Str8tsに対するカードベースゼロ知識証明

本間 日綺[1,a)]　品川 和雅[2,3,b)]

概要：パズルに対するカードベースゼロ知識証明プロトコルとは、パズルの解を知っている証明者が、解に関する情報を一切明かすことなく、解が存在することをカードによる操作で検証者に納得させるプロトコルである。本稿では、Str8ts（ストレイツ）に対するカードベースゼロ知識証明プロトコルを提案する。Str8ts には、指定された区画内で数字が連番でなければならないという条件があり、これはパイルシフティングシャッフルによって検証できる。しかし、そのような検証を素直に実行した場合、パイルシフティングシャッフルの回数はパズルインスタンスに依存した値になる。そこで我々は複数回のパイルシフティングシャッフルを 2 回のパイルスクランブルシャッフルで実行する巡回バッチングという新しい技術を提案する。提案プロトコルは、巡回バッチングを用いることにより、3 回のパイルスクランブルシャッフルによって実行できる。

キーワード：カードベース暗号、ゼロ知識証明、Str8ts

Card-Based Zero-Knowledge Proof for Str8ts

Haruki Homma[1,a)]　Kazumasa Shinagawa[2,3,b)]

Abstract: A card-based zero-knowledge proof protocol for a puzzle is a protocol in which a prover, who knows a solution to the puzzle, convinces a verifier of the existence of a valid solution using card-based operations without revealing any information about the solution itself. In this paper, we propose a card-based zero-knowledge proof protocol for the puzzle Str8ts. A key constraint in Str8ts is that the numbers within designated compartments must form consecutive sequences, which can be verified through pile-shifting shuffles. However, if this verification is performed straightforwardly, the number of pile-shifting shuffles required depends on the specific puzzle instance. To address this issue, we introduce a novel technique called *cyclic batching*, which simulates multiple pile-shifting shuffles using just two pile-scramble shuffles. By employing cyclic batching, our protocol can be executed with only three pile-scramble shuffles.

Keywords: Card-based cryptography, Zero-knowledge proof, Str8ts

1. はじめに

1.1 Str8ts

Str8ts [4, 5] は、Jeff Widderich と Andrew Stuart が共同で考案したペンシルパズルである。本パズルの性質から、一種の変種数独と考えることができる。盤面には 9×9 個のマスが存在し、初期状態として、マス内のにいくつかの数字が書かれている。さらに、数字を置くことができない黒マスも点在している。各行および列には、白マスの連なりで構成された区画がいくつか存在し、この区画を「ブロック」とする。また、白マスのうち、初期状態で数字が与えられているものを「制約白マス」とし、与えられていないものを「空白マス」とする。一方、黒マスのうち、数字が定まっているものを特に「制約黒マス」とする。解答者は以下の 3 つの条件を満たすように 1 から 9 の数字をす

[1] 茨城大学
Ibaraki University
[2] 筑波大学
University of Tsukuba
[3] 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology
[a)] 22t4071l@vc.ibaraki.ac.jp
[b)] shinagawa@cs.tsukuba.ac.jp

べての白マスに満たせばパズルの解が完成する。

ユニーク条件　各行および各列について、重複しないように白マスの数字を定める。

制約黒マス条件　制約黒マスに数字がある場合、その制約黒マスが属する行および列で同じ数字を使わない。

ストレート条件　各ブロックについて、白マスは連続した数字の並びである（ただし、数字の順序は問わない）。

例題を Samples Document [4] 表紙から引用し、図 1 に示す。この例題に対する答えからブロックを確認する。1 行目について、6,7 の入った白マスと 3,2 の入った白マスはそれぞれ別のブロックである。2 行目について、8 マスの白マスが連なるが、行全体がブロックである。

