, V does not learn any information about the index i of the chosen card and is sure that only one card was designated.

3 Zero-Knowledge Proof for Norinori

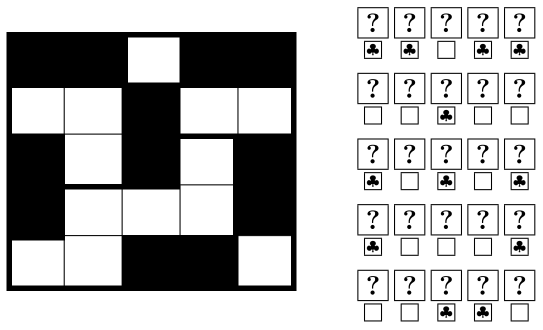
We are now ready to describe our construction of a zero-knowledge proof for Norinori. For a puzzle instance of board size $m \times n$ including exactly t rooms, assume that the prover P knows the solution. Our protocol consists of three phases, namely:

1. Setup phase,
2. Pair Verification phase,
3. Room Verification phase.

It is important to perform the phases in this order. The Room Verification phase has to be the last one and the Setup phase the first phase.

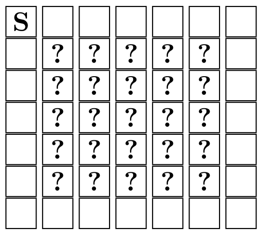
Setup phase: This phase has two steps:

1. P puts one face-down card on each cell according to the solution. That is, a black card is placed on every cell where a black square exists in the solution, and white cards are placed on all the other cells. For example, for the puzzle instance in Fig. 1, the cards are placed as follows:



These cards are P's input, and we assume that P knows his or her input. This implies that if the cards conform to the solution, P knows the solution. Note that $2 \times t$ black cards and $m \times n - (2 \times t)$ white cards on a board of dimension $m \times n$ including exactly t rooms.

2. Next, remembering that m is the number of rows and n is the number of columns, V takes $2 \times m + 2 \times n + 3$ white cards and one starting card. Then, V puts these cards around the $m \times n$ "matrix" above to expand it to an $(m+1) \times (n+1)$ "matrix" as follows, where the starting card is placed at the top-left corner:

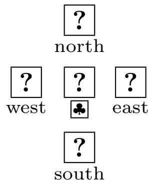


(1)

Pick each row from top to bottom to make a sequence of cards:

$$\underset{\mathbf{s}}{\boxed{?}}\ \overbrace{\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\cdots\boxed{?}\boxed{?}}^{m \times n + 2 \times m + 2 \times n + 3 \text{ cards}}$$

Notation: For any black card (whose location is known only to P) in the matrix, we define its four adjacent cards called *north*, *east*, *west*, and *south* cards:



Pair Verification Phase: The verifier V verifies that P's input satisfies the Pair condition. If P has placed the cards correctly, every black card will have exactly one black card among its adjacent cards. The verification of the Pair condition is to guarantee this, and it proceeds as follows:

1. P selects a black card mentally from the card sequence, and performs the Card Choosing Protocol to choose that card:

$$\begin{array}{l} \boxed{?}\cdots\boxed{?}\boxed{?}\boxed{?}\cdots\boxed{?} \\ \boxed{}\cdots\boxed{}\boxed{\clubsuit}\boxed{}\cdots\boxed{} \end{array}$$

Then, V opens the card chosen by P to make sure that the chosen card is black:

$$\begin{array}{l} \qquad\quad\ \ \downarrow \\ \boxed{?}\cdots\boxed{?}\boxed{\clubsuit}\boxed{?}\cdots\boxed{?} \\ \boxed{}\cdots\boxed{}\boxed{\clubsuit}\boxed{}\cdots\boxed{} \end{array}$$

After confirming that the card is black, V replaces it by a marker card:

$$\begin{array}{c} \boxed{?}\cdots\boxed{?}\overset{\downarrow}{\boxed{\checkmark}}\boxed{?}\cdots\boxed{?} \\ \boxed{}\cdots\boxed{}\boxed{\clubsuit}\boxed{}\cdots\boxed{} \end{array}$$

This operation prevents P from choosing the same black card again at later verifications.

