How to Physically Verify a Rectangle in a Grid: A Physical ZKP for Shikaku

Suthee Ruangwises ✉ ⓘ
Department of Mathematical and Computing Science, Tokyo Institute of Technology, Japan

Toshiya Itoh ✉ ⓘ
Department of Mathematical and Computing Science, Tokyo Institute of Technology, Japan

─── **Abstract** ───

Shikaku is a pencil puzzle consisting of a rectangular grid, with some cells containing a number. The player has to partition the grid into rectangles such that each rectangle contains exactly one number equal to the area of that rectangle. In this paper, we propose two physical zero-knowledge proof protocols for Shikaku using a deck of playing cards, which allow a prover to physically show that he/she knows a solution of the puzzle without revealing it. Most importantly, in our second protocol we develop a general technique to physically verify a rectangle-shaped area with a certain size in a rectangular grid, which can be used to verify other problems with similar constraints.

1 Introduction

Shikaku is a pencil puzzle introduced by Nikoli, a Japanese publisher that developed many popular pencil puzzles such as Sudoku, Kakuro, and Slitherlink. The puzzle has become popular and many Shikaku mobile apps have been developed [7]. A Shikaku puzzle consists of a rectangular grid of size $m \times n$, with some cells containing a number. The objective of this puzzle is to partition the grid into rectangles such that each rectangle contains exactly one number, which must be equal to the area of that rectangle (see Figure 1). Determining whether a given Shikaku puzzle has a solution is an NP-complete problem [23].

Suppose that Paimon, an expert in Shikaku, created a difficult Shikaku puzzle and challenged her friend Venti to solve it. After a while, Venti could not solve her puzzle and began to doubt whether the puzzle actually has a solution. Paimon wants to convince him that her puzzle indeed has a solution without revealing it (which would render the challenge pointless). To achieve this, Paimon needs a *zero-knowledge proof (ZKP)*.
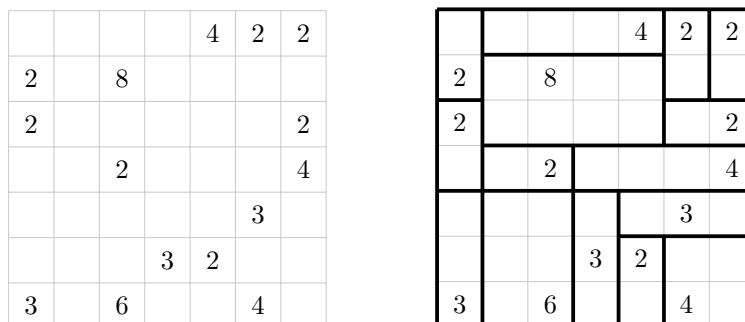

Figure 1 An example of a 7×7 Shikaku puzzle (left) and its solution (right).

1.1   Zero-Knowledge Proof

First introduced in 1989 by Goldwasser et al. [6], a ZKP is an interactive protocol between a prover P and a verifier V. Both P and V are given a computational problem x, but only P knows a solution w of x. A ZKP enables P to convince V that he/she knows w without revealing any information about w. It must satisfy the following three properties.

1. **Completeness:** If P knows w, then V accepts with high probability. (In this paper, we consider only the *perfect completeness* property where V always accepts.)
2. **Soundness:** If P does not know w, then V rejects with high probability. (In this paper, we consider only the *perfect soundness* property where V always rejects.)
3. **Zero-knowledge:** V learns nothing about w. Formally, there exists a probabilistic polynomial time algorithm S (called a *simulator*), not knowing w but having an access to V, such that the outputs of S follow the same probability distribution as the ones of the real protocol.

As there exists a ZKP for every NP problem [5], one can construct a computational ZKP for Shikaku. However, such construction requires cryptographic primitives and thus is not intuitive or practical.

Instead, many results so far aimed to develop physical ZKP protocols using a deck of playing cards. These card-based protocols have benefits that they use only portable objects found in everyday life and do not require computers. They also allow external observers to verify that the prover truthfully executes the protocol (which is often a challenging task for digital protocols). In addition, these protocols have great didactic values to teach the concept of a ZKP to non-experts.

1.2   Related Work

Card-based ZKP protocols for many other popular pencil puzzles have been developed, including Sudoku [8, 17, 21], Nonogram [3, 16], Akari [1], Takuzu [1, 12], Kakuro [1, 13], KenKen [1], Makaro [2], Norinori [4], Slitherlink [11], Juosan [12], Numberlink [18], Suguru [14], Ripple Effect [19], Nurikabe [15], Hitori [15], Cryptarithmetic [9], and Bridges [20].