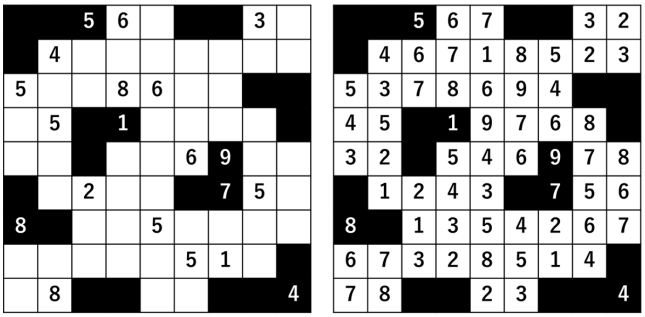


図 1　Str8ts の問題 (左) とその解答 (右)

1.2　ゼロ知識証明（ZKP）

ゼロ知識証明とは、Goldwasser–Mikali–Rackoff [1] が提案した暗号技術である。証明者 P は自身が持つ主張が真であることを、真であること以外のいかなる情報も与えずに検証者 V に伝える技術である。

特に、「パズルの解を知っている」という主張に対するゼロ知識証明をパズルに対するゼロ知識証明という。言い換えると、証明者がパズルの解を知っていることを、答えに関する一切の情報を与えることなく、検証者に納得させる証明である。

証明は、以下の 3 つの条件を満たす必要がある。

完全性　P が解を知っているならば、V は必ず証明を受理する。

健全性　P が解を知らないならば、V は必ず証明を棄却する。

ゼロ知識性　V は解の存在以外に、解に関する一切の情報を得ない。

一方、カードやスリーブなどの身の回りにある道具を用いて、秘密計算を行うカードベース暗号と呼ばれる研究分野が存在する。カードベースゼロ知識証明とは、このような物理的な道具を用いた証明方法である。

1.3　関連する既存研究

今日までに、多くのパズルに対するゼロ知識証明が提案されてきた。特に、$n \times n$ 数独に対するカードベースゼロ知識証明 [2] とその効率化に関する研究 [8–10, 12, 13] が多く残されている。実行時間を短縮するために、プロトコルにおけるシャッフル回数の削減が進められた。また、扱う道具を単純化するために、カード枚数の削減も効率化の指標として考えられている。なお、シャッフル回数が減ると、カードで表現すべき情報量が増えるため、カード枚数が増える傾向がある。どのようなプロトコルでもパズルインスタンスに依存しない定数回のシャッフルで実行できることが望ましい。現在、数独に対するカードベース ZKP は非対話的なプロトコルに限れば、2 回のシャッフルだけで示せることがわかっている。したがって、数独の性質を持ち合わせた Str8ts も、それに準ずる定数回のシャッフル操作でプロトコルを実現できると考えられる。

1.4　貢献

本稿では、Str8ts に対する初のカードベースゼロ知識証明プロトコルを提案する。提案プロトコルではパズルインスタンスに依存せずに、3 回のシャッフル操作でプロトコルを実行する。今回、「巡回バッチング」と名付けるシャッフル操作を導入する。これは、並列的に実行される複数回のパイルシフティングシャッフルを 2 回のパイルスクランブルシャッフルで実現する技術である。巡回バッチングは 3 節で説明する。また、巡回バッチングで利用するバッチングの概要は 2 節で述べる。

2.　準備

プロトコルで使用するカードとシャッフル操作について述べる。

2.1　使用するカード

2.1.1　二色カード

表面にはハートまたはクラブの絵柄が書かれており、裏面はすべて「？」の絵柄が書かれている。

表：[♡][♣]　　裏：[?][?]