2. V picks the four adjacent cards of the chosen black card. Because the sequence of cards was shifted cyclically, V can find such four cards, namely, the north, east, west, and south cards, by counting the distances[7]:

$$\underset{\text{west}}{\boxed{?}}\ \underset{\text{north}}{\boxed{?}}\ \underset{\text{east}}{\boxed{?}}\ \underset{\text{south}}{\boxed{?}}$$

3. V puts number cards $\boxed{1}\,\boxed{2}\,\boxed{3}\,\boxed{4}$ in this order below these cards, and turns them over:

$$\begin{array}{cccc} \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ \boxed{1} & \boxed{2} & \boxed{3} & \boxed{4} \end{array} \rightarrow \begin{array}{cccc} \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array}$$

4. P regards cards in the same column as a pile and applies a Pile-Scramble Shuffle to the sequence of piles:

$$\left\langle \begin{array}{c|c|c|c} \boxed{?} & \boxed{?} & & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{array} \right\rangle \rightarrow \begin{array}{cccc} \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array}$$

5. V reveals the top row and checks that there is exactly one black or one marker card:

$$\begin{array}{cccc} \boxed{} & \boxed{\clubsuit} & \boxed{} & \boxed{} \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array} \quad \text{or} \quad \begin{array}{cccc} \boxed{} & \boxed{\checkmark} & \boxed{} & \boxed{} \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array}$$

After checking it, the cards in the top row are turned over.

6. P applies a Pile-Scramble Shuffle to the piles again, and reveals the bottom row:

$$\left\langle \begin{array}{c|c|c|c} \boxed{?} & \boxed{?} & & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{array} \right\rangle \rightarrow \begin{array}{cccc} \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \end{array} \rightarrow \begin{array}{cccc} \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ \boxed{2} & \boxed{3} & \boxed{4} & \boxed{1} \end{array}$$

Because the number cards indicate the original order, V can place each top card back in the original position in the card sequence:

$$\boxed{?}\boxed{?}\cdots\underset{\text{north}}{\boxed{?}}\cdots\underset{\text{west}}{\boxed{?}}\boxed{\checkmark}\underset{\text{east}}{\boxed{?}}\cdots\underset{\text{south}}{\boxed{?}}\cdots\boxed{?}$$

Turn over the face-up marker card. Note that P knows the locations of the other black cards and the starting card.

[7] For example, if the size of the puzzle board is 3×4, the north card is the fifth card away from the chosen card to the left, and the south card is the sixth card away from it to the right.

P and V repeat Steps 1 to 6 above $2 \times t - 1$ times more; recall that $2 \times t$ is the number of black cards. Furthermore, as a targeted black card is replaced by a marker card, V is convinced that $2t$ black cards are verified.

Next, P and V place the card sequence back on the puzzle board, as follows:

1. P chooses the starting card in the card sequence by performing a Card Choosing Protocol:

$$\begin{array}{c} \boxed{?}\cdots\boxed{?}\overset{\downarrow}{\boxed{\mathbf{S}}}\boxed{?}\cdots\boxed{?} \\ \boxed{}\cdots\boxed{}\boxed{?}\boxed{}\cdots\boxed{} \end{array}$$

2. Shift the cards so that the starting card is leftmost:

$$\boxed{\mathbf{S}}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\cdots\boxed{?}$$

   From the Card Choosing Protocols, the order of the cards is the same as the order of the card sequence generated in the Setup phase (although $2t$ black cards have been replaced by marker cards), and hence P and V can reconstruct the puzzle-board placement (1) by reversing the operations of the Setup phase.