In a recent work of Robert et al. [15], the authors posed an open problem to extend the idea of their protocol to verify a solution of Shikaku or other puzzles that require to draw rectangles with certain sizes in a grid.

1.3   Our Contribution

In this paper, we answer the open problem posed by Robert et al. [15] by developing two card-based ZKP protocols with perfect completeness and soundness for Shikaku: a brute force protocol and a more elegant, intuitive *flooding protocol*. The two protocols use $\Theta(m^2n^2)$ cards and $\Theta(mn)$ cards, respectively.

Most importantly, in the flooding protocol we develop a general technique to physically verify a rectangle-shaped area with a certain size in a rectangular grid, which can be used to verify other problems with similar constraints.

2   First Attempt: Brute Force Protocol

Every card used in this paper has an integer on the front side. All cards have indistinguishable back sides denoted by $\boxed{?}$.

Let (x, y) denote a cell located in the x-th topmost row and y-th leftmost column of the Shikaku grid. Let $p_2, p_3, ..., p_{k+1}$ be the k numbers written on the grid[1], with each number p_i in a cell (x_i, y_i). Note that we must have $p_2 + p_3 + ... + p_{k+1} = mn$.

Suppose that in P's solution, the grid is divided into k rectangles $Z_2, Z_3, ..., Z_{k+1}$ such that each Z_i contains the number p_i. Each rectangle Z_i is represented by its top-left and bottom-right corner cells (a_i, b_i) and (a_i', b_i'), respectively. To verify that the solution is correct, it is sufficient to show that

1. $a_i \leq x_i \leq a_i'$ and $b_i \leq y_i \leq b_i'$ (a cell with the number p_i is inside Z_i) for every $i \in \{2, 3, ..., k+1\}$,

2. $(a_i' - a_i + 1)(b_i' - b_i + 1) = p_i$ (the area of Z_i is equal to p_i) for every $i \in \{2, 3, ..., k+1\}$, and

3. $a_i' < a_j$ or $a_j' < a_i$ or $b_i' < b_j$ or $b_j' < b_i$ (Z_i and Z_j do not overlap) for every distinct $i, j \in \{2, 3, ..., k+1\}$.

These three conditions can be verified by applying the combination of the copy, addition, multiplication, and equality protocols [20], and a protocol to compare two numbers [2].

This protocol, however, involves a lot of messy calculations and thus has lost its didactic values as it becomes more computational and less intuitive. Moreover, it requires up to $\Theta(m^2 n^2)$ cards (as we have to multiply integers in modulo $mn)^2$, which is far too many to be practical. Instead, we are looking for an elegant and intuitive protocol that uses a reasonable number of cards.

3 Verifying an Area of Connected Cells

In a recent work, Robert et al. [15] developed a *sea formation protocol* that allows the prover P to convince the verifier V that a given area in a grid consists of t cells that are connected to each other horizontally or vertically. We will first show the necessary subprotocols and then explain the sea formation protocol.

3.1 Pile-Shifting Shuffle

Given a $p \times q$ matrix of cards, a *pile-shifting shuffle* rearranges the columns of the matrix by a random cyclic shift, i.e. shifts the columns cyclically to the right by x columns for a uniformly random $x \in \mathbb{Z}/q\mathbb{Z}$, unknown to all parties.

The pile-shifting shuffle was developed by Shinagawa et al. [22]. It can be performed in real world by putting the cards in each column into an envelope and then taking turns to apply *Hindu cuts* (taking several envelopes from the bottom and putting them on the top) to the sequence of envelopes [24].

3.2 Chosen Cut Protocol

Given a sequence of q face-down cards $C = (c_1, c_2, ..., c_q)$, a *chosen cut protocol* for q cards allows P to select a card c_i he/she wants (to use in other operations) without revealing i to V. This protocol also reverts the sequence C back to its original state after P finishes using c_i. It was developed by Koch and Walzer [10].

---

[1] We intentionally start the indices at 2 so that our second protocol, which will be introduced later, will be easier to understand.

[2] In this protocol, an integer x in modulo mn is encoded by a sequence of mn consecutive cards, with all of them being $\boxed{0}$s except the $(x+1)$-th leftmost card being a $\boxed{1}$.
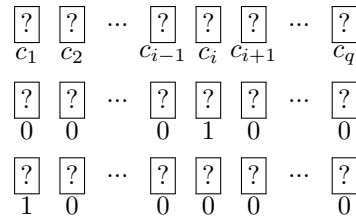
Figure 2 A $3 \times q$ matrix M constructed in Step 1 of the chosen cut protocol.