2 枚でコミットメントを表現した場合、以下のように 0 と 1 を符号化できる。

$$[♣][♡] := 0,\ [♡][♣] := 1$$

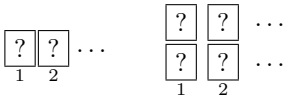
2.1.2　数字カード

表面には 0 以上の整数のいずれかが書かれており、裏面はすべて「？」の絵柄が書かれている。

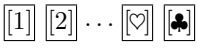
表：[0][1][2]$\cdots$　裏：[?][?][?]$\cdots$

2.2　カードの表記

カード (束) の下に小さく数字が書かれている場合、そのカード（束）列の数え番号とする。

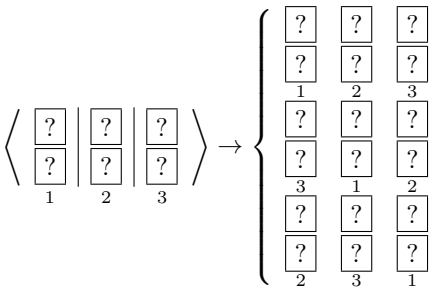


カードが裏向きのとき、表面の絵柄や数字を「[ ]」で囲んで表現する場合がある。



2.3 パイルシフティングシャッフル（PShift）

Shinagawa ら [11] はカード束の列に対して右へシフトさせる巡回的なシャッフル操作を提案した。同じカード枚数からなる q 個のカード束に対して、秘匿された一様ランダムな値 $r \in \mathbb{Z}/q\mathbb{Z}$ の回数シフトを繰り返す。2 枚からなる 3 個のカード束に対する PShift は次のようになる。

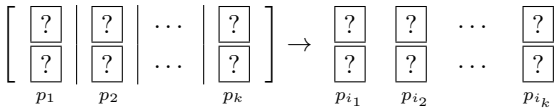


上記のように、3 通りのカード束の並びが考えられる。

なお、カード束のカード枚数が 1 枚のときを特にランダムカット（RC）と呼ぶ。

2.4 パイルスクランブルシャッフル（PSS）

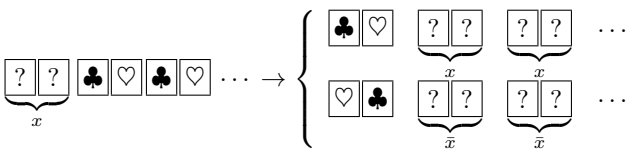
Ishikawa–Chida–Mizuki [3] は同じカード枚数からなるカード束の列に対するシャッフル操作を提案した。k 個のカード束に対して k 次対称群 S_k に属する一様ランダムな置換 π を適用する。カード束の列 $(p_1, p_2, ..., p_k)$ に対して PSS を適用すると、新たなカード束の列 $(p_{\pi^{-1}(1)}, p_{\pi^{-1}(2)}, ..., p_{\pi^{-1}(k)})$ が出力される。



ただし $i_\ell = \pi^{-1}(\ell)$（$1 \le \ell \le k$）である。

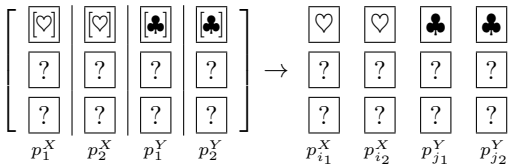
2.5 コピープロトコル

Mizuki–Sone [7] は、2 枚からなるコミットメントを秘匿しながらコピーする、コピープロトコルを提案した。k 個の ♣♡ を追加カードとして用いることで、出力として入力コミットメント x を k 個得ることができる。



2.6 バッチング

バッチング技術とは、並列的に実行される複数回の PSS を、追加カードを用いて、1 回の PSS で同時に行う技術である。各シャッフルのカード束に対して、それぞれのシャッフル群を識別できるように追加カードを用意する。例として、2 枚のカード束からなる 2 つの PSS 群 X, Y について、♡ と ♣ で識別した場合、以下のようにバッチングできる。

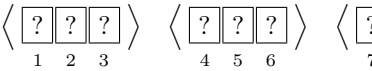


ここで $(i_1, i_2) \in \{(1,2),(2,1)\}$ と $(j_1, j_2) \in \{(1,2),(2,1)\}$ は一様ランダムかつ独立に選ばれる。以上のように、二色カードの値からカード束を並び替える。