Room Verification Phase: In this phase, V verifies the Room condition. To this end, the following is performed for each of the t rooms:

1. V picks all the cards from the room. Note that, regardless of the size of the targeted room, exactly two of the cards should be marker cards.
2. P shuffles the cards and reveals them.
3. V checks that exactly two marker cards appear.

If all phases have been passed, then the verifier accepts the proof by outputting 1.

Performance Analysis Let us mention the performance of our protocol in terms of the numbers of shuffles and cards. The total number of shuffles is $7 \times t + 1$ (where t is the number of rooms), and the total number of required cards is $2 \times m \times n + 4 \times m + 4 \times n + 2 \times t + 12$, whose distribution is shown in Table 2 (where we have an $m \times n$ board).

4 Security Analysis

We can easily see that our protocol satisfies the three properties of a zero-knowledge proof, as follows:

Correctness: If the prover P places the cards according to the solution in the Setup phase, P's input passes all verifications. Therefore, P who knows the solution, can always convince the verifier V.

Table 2. Numbers of cards required to execute our protocol.

Type of card	# of cards
White card ($\boxed{}$)	$2 \times m \times n + 4 \times m + 4 \times n - 2 \times t + 6$
Black card ($\boxed{\clubsuit}$)	$2 \times t + 1$
Marker card ($\boxed{\checkmark}$)	$2 \times t$
Starting card ($\boxed{\mathbf{S}}$)	1
Number card ($\boxed{1}\boxed{2}\boxed{3}\boxed{4}$)	4

Extractability: If P's input is invalid, V can detect it in the Verification phases. Therefore, if P does not know the solution, P cannot convince V.

Zero-knowledgeness: Since all the cards have been shuffled before they are opened, V learns nothing about the solution.

More precisely, we prove the following lemmas.

Lemma 1 (Correctness). *If the prover P has a solution for the Norinori puzzle, then P can always convince the verifier V (i.e., V outputs 1).*

Proof. We show that for a prover P with a solution, the verifier V never outputs 0. We look at the three phases:

Setup: In this phase the verifier just needs to check that the cards given by P correspond to size of the board. After that, V needs to place some extra cards and place all the cards in order to form a sequence. If P does not give the right number of cards, it is clear that he does not know the solution since he does not even know the puzzle itself.

Pair Verification: In this phase the goal of the verifier is to be convinced that all black cells come in pairs. If P knows a solution then he can place correctly the black cells. Hence the verifier never founds more than one marked or black card when he is opening the four adjacent cards of a black card. The only remaining uncertainty is whether P used exactly $2t$ black cards or more.

Room Verification: In this last phase, the verifier discovers exactly two black cards by room only if the prover knows a solution. As there are t rooms, this also proves that there were exactly $2t$ black cards to begin with.

Lemma 2 (Perfect Extractability). *If the prover does not know a solution for the Norinori puzzle, then the verifier V always rejects (i.e., V outputs 0) regardless of the prover P's behavior.*

Proof. It is important to notice that the order of our 3 phases is crucial and the fact that we can use the same cards for all the steps to ensure that we have no soundness error. In our proof, we consider two cases: the Pair condition is violated or the Room condition is violated.

Pair Condition: If the solution given by P does not satisfy the Pair condition, it means that some black cards do not come in pairs: some black cell are adjacent to more than one black card, or some black cells are isolated. We have three possible cases (modulo rotations and symmetries) that can occur:

1. There are 3 aligned black cards as follows: $\boxed{\clubsuit}\boxed{\clubsuit}\boxed{\clubsuit}$
2. There are 3 black cards that form an "L" as follows: $\begin{array}{ll}\boxed{\clubsuit} & \\ \boxed{\clubsuit} & \boxed{\clubsuit}\end{array}$
3. A single black card is surrounded by white cards.