1. Construct the following $3 \times q$ matrix M (see Figure 2).
   a. In Row 1, publicly place the sequence C.
   b. In Row 2, secretly place a face-down $\boxed{1}$ at Column i and a face-down $\boxed{0}$ at each other column.
   c. In Row 3, secretly place a face-down $\boxed{1}$ at Column 1 and a face-down $\boxed{0}$ at each other column.
2. Apply the pile-shifting shuffle to M.
3. Turn over all cards in Row 2. Locate the position of the only $\boxed{1}$. A card in Row 1 directly above that $\boxed{1}$ will be the card c_i as desired.
4. After we finish using c_i in other operations, place c_i back into M at the same position.
5. Turn over all face-up cards in Row 2 and apply the pile-shifting shuffle to M again.
6. Turn over all cards in Row 3. Locate the position of the only $\boxed{1}$. Shift the columns of M cyclically such that this $\boxed{1}$ moves to Column 1. This reverts M back to its original state.

Note that Steps 3 and 6 of this protocol guarantee that the cards in Row 2 and Row 3 are in a correct format (each row having one $\boxed{1}$ and $q - 1$ $\boxed{0}$s).

3.3 Sea Formation Protocol

First, publicly place a face-down $\boxed{0}$ on every cell in the Shikaku grid. To handle the case where a selected cell is on the edge of the grid, we publicly place face-down "dummy cards" $\boxed{-1}$s around the grid. We now have an $(m + 2) \times (n + 2)$ matrix of cards (see Figure 3).
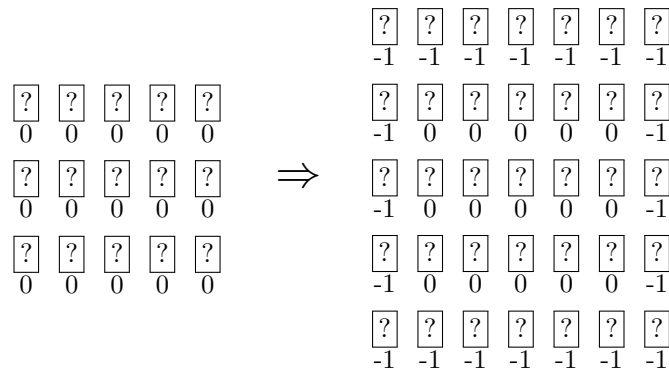
Figure 3 The way we place cards on a 3×5 Shikaku grid during the setup of the sea formation protocol.

Start at the top-left corner of the matrix and pick all cards in the order from left to right in Row 1, then from left to right in Row 2, and so on. Arrange them into a single sequence $D = (d_1, d_2, ..., d_{(m+2)(n+2)})$. Note that we know exactly where the four neighbors of any

given card are. Namely, the cards on the neighbor to the left, right, top, and bottom of a cell containing d_i are d_{i-1}, d_{i+1}, d_{i-n-2}, and d_{i+n+2}, respectively.

The sea formation protocol to verify a connected area with size t works as follows.

1. P applies the chosen cut protocol for $(m+2)(n+2)$ cards to select a $\boxed{0}$ that he/she wants to replace.
2. P reveals the selected card to V that it is a $\boxed{0}$ (otherwise V rejects) and then replaces it with a $\boxed{1}$.
3. P repeatedly performs the following steps for $t-1$ iterations.
   a. P applies the chosen cut protocol for $(m+2)(n+2)$ cards to select a $\boxed{1}$ he/she wants.
   b. P reveals the selected card to V that it is a $\boxed{1}$ (otherwise V rejects).
   c. P picks the four neighbors of the selected card and applies the chosen cut protocol for four cards to select one of the four neighbors, which is a $\boxed{0}$ that he/she wants to replace.
   d. P reveals the selected neighbor to V that it is a $\boxed{0}$ (otherwise V rejects) and then replaces it with a $\boxed{1}$.

We can see that in each iteration, the "sea" of $\boxed{1}$s expands by one cell, while all $\boxed{1}$s remain connected to each other. Therefore, after $t-1$ steps, V is convinced that there is an area of t $\boxed{1}$s in the grid that are connected to each other.

## 4   Idea to Verify a Rectangle-Shaped Area

The sea formation protocol, however, does not say anything about the shape of the area. By extending the idea of the sea formation protocol, we propose the following *flooding protocol*, which allows P to convince V that the area is a rectangle with size t.