3. 巡回バッチング

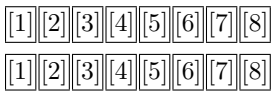
本節では、並列的に実行される複数回の PShift（または RC）を、追加カードを用いて 2 回の PSS に変換するための技術（巡回バッチング技術）を提案する。なお、1 回の PShift を 2 回の PSS に変換する方法は Miyamoto–Shinagawa [6] のグラフシャッフルプロトコルによって与えられている。巡回バッチング技術は、複数回のグラフシャッフルプロトコルに対してバッチング技術を適用したものである。

それでは巡回バッチング技術について説明する。具体例として、以下の 3 つの RC を実行したい状況を考える。

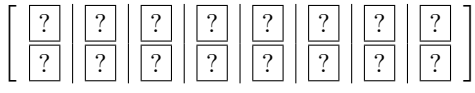


以下の手順により、これらを 2 回の PSS で実現する。

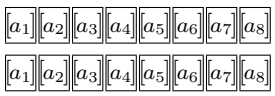
(1) 以下のようにカードを並べる。



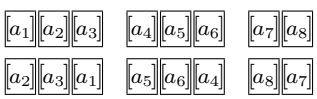
(2) 以下のように PSS を適用する。



シャッフルの結果、以下のように $(a_1, a_2, \ldots, a_8)$ の列が得られたものとする。

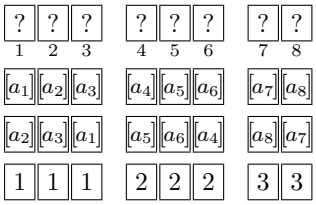


(3) 以下のように 3 つの束を作る。



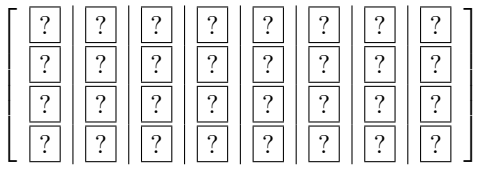
左の束は置換 (a_1, a_2, a_3) を表し、中央の束は置換 (a_4, a_5, a_6) を表し、右の束は置換 (a_7, a_8) を表す。

(4) 以下のようにカードを並べる。



ここで最上段は RC の適用先のカード列であり、最下段は何番目の RC かを識別するためのカード列である。

(5) 最下段のカードを伏せ、以下のように PSS を適用する。



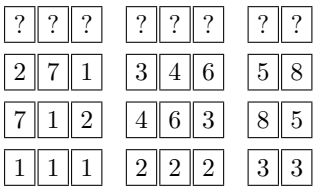
(6) 最上段を除くすべてのカードをめくり、以下の条件を満たすようにカード束ごと並べ替える。

(a) 最下段は $(1, 1, 1, 2, 2, 2, 3, 3)$ と昇順に並ぶ。

(b) 2 段目を $(b_1, b_2, \cdots, b_8)$、3 段目を $(c_1, c_2, \cdots, c_8)$ としたとき、以下を等式をすべて満たす。

- $c_1 = b_2, c_2 = b_3, c_3 = b_1$
- $c_4 = b_5, c_5 = b_6, c_6 = b_4$
- $c_7 = b_8, c_8 = b_7$

例えば、以下のように並べ替える。



最上段のカード列を出力する。左の 3 枚、中央の 3 枚、右の 2 枚が、それぞれの RC の出力結果に対応する。

以上が巡回バッチングの手順である。上記の例の場合、24 枚の追加カードと 2 回の PSS を用いる。一般の場合、適用したい PShift のパイルの総数を k とすると、$3k$ 枚の追加カードと 2 回の PSS を用いる。

なお、上記の例の場合、3 枚の RC が 2 つあるため、それらを識別するために最下段のカード列を要するが、もし PShift のパイル数がそれぞれ異なる場合は最下段のカード列は不要である。また、応用先によってはパイル数の等しい PShift の出力カード列を区別する必要がなく、それらが入れ替わった状態で出力されてもよい場合があり、その場合も最下段のカード列は不要である。これらの場合、追加カードは $2k$ 枚でよい。