In the first two cases, when V opens the four adjacent cards of a black card, he detects that there is strictly more than one extra black card in the neighborhood. This shows that the prover does not have a solution that satisfies the Pair condition. Similarly, in case of an isolated black card, the neighborhood will show only white cards. Overall, the verifier is convinced after this phase if and only if all black cards come in pairs.

Room Condition: If the solution given by P does not satisfy the Room condition, it means that a room contains one or no black card, or that a room contains three or more black cards. Both situation are detected by the verifier that opens all the cards in all the rooms.

Lemma 3 (Zero-knowledge). *During an execution of our protocol, the verifier V learns nothing about P's Norinori solution.*

Proof. In order to prove this, we have to describe an efficient simulator that simulates any interaction between a cheating verifier and a real prover. The simulator does not have a correct solution, but it does have an advantage over the prover: when shuffling decks, it is allowed to swap the packets for different ones. We thus show how to construct a simulator for each challenge.

Setup: V learns only the size of the board when he receives P's inputs, which is a known data already publicly avaible in the puzzle grid.

Pair Condition: In this phase the verifier does not learn any information except the number of black cards, which is $2 \times t$. This is not a secret information since it is a known data already publicly available in the puzzle grid. It is sufficient to show that all distributions of opening values are simulated without knowing the prover's solution.

Now, from [8,12,14], respectively, we know that the subprotocols Pile-Scramble Shuffle, Pile-Shifting Shuffle and Card Choosing Protocol are zero-knowledge. Note also, that during this verification phase, there is no need for the simulator to respect the Room condition. Thus a possible simulator is as follows: the simulator places at least one black card randomly on the grid. Then, for each pair verification, the simulator designates a black card and, at the Pile-Scramble step, P replaces the four adjacent cards by a single black and three white cards, e.g., as follows:

$$\begin{array}{cccc}\boxed{\clubsuit} & \boxed{} & \boxed{} & \boxed{} \\ \boxed{1} & \boxed{2} & \boxed{3} & \boxed{4}\end{array}$$

For the first step, the Card Choosing Protocol ensures that V does not learn anything: indeed it reveals that there is at least one black card on the board, but nothing about its position. Hence this subprotocol does not leak any information and is indistinguishable from the simulator.

For Step 4, the fact that P uses the Pile-Scramble Shuffle to the sequence of piles implies that their order is uniformly distributed and then can be simulated without knowing the solution. Therefore the position of the adjacent black cell is not leaked either.

At Step 6, using the Pile-Scramble Shuffle, the simulator replaces the initial cards (not even switching the tested black card by a marked card, see in the next phase). Note that the fact that we can replace the verified cards in their initial position without leaking any information comes from the usage of Pile-Scramble Shuffle in Step 6 and the usage of Card Choosing Protocol in the next step. It allows us to use the same cards for the next verification[8].

Room Condition: This phase is similar to a room/row/column condition in Sudoku [7, Protocol 3]: since P shuffles all the cards within a room before revealing them, this is indistinguishable from a simulation putting randomly two marked cards among white ones. Thus V only learns that there were exactly two black cards in each rooms.

5 Conclusion

In this paper, we have designed an interactive zero-knowledge proof protocol for the famous puzzle, Norinori. Our protocol is quite simple and easy to implement by humans. By solving this Nikoli's puzzle, we also demonstrate that it is possible to physically prove that a particular element is present in a list, without revealing any other value in the list, and without relvealing the actual position of that element in the list. This functionality could be used to construct zero-knowledge proof protocols for other puzzles.

An interesting open problem is to design a more efficient protocol in terms of the numbers of cards and shuffles.