The idea is to always start at the top-left corner of the rectangle. At first, P changes the card on the top-left corner cell of the rectangle from a $\boxed{0}$ to a $\boxed{1}$. Similarly to the sea formation protocol, in each step P selects a cell with a $\boxed{1}$ and changes the card on one of its neighbor from a $\boxed{0}$ to a $\boxed{1}$. However, the difference from the sea formation protocol is that P can only select the neighbor to the right or to the bottom (but not to the left or to the top). We call this process a *flood*, which starts at the top-left corner and goes downwards or rightwards in each step until it eventually fills the whole rectangle in $t-1$ steps.

To be more specific, at first the flood can only go downwards (i.e. P can only select the neighbor to the bottom) to fill cells along the left edge of the rectangle. Then, right after it just filled all cells along the left edge, the flood suddenly changes direction and can only go rightwards (i.e. P can only select the neighbor to the right) to fill the rest of the cells in the rectangle. In particular, V must not know the exact time when the flood changes direction (otherwise V will know the height of the rectangle).

The technique to achieve this "one-time direction change" is to let P keep a secret variable r, which controls the direction of the flood (if $r=0$, then the flood goes downwards; if $r=1$, then the flood goes rightwards). At the beginning, P shows V that $r=0$. Before each step, P secretly chooses whether to add 1 to r or not, then shows V that $r \neq 2$ (without revealing the actual value of r). This technique works because while $r=0$, r can become either 0 or 1 in the next step, but once r becomes 1, it must remain 1 forever (see a subprotocol in Section 5.2 on how to make the selected neightbor depend on the value of r).

P performs the above process for $t - 1$ times to change all $\boxed{0}$s in the rectangle to $\boxed{1}$s. However, the protocol is not finished yet, as V is not yet convinced that the area is a rectangle. In fact, P has only shown that the area has a straight left edge; it may look like one of the shapes in Figure 4.
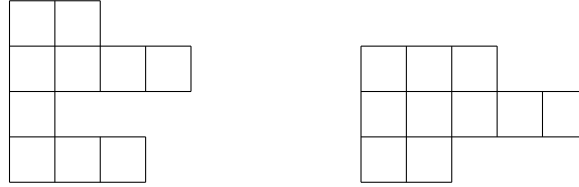
■ **Figure 4** Examples of possible shapes with a straight left edge, each with area 10.

To convince V that the area is a rectangle, P needs to perform the "second flood". The second flood starts at the bottom-right corner and goes into the cells already visited by the "first flood" in the opposite direction from the first flood – originally the flood can only go upwards (i.e. P can only select the neighbor to the top), then right after it just filled all cells along the right edge, the flood changes direction and can only go leftwards (i.e. P can only select the neighbor to the left).

Formally, P starts at a bottom-right corner of the rectangle and replaces a $\boxed{1}$ with a $\boxed{2}$. P sets $r = 0$ and shows it to V. In each step, P secretly chooses whether to add 1 to r or not, then shows V that $r \neq 2$. If $r = 0$ (resp. $r = 1$), P selects a cell with a $\boxed{2}$ and changes the card on its neighbor to the top (resp. to the left) from a $\boxed{1}$ to a $\boxed{2}$. P performs this for $t - 1$ steps to change all $\boxed{1}$s in the rectangle to $\boxed{2}$s.

After the second flood, P have shown that the area also has a straight right edge. This is sufficient to convince V that the area is a rectangle with size t (see the proof of Lemma 2 for the full proof of perfect soundness).

In the next section, we will show the necessary subprotocols that enable us to formalize this idea into an actual protocol.

5 Subprotocols

5.1 Addition Protocol for $\mathbb{Z}/3\mathbb{Z}$

We use a sequence of three consecutive cards to encode each integer in $\mathbb{Z}/3\mathbb{Z}$. Namely, we use $\boxed{1}\boxed{0}\boxed{0}$, $\boxed{0}\boxed{1}\boxed{0}$, and $\boxed{0}\boxed{0}\boxed{1}$ to encode 0, 1, and 2, respectively.

Suppose we have sequences R and S encoding integers r and s in $\mathbb{Z}/3\mathbb{Z}$, respectively. This protocol, developed by Shinagawa et al. [22], computes the sum $r + s$ without revealing r or s.

1. Swap the two rightmost cards of S. This modified sequence, called S', now encodes $-s$ (mod 3).
2. Construct a 2×3 matrix M by placing S' in Row 1 and R in Row 2.
3. Apply the pile-shifting shuffle to M. Note that Row 1 and Row 2 of M now encode $-s + x$ (mod 3) and $r + x$ (mod 3), respectively, for some uniformly random $x \in \mathbb{Z}/3\mathbb{Z}$.
4. Turn over all cards in Row 1 of M. Locate the position of a $\boxed{1}$. Shift the columns of M cyclically such that this $\boxed{1}$ moves to Column 1.
5. The sequence in Row 2 of M now encodes $(r + x) - (-s + x) \equiv r + s$ (mod 3) as desired.