4. 基本プロトコル

本節では、$n \times n$ の Str8ts に対応した、PSS と PShift を組み合わせた基本プロトコルの手順について述べる。4.2 節の入力以降は V が操作する。

4.1 Str8ts の用語の定義

行数と列数を n とし、ブロック数を m とする。i 行目（$1 \le i \le n$）の白マスの座標の集合を $\mathcal{C}^{(i)}$ とする。j 行目（$1 \le j \le n$）の白マスの座標の集合を $\mathcal{R}^{(j)}$ とする。k ブロック目（$1 \le k \le m$）の白マスの座標の集合を $\mathcal{B}^{(k)}$ とする。また、白マス全体の集合を $\mathcal{W}$ とおくと、以下の式が成り立つ。

$$\mathcal{W} = \bigcup_{1 \le i \le n} \mathcal{C}^{(i)} = \bigcup_{1 \le j \le n} \mathcal{R}^{(j)} = \bigcup_{1 \le k \le m} \mathcal{B}^{(k)}$$

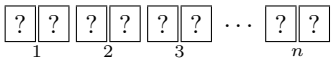
図 1 の場合、$n = 9$ および $m = 32$ であり、$\mathcal{C}^{(1)} = \{(1,4),(1,5),(1,8),(1,9)\}, \mathcal{B}^{(1)} = \{(1,4),(1,5)\}$ である。

$\mathcal{W}$ の部分集合として、空白マスの集合を $\mathcal{W}_{\mathsf{blank}}$ とおき、制約白マスの集合を $\mathcal{W}_{\mathsf{hint}}$ とおく。Str8ts の答えにおいて白マス $\alpha \in \mathcal{W}$ に置かれる数字 (制約白マスの場合は最初から置かれている数字) を $x_\alpha \in \{1, 2, \ldots, n\}$ とおくと、Str8ts の答えは $\{x_\alpha\}_{\alpha \in \mathcal{W}}$ によって与えられる。

制約黒マス全体の集合を $\mathcal{H}$ とおき、制約黒マス $\delta \in \mathcal{H}$ に最初から置かれている数字を $x_\delta \in \{1, 2, \ldots, n\}$ とおく。

4.2 入力

P はすべての空白マス内に n 個のコミットメントを並べたカード列を次のように裏向きで配置する。各空白マスの数字と同じ位置にあるコミットメントを $\boxed{\heartsuit}\boxed{\clubsuit}$ とし、それ以外を $\boxed{\clubsuit}\boxed{\heartsuit}$ とする。このカード列を「入力列」とする。



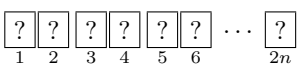
4.3 検証用カード列の作成

検証で使用する整数符号化のカード列を作成する。

(1) 各空白マス $\alpha \in \mathcal{W}_{\mathsf{blank}}$ に対して、以下の操作を行う。

(a) n 個のコミットメントをコピープロトコルでコピーすることによって、空白マス内の入力列を 2 個ずつに増やす。コピープロトコルの途中で不正な入力（$\boxed{\clubsuit}\boxed{\clubsuit}$ や $\boxed{\heartsuit}\boxed{\heartsuit}$ など符号化に従わない入力）が検出された場合、証明を棄却する。

(b) 2 個の入力列に対して以下の操作を実行する。入力列のカード枚数を左から順に数える。



以下のように奇数番目 $(1, 3, \ldots, 2n-1)$ のカード列と偶数番目 $(2, 4, \ldots, 2n)$ のカード列にわける。すると、前者は $E_n^{\heartsuit}(x_\alpha)$（$\boxed{\heartsuit}$ の位置 ℓ で x_a を表す整数符号化）となり、後者は $E_n^{\clubsuit}(x_\alpha)$（$\boxed{\clubsuit}$ の位置 ℓ で x_a を表す整数符号化）となる。

$$E_n^{\heartsuit}(x_\alpha) = \underset{1}{\boxed{?}}\,\underset{2}{\boxed{?}}\cdots\underset{\ell}{\boxed{\heartsuit}}\cdots\underset{n}{\boxed{?}}$$
$$E_n^{\clubsuit}(x_\alpha) = \underset{1}{\boxed{?}}\,\underset{2}{\boxed{?}}\cdots\underset{\ell}{\boxed{\clubsuit}}\cdots\underset{n}{\boxed{?}}$$