References

1. Biro, M., Schmidt, C.: Computational complexity and bounds for Norinori and LITS. In: Proceedings of the 33rd European Workshop on Computational Geometry (EuroCG 2017), Malmö, Sweden. pp. 29–32 (Apr 2017)
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.) 8th International Conference on Fun with Algorithms, FUN 2016, June 8-10, 2016, La Maddalena, Italy. LIPIcs, vol. 49, pp. 8:1–8:20 (2016)
3. Bultel, X., Dreier, J., Dumas, J., Lafourcade, P., Miyahara, D., Mizuki, T., Nagao, A., Sasaki, T., Shinagawa, K., Sone, H.: Physical zero-knowledge proof for Makaro. In: Stabilization, Safety, and Security of Distributed Systems - 20th International

[8] This trick is inspired from [15] and allows us to also have no soundness error.

Symposium, SSS 2018, Tokyo, Japan, November 4-7, 2018, Proceedings. pp. 111–125 (2018). https://doi.org/10.1007/978-3-030-03232-6_8

4. Chien, Y.F., Hon, W.K.: Cryptographic and physical zero-knowledge proof: From Sudoku to Nonogram. In: Boldi, P., Gargano, L. (eds.) Fun with Algorithms, 5th International Conference, FUN 2010, Ischia, Italy, June 2-4, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6099, pp. 102–112. Springer (2010)
5. Demaine, E.D.: Playing games with algorithms: Algorithmic combinatorial game theory. In: Sgall, J., Pultr, A., Kolman, P. (eds.) Mathematical Foundations of Computer Science 2001, MFCS 2001. Lecture Notes in Computer Science, vol. 2136, pp. 18–32. Springer (2001)
6. Goldreich, O., Micali, S., Wigderson, A.: How to prove all NP-Statements in zero-knowledge, and a methodology of cryptographic protocol design. In: Odlyzko, A.M. (ed.) Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA. Lecture Notes in Computer Science, vol. 263, pp. 171–185. Springer (1987). https://doi.org/10.1007/3-540-47721-7_11, https://doi.org/10.1007/3-540-47721-7
7. Gradwohl, R., Naor, M., Pinkas, B., Rothblum, G.N.: Cryptographic and physical zero-knowledge proof systems for solutions of Sudoku puzzles. In: Crescenzi, P., Prencipe, G., Pucci, G. (eds.) Fun with Algorithms, 4th International Conference, FUN 2007, Castiglioncello, Italy, June 3-5, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4475, pp. 166–182. Springer (2007). https://doi.org/10.1007/978-3-540-72914-3_16
8. 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 - 14th International Conference, UCNC 2015, Auckland, New Zealand, August 30 - September 3, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9252, pp. 215–226. Springer (2015). https://doi.org/10.1007/978-3-319-21819-9_16
9. Ito, H., Leonardi, S., Pagli, L., Prencipe, G. (eds.): 9th International Conference on Fun with Algorithms, FUN 2018, La Maddalena, Italy, LIPIcs, vol. 100. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (Jun 2018)
10. Iwamoto, C., Haruishi, M., Ibusuki, T.: Herugolf and Makaro are NP-complete. In: Ito et al. [9], pp. 24:1–24:11. https://doi.org/10.4230/LIPIcs.FUN.2018.24, https://doi.org/10.4230/LIPIcs.FUN.2018.24
11. Kendall, G., Parkes, A.J., Spoerer, K.: A survey of NP-complete puzzles. ICGA Journal **31**(1), 13–34 (2008)
12. Koch, A., Walzer, S.: Foundations for actively secure card-based cryptography. IACR Cryptology ePrint Archive **2017**, 423 (2017), http://eprint.iacr.org/2017/423
13. Nikoli: Norinori, http://www.nikoli.co.jp/en/puzzles/norinori.html
14. Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: Pile-shifting scramble for card-based protocols. IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences **101**(9), 1494–1502 (2018)
15. Sasaki, T., Mizuki, T., Sone, H.: Card-based zero-knowledge proof for Sudoku. In: Ito et al. [9], pp. 29:1–29:10. https://doi.org/10.4230/LIPIcs.FUN.2018.29, https://doi.org/10.4230/LIPIcs.FUN.2018.29