Note that Step 4 of this protocol guarantees that S is in a correct format (having one $\boxed{1}$ and two $\boxed{0}$s). In each step of the flooding protocol, P secretly selects $s \in \{0, 1\}$ and places S accordingly. Then, P reveals the rightmost card of S that it is a $\boxed{0}$ to show V that $s \neq 2$. Similarly, after computing the sum $r + s$, P reveals the rightmost card of the resulting sequence to show V that $r + s \neq 2$.

5.2 Neighbor Selection Protocol

In $\mathbb{Z}/2\mathbb{Z}$, we use $\boxed{1}\boxed{0}$ and $\boxed{0}\boxed{1}$ to encode 0 and 1, respectively. Suppose we have two face-down cards c_0 and c_1, and a sequence R encoding an integer $r \in \mathbb{Z}/2\mathbb{Z}$. We want to select a card c_r to use in other operations without revealing r, and also put c_0 and c_1 back to where they came from.

We can do so by applying the chosen cut protocol for two cards. However, in Step 1.b, we instead place a sequence R in Row 2 (without revealing R). Also, at the end of the chosen cut protocol, M is reverted to its original state, so we can put c_0 and c_1 back to where they came from.

In each step of the flooding protocol, after showing that $r \neq 2$, P picks only the two leftmost cards of a sequence encoding r in $\mathbb{Z}/3\mathbb{Z}$. This truncated sequence encodes r in $\mathbb{Z}/2\mathbb{Z}$ as desired. During the first flood, P chooses the cards on the neighbor to the bottom and to the right of the selected cell as c_0 and c_1, respectively; during the second flood, P chooses the cards on the neighbor to the top and to the left of the selected cell as c_0 and c_1, respectively.

6 Formal Steps of the Flooding Protocol

Similarly to the sea formation protocol, we first publicly place a face-down $\boxed{0}$ on every cell in the Shikaku grid, and also place face-down $\boxed{\text{-}1}$s around the grid. We now have an $(m + 2) \times (n + 2)$ matrix of cards (see Figure 3).

Let $h_i = a_i' - a_i + 1$ be the height of a rectangle Z_i ($i \in \{2, 3, ..., k + 1\}$). To verify that Z_i is a rectangle with area p_i and also contains a cell with the number p_i, P performs the following two phases: the first flood and the second flood.

6.1 First Flood

1. P applies the chosen cut protocol for $(m + 2)(n + 2)$ cards to select a card on the top-left corner cell of Z_i.
2. P reveals the selected card to V that it is a $\boxed{0}$ (otherwise V rejects) and then replaces it with a $\boxed{1}$.
3. P publicly constructs a sequence R of three cards encoding an integer $r = 0$.
4. P repeatedly performs the following steps for $p_i - 1$ iterations.
   a. P secretly constructs a sequence S of three cards encoding an integer $s \in \{0, 1\}$. If this is the h_i-th iteration, P must choose $s = 1$; otherwise, P must choose $s = 0$.
   b. P reveals the rightmost card of S to V that it is a $\boxed{0}$ to show that $s \neq 2$ (otherwise V rejects).
   c. P applies the addition protocol to compute $r + s$ and reveals the rightmost card of the resulting sequence to V that it is a $\boxed{0}$ to show that $r + s \neq 2$ (otherwise V rejects). From now on, set $r := r + s$.
   d. P applies the chosen cut protocol for $(m + 2)(n + 2)$ cards to select a $\boxed{1}$ he/she wants from the Shikaku grid. If this is during the first $h_i - 1$ iterations, P must choose the bottommost $\boxed{1}$; otherwise, P may choose any card that is the rightmost $\boxed{1}$ in its row and is not located in the rightmost column of Z_i.

  e. P reveals the selected card to V that it is a $\boxed{1}$ (otherwise V rejects).
  f. P chooses the neighbors to the bottom and to the right of the selected card as c_0 and c_1, respectively, and applies the neighbor selection protocol to select a card c_r (using the two leftmost cards of a sequence encoding r as inputs).
  g. P reveals the selected neighbor to V that it is a $\boxed{0}$ (otherwise V rejects) and then replaces it with a $\boxed{1}$.

  After the first flood, all cards on the cells in Z_i are now changed to $\boxed{1}$s.