(2) 各制約白マス $\alpha \in \mathcal{W}_{\mathsf{hint}}$ に対して、$E_n^{\heartsuit}(x_\alpha)$ および $E_n^{\clubsuit}(x_\alpha)$ を 2 個ずつ作成する。

以上の手順により、各白マス $\alpha \in \mathcal{W}$ には $E_n^{\heartsuit}(x_\alpha)$ と $E_n^{\clubsuit}(x_\alpha)$ が 2 個ずつ置かれる。

4.4 制約黒マス条件の検証

制約黒マス条件に対する検証を行う。

各制約黒マス $\delta \in \mathcal{H}$ に対して以下の操作を行う。

(1) $\delta = (i, j)$ の行 i および列 j に属するすべての空白マス $\alpha \in \mathcal{W}_{\mathsf{blank}}$ から $E_n^{\heartsuit}(x_\alpha)$ を 1 個ずつ取り出す。

(2) $x_\delta = \ell$ のとき、取り出した $E_n^{\heartsuit}(x_\alpha)$ の ℓ 番目のカードだけを表向きにして $\boxed{\clubsuit}$ であることを確認する。

$$E_n^{\heartsuit}(x_\alpha):\ \underset{1}{\boxed{?}}\,\underset{2}{\boxed{?}}\cdots\underset{\ell}{\boxed{\clubsuit}}\cdots\underset{n}{\boxed{?}}$$

表向きにしたカードが $\boxed{\heartsuit}$ の場合、証明を棄却する。

(3) 表向きにしたカードを裏向きに直し、取り出した空白マスへ戻す。

4.5 ユニーク条件の検証

ユニーク条件に対する検証を行う。

(1) i 行目の白マス座標の集合を $\mathcal{C}^{(i)} = \{\alpha_1, \ldots, \alpha_{c_i}\}$（ただし $c_i := |\mathcal{C}^{(i)}|$）とおき、$j$ 列目の白マス座標の集合を $\mathcal{R}^{(j)} = \{\beta_1, \ldots, \beta_{r_j}\}$（ただし $r_j := |\mathcal{R}^{(j)}|$）とおいたとき、$c_i \times n$ のカード行列 $C^{(i)}$ および $r_j \times n$ のカード行列 $R^{(j)}$ を以下のように定義する。

$$C^{(i)} = \begin{pmatrix} E_n^{\heartsuit}(x_{\alpha_1}) \\ E_n^{\heartsuit}(x_{\alpha_2}) \\ \vdots \\ E_n^{\heartsuit}(x_{\alpha_{c_i}}) \end{pmatrix},\ R^{(j)} = \begin{pmatrix} E_n^{\heartsuit}(x_{\beta_1}) \\ E_n^{\heartsuit}(x_{\beta_2}) \\ \vdots \\ E_n^{\heartsuit}(x_{\beta_{r_j}}) \end{pmatrix}$$

カード行列 $C^{(i)}$ および $R^{(j)}$ の ℓ 列目（$1 \le \ell \le n$）を $C_\ell^{(i)}$ および $R_\ell^{(j)}$ とおく（$C_\ell^{(i)}$ は c_i 枚のカードが縦に並んだカード列であり、$R_\ell^{(j)}$ は r_j 枚のカードが縦に並んだカード列である）。