6.2 Second Flood

1. P applies the chosen cut protocol for $(m+2)(n+2)$ cards to select a card on the bottom-right corner cell of Z_i.
2. P reveals the selected card to V that it is a $\boxed{1}$ (otherwise V rejects) and then replaces it with an $\boxed{i}$.
3. P publicly constructs a sequence R of three cards encoding an integer $r = 0$.
4. P repeatedly performs the following steps for $p_i - 1$ iterations.
  a. P secretly constructs a sequence S of three cards encoding an integer $s \in \{0, 1\}$. If this is the h_i-th iteration, P must choose $s = 1$; otherwise, P must choose $s = 0$.
  b. P reveals the rightmost card of S to V that it is a $\boxed{0}$ to show that $s \neq 2$ (otherwise V rejects).
  c. P applies the addition protocol to compute $r + s$ and reveals the rightmost card of the resulting sequence to V that it is a $\boxed{0}$ to show that $r + s \neq 2$ (otherwise V rejects). From now on, set $r := r + s$.
  d. P applies the chosen cut protocol for $(m+2)(n+2)$ cards to select an $\boxed{i}$ he/she wants from the Shikaku grid. If this is during the first $h_i - 1$ iterations, P must choose the topmost $\boxed{i}$; otherwise, P may choose any card that is the leftmost $\boxed{i}$ in its row and is not located in the leftmost column of Z_i.
  e. P reveals the selected card to V that it is an $\boxed{i}$ (otherwise V rejects).
  f. P chooses the neighbors to the top and to the left of the selected card as c_0 and c_1, respectively, and applies the neighbor selection protocol to select a card c_r (using the two leftmost cards of a sequence encoding r as inputs).
  g. P reveals the selected neighbor to V that it is a $\boxed{1}$ (otherwise V rejects) and then replaces it with an $\boxed{i}$.

  After the second flood, all cards on the cells in Z_i are now changed to $\boxed{i}$s. Finally, P turns over a card on the cell with the number p_i to show that it is an $\boxed{i}$, i.e. Z_i contains the cell with the number p_i (otherwise V rejects).

  P performs the above two phases for every $i \in \{2, 3, ..., k+1\}$. If all verification steps pass, then V accepts.

  The number of cards used in the flooding protocol is $\Theta(mn)$, which is much lower than the brute force protocol.

7 Proof of Security

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

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

Proof. Suppose that P knows a solution of the puzzle. Consider the verification of each Z_i.

In the first flood, during the first $h_i - 1$ iterations P chooses $s = 0$ and chooses the bottommost $\boxed{1}$, so the area of $\boxed{1}$s expands downwards by one cell. After $h_i - 1$ iterations, all cards along the left edge of Z_i have been changed to $\boxed{1}$s. In the h_i-th iteration, P chooses $s = 1$ and chooses any $\boxed{1}$, so the flood direction is changed to rightwards and the area of $\boxed{1}$s expands by one cell. After that, in each iteration P chooses $s = 0$ and chooses any card that is the rightmost $\boxed{1}$ in its row and is not located in the rightmost column of Z_i, so the area of $\boxed{1}$s expands by one cell inside Z_i. Therefore, at the end of the first flood, all cards in Z_i has been changed to $\boxed{1}$s.

Analogously, in the second flood, during the first $h_i - 1$ iterations P chooses $s = 0$ and chooses the topmost $\boxed{i}$, so the area of $\boxed{i}$s expands upwards by one cell. After $h_i - 1$ iterations, all cards along the right edge of Z_i have been changed to $\boxed{i}$s. In the h_i-th iteration, P chooses $s = 1$ and chooses any $\boxed{i}$, so the flood direction is changed to leftwards and the area of $\boxed{i}$s expands by one cell. After that, in each iteration P chooses $s = 0$ and chooses any card that is the leftmost $\boxed{i}$ in its row and is not located in the leftmost column of Z_i, so the area of $\boxed{i}$s expands by one cell inside Z_i. Therefore, at the end of the first flood, all cards in Z_i has been changed to $\boxed{i}$s, thus a card on the cell containing the number p_i must also be an $\boxed{i}$.

Since the verification passes for every Z_i, V always accepts. ◀

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

Proof. We will prove the contrapositive of this statement. Suppose that V accepts, meaning that the flooding protocol passes for every Z_i. We will prove that P must know a solution.

First, note that the chosen cut protocol in Section 3.2 and the addition protocol in Section 5.1 guarantee that the inputs from P must be in a correct format. Consider the verification of Z_i. Suppose that the first flood goes downwards for $h - 1$ steps before changing direction to rightwards. The area that contains $\boxed{1}$s after the first flood must have a straight left edge with height h, and have a shape like h horizontal bars placing on top of each other. Let $\ell_1, \ell_2, ..., \ell_h$ be the length of these bars from top to bottom. For example, in Figure 5 we have $h = 4$, $\ell_1 = 2$, $\ell_2 = 4$, $\ell_3 = 1$, and $\ell_4 = 3$.
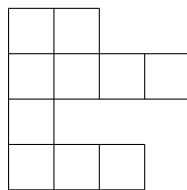
■ **Figure 5** An example of a possible shape with a straight left edge.

Since all p_i $\boxed{1}$s in this area have been replaced by $\boxed{i}$s after the second flood, all cells in the area must be reachable from the starting point of the second flood by moving only upwards or leftwards. Thus, the only possible starting point of the second flood is the rightmost cell of the bottommost bar (the one with length ℓ_h).

Moreover, for any $j < h$, we must have $\ell_j \leq \ell_h$ (otherwise there is a cell in the j-th bar which is located to the right of the starting point and thus not reachable by the second flood) However, if $\ell_j < \ell_h$, the second flood cannot go directly from the starting point to the j-th

bar by only moving upwards; it has to change direction at least twice, a contradiction since the flood can change direction at most once. Therefore, we must have $\ell_j = \ell_h$ for every $j \in \{1, 2, ..., h-1\}$, which means the area must be a rectangle.

Therefore, Z_i is a rectangle with area p_i that contains a cell with the number p_i for every $i \in \{2, 3, ..., k+1\}$. Since any two rectangles do not overlap, and $p_2 + p_3 + ... + p_{k+1} = mn$, they must be a partition of the grid. Hence, we can conclude that P knows a valid solution of the puzzle. ◀

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

Proof. To prove the zero-knowledge property, 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 6 of the chosen cut protocol in Section 3.2, the $\boxed{1}$ has an equal probability to be at any of the q positions, so this step can be simulated by S.
- In Step 4 of the addition protocol in Section 5.1, the $\boxed{1}$ has an equal probability to be at any of the three positions, so this step can be simulated by S.
- In the flooding protocol, during the verification of each Z_i, there is only one deterministic pattern of the cards that are turned face-up. This pattern solely depends on p_i, which is public information, so the whole protocol can be simulated by S. ◀

8   Future Work

We developed a physical ZKP protocol with perfect completeness and soundness for Shikaku using $\Theta(mn)$ cards. Most importantly, we also developed a general technique to physically verify a rectangle-shaped area with a certain size in a rectangular grid.

A possible future work is to develop physical ZKP protocols to verify other geometric shapes or other puzzles with constraints related to shapes (e.g. Shakashaka). Another interesting future work is to develop an equivalent protocol for Shikaku that can be implemented using a deck of all different cards (like the one for Sudoku [17]).

References

1   X. Bultel, J. Dreier, J.-G. Dumas, and P. Lafourcade. Physical zero-knowledge proofs for Akari, Takuzu, Kakuro and KenKen. In *Proceedings of the 8th International Conference on Fun with Algorithms (FUN)*, pages 8:1–8:20, 2016. `doi:10.4230/LIPIcs.FUN.2016.8`.

2   X. Bultel, J. Dreier, J.-G. Dumas, P. Lafourcade, D. Miyahara, T. Mizuki, A. Nagao, T. Sasaki, K. Shinagawa, and H. Sone. Physical zero-knowledge proof for Makaro. In *Proceedings of the 20th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 111–125, 2018. `doi:10.1007/978-3-030-03232-6_8`.

3   Y.-F. Chien and W.-K. Hon. Cryptographic and physical zero-knowledge proof: From Sudoku to Nonogram. In *Proceedings of the 5th International Conference on Fun with Algorithms (FUN)*, pages 102–112, 2010. `doi:10.1007/978-3-642-13122-6_12`.

4   J.-G. Dumas, P. Lafourcade, D. Miyahara, T. Mizuki, T. Sasaki, and H. Sone. Interactive physical zero-knowledge proof for Norinori. In *Proceedings of the 25th International Computing and Combinatorics Conference (COCOON)*, pages 166–177, 2019. `doi:10.1007/978-3-030-26176-4_14`.

5   O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design. *Journal of the ACM*, 38(3):691–729, 1991. `doi:10.1145/116825.116852`.