(2) 各 $C^{(i)}$（$1 \le i \le n$）に対して、以下の操作を行う。

(a) $C_1^{(i)}, \ldots, C_n^{(i)}$ をそれぞれカード束として、n 個のカード束に対して PSS を行う。

$$\left[\ \underset{C_1^{(i)}}{\begin{matrix}\boxed{?}\\ \vdots \\ \boxed{?}\end{matrix}}\ \middle|\ \underset{C_2^{(i)}}{\begin{matrix}\boxed{?}\\ \vdots \\ \boxed{?}\end{matrix}}\ \middle|\ \cdots\ \middle|\ \underset{C_n^{(i)}}{\begin{matrix}\boxed{?}\\ \vdots \\ \boxed{?}\end{matrix}}\ \right]$$

シャッフル適用後のカード行列を $\widetilde{C}^{(i)}$ とする。

(b) すべてのカードを表向きにする。

(c) $\widetilde{C}^{(i)}$ の各行に $\boxed{\heartsuit}$ が 1 枚だけであることを確認する。0 枚または 2 枚以上ある場合は、証明を棄却する。

(d) $\widetilde{C}^{(i)}$ の各列に $\boxed{\heartsuit}$ が 0 枚または 1 枚であることを確認する。2 枚以上ある場合は、証明を棄却する。

(3) 手順 2 を各 $R^{(j)}$（$1 \le j \le n$）に対しても行う。

4.6 ストレート条件の検証

ストレート条件に対する検証を行う。

(1) k ブロック目の白マス座標の集合を $\mathcal{B}^{(k)} = \{\alpha_1, \ldots, \alpha_{b_k}\}$（ただし $b_k := |\mathcal{B}^{(k)}|$）とおいたとき、$b_k \times n$ のカード行列 $B^{(k)}$ を以下のように定義する。

$$B^{(k)} = \begin{pmatrix} E_n^{\clubsuit}(x_{\alpha_1}) \\ E_n^{\clubsuit}(x_{\alpha_2}) \\ \vdots \\ E_n^{\clubsuit}(x_{\alpha_{b_k}}) \end{pmatrix}$$

$E_n^{\clubsuit}(x_{\alpha_\ell})$（$1 \le \ell \le b_k$）を $B_\ell^{(k)}$ で表すことにする。

(2) 各 $B^{(k)}$（$1 \le k \le m$）に対して、以下の操作を行う。

(a) $B_1^{(k)}, \ldots, B_{b_k}^{(k)}$ をそれぞれカード束として、b_k 個のカード束に対して PSS を行う。

$$\begin{matrix} B_1^{(k)} \\ B_2^{(k)} \\ \vdots \\ B_{b_k}^{(k)} \end{matrix}\ \left[\begin{array}{cccc} \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \hline \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \hline \vdots & \vdots & \ddots & \vdots \\ \hline \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{array}\right]$$

シャッフル適用後のカード行列を $\widetilde{B}^{(k)}$ とする。$\widetilde{B}^{(k)}$ の ℓ 列目（$1 \le \ell \le n$）を $\widetilde{B}_\ell^{(k)}$ とおく。

(b) $\widetilde{B}^{(k)}$ の各行に対して、$n+1$ 番目に $\boxed{\heartsuit}$ を 1 枚ずつ追加し、$\widetilde{B}_{n+1}^{(k)}$ とする。

(c) $\widetilde{B}_1^{(k)}, \ldots, \widetilde{B}_{n+1}^{(k)}$ をそれぞれカード束として、$n+1$ 個のカード束に対して PShift を行う。

$$\left\langle\ \underset{\widetilde{B}_1^{(k)}}{\begin{matrix}\boxed{?}\\ \vdots \\ \boxed{?}\end{matrix}}\ \middle|\ \underset{\widetilde{B}_2^{(k)}}{\begin{matrix}\boxed{?}\\ \vdots \\ \boxed{?}\end{matrix}}\ \middle|\ \cdots\ \middle|\ \underset{\widetilde{B}_{n+1}^{(k)}}{\begin{matrix}\boxed{\heartsuit}\\ \vdots \\ \boxed{\heartsuit}\end{matrix}}\ \right\rangle$$