6   S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. `doi:10.1137/0218012`.

7   Google Play: Shikaku. `https://play.google.com/store/search?q=Shikaku&c=apps`.

8   R. Gradwohl, M. Naor, B. Pinkas, and G.N. Rothblum. Cryptographic and physical zero-knowledge proof systems for solutions of sudoku puzzles. *Theory of Computing Systems*, 44(2):245–268, 2009. `doi:10.1007/s00224-008-9119-9`.

9   R. Isuzugawa, D. Miyahara, and T. Mizuki. Zero-knowledge proof protocol for crypt-arithmetic using dihedral cards. In *Proceedings of the 19th International Conference on Unconventional Computation and Natural Computation (UCNC)*, pages 51–67, 2021. `doi:10.1007/978-3-030-87993-8_4`.

10   A. Koch and S. Walzer. Foundations for actively secure card-based cryptography. In *Proceedings of the 10th International Conference on Fun with Algorithms (FUN)*, pages 17:1–17:23, 2020. `doi:10.4230/LIPIcs.FUN.2021.17`.

11   P. Lafourcade, D. Miyahara, T. Mizuki, L. Robert, T. Sasaki, and H. Sone. How to construct physical zero-knowledge proofs for puzzles with a "single loop" condition. *Theoretical Computer Science*, 888:41–55, 2021. `doi:10.1016/j.tcs.2021.07.019`.

12   D. Miyahara, L. Robert, P. Lafourcade, S. Takeshige, T. Mizuki, K. Shinagawa, A. Nagao, and H. Sone. Card-based zkp protocols for takuzu and juosan. In *Proceedings of the 10th International Conference on Fun with Algorithms (FUN)*, pages 20:1–20:21, 2020. `doi:10.4230/LIPIcs.FUN.2021.20`.

13   D. Miyahara, T. Sasaki, T. Mizuki, and H. Sone. Card-based physical zero-knowledge proof for Kakuro. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E102.A(9):1072–1078, 2019. `doi:10.1587/transfun.E102.A.1072`.

14   L. Robert, D. Miyahara, P. Lafourcade, and T. Mizuki. Physical zero-knowledge proof for suguru puzzle. In *Proceedings of the 22nd International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, pages 235–247, 2020. `doi:10.1007/978-3-030-64348-5_19`.

15   L. Robert, D. Miyahara, P. Lafourcade, and T. Mizuki. Interactive physical zkp for connectivity: Applications to nurikabe and hitori. In *Proceedings of the 17th Conference on Computability in Europe (CiE)*, pages 373–384, 2021. `doi:10.1007/978-3-030-80049-9_37`.

16   S. Ruangwises. An improved physical zkp for nonogram. In *Proceedings of the 15th Annual International Conference on Combinatorial Optimization and Applications (COCOA)*, pages 262–272, 2021. `doi:10.1007/978-3-030-92681-6_22`.

17   S. Ruangwises. Two standard decks of playing cards are sufficient for a zkp for sudoku. In *Proceedings of the 27th International Computing and Combinatorics Conference (COCOON)*, pages 631–642, 2021. `doi:10.1007/978-3-030-89543-3_52`.

18   S. Ruangwises and T. Itoh. Physical zero-knowledge proof for numberlink puzzle and k vertex-disjoint paths problem. *New Generation Computin*, 39(1):3–17, 2021. `doi:10.1007/s00354-020-00114-y`.

19   S. Ruangwises and T. Itoh. Physical zero-knowledge proof for ripple effect. *Theoretical Computer Science*, 895:115–123, 2021. `doi:10.1016/j.tcs.2021.09.034`.

20   S. Ruangwises and T. Itoh. 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)*, pages 149–163, 2021. `doi:10.1007/978-3-030-87993-8_10`.

21   T. Sasaki, D. Miyahara, T. Mizuki, and H. Sone. Efficient card-based zero-knowledge proof for sudoku. *Theoretical Computer Science*, 839:135–142, 2020. `doi:10.1016/j.tcs.2020.05.036`.

22   K. Shinagawa, T. Mizuki, J.C.N. Schuldt, K. Nuida, N. Kanayama, T. Nishide, G. Hanaoka, and E. Okamoto. Card-based protocols using regular polygon cards. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E100.A(9):1900–1909, 2017. `doi:10.1587/transfun.E100.A.1900`.

23   Y. Takenaga, S. Aoyagi, S. Iwata, and T. Kasai. Shikaku and ripple effect are np-complete. *Congressus Numerantium*, 216:119–127, 2013.

24   I. Ueda, D. Miyahara, A. Nishimura, Y. Hayashi, T. Mizuki, and H. Sone. Secure implementations of a random bisection cut. *International Journal of Information Security*, 19(4):445–452, 2020. `doi:10.1007/s10207-019-00463-w`.