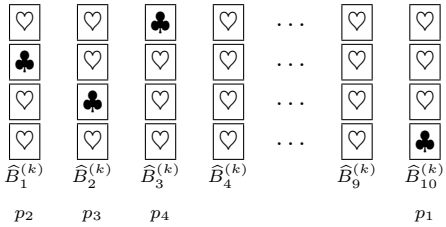
シャッフル適用後のカード行列を $\widehat{B}^{(k)}$ とする。$\widehat{B}^{(k)}$ の ℓ 列目（$1 \le \ell \le n+1$）を $\widehat{B}_\ell^{(k)}$ とおく。

(d) すべてのカードを表向きにする。

(e) $\widehat{B}^{(k)}$ の $n+1$ 個の列のうち $\boxed{\clubsuit}$ を含む列が b_k 個連続して隣り合っていることを確認する。ただし、$n+1$ 列目と 1 列目は隣り合った列であるとみなす。連続していない場合は、証明を棄却する。

例えば、$n = 9, b_k = 4$ の場合に、シャッフル操作

によって以下の配置が得られたとする。



このとき、♣ が存在する $\widehat{B}^{(k)}$ の列を p_1, p_2, p_3, p_4 とすると、列は 4 個連続して隣り合っているので正しいことが確認できる。

5. 提案プロトコル

本節では、基本プロトコルに変更を加えて、PSS のみで実行する提案プロトコルの手順について述べる。

5.1 ストレート条件検証の変更

4.6 節手順 2-b に続いて以下のような手順に変更する。

(c′) $n+1$ 種類の数字カードを 2 枚ずつ使用し、3 節手順 1 に該当する配置を作成し、カード行列 $U^{(k)}$ を定義する（ここでは、0 から n までのカードを使うことにする）。

$$U^{(k)} = \begin{pmatrix} [0] & [1] & \cdots & [n] \\ [0] & [1] & \cdots & [n] \end{pmatrix}$$

$U^{(k)}$ の ℓ 列目（$1 \le \ell \le n+1$）を $U_\ell^{(k)}$ とおく。

(d′) $U_1^{(k)}, \ldots, U_{n+1}^{(k)}$ をそれぞれカード束として、$n+1$ 個のカード束に対して PSS を行う。

$$\left[\begin{array}{c|c|c|c} [0] & [1] & \cdots & [n] \\ [0] & [1] & \cdots & [n] \end{array}\right]$$
$$U_1^{(k)} \quad U_2^{(k)} \quad \quad U_{n+1}^{(k)}$$

シャッフル適用後のカード行列を $\widetilde{U}^{(k)}$ とする。この操作は 3 節手順 2 に該当する。

(e′) 各 $\widetilde{U}^{(k)}$ は、以下のように $(u_1, u_2, \ldots, u_n)$ の列が得られたものとする。

$$\widetilde{U}^{(k)} = \begin{pmatrix} [u_0] & [u_1] & \cdots & [u_n] \\ [u_0] & [u_1] & \cdots & [u_n] \end{pmatrix}$$

$\widetilde{U}^{(k)}$ の下段に対して、置換 $(u_n\ u_{n-1}\ \cdots\ u_0)$ を行う。$\widetilde{B}^{(k)}$ と以下のように結合し、カード行列 $V^{(k)}$ とする。

$$V^{(k)} = \begin{pmatrix} [u_0] & [u_1] & \cdots & [u_n] \\ [u_1] & [u_2] & \cdots & [u_0] \\ \widetilde{B}_1^{(k)} & \widetilde{B}_2^{(k)} & \cdots & \widetilde{B}_{n+1}^{(k)} \end{pmatrix}$$

$V^{(k)}$ の ℓ 列目（$1 \le \ell \le n+1$）を $V_\ell^{(k)}$ とおく（$V_\ell^{(k)}$ は $b_k + 2$ 枚のカードが縦に並んだカード列である）。この操作は 3 節手順 3 および 4 に該当する。

(f′) $V_1^{(k)}, \ldots, V_{n+1}^{(k)}$ をそれぞれカード束として、$n+1$ 個のカード束に対して PSS を行う。

$$\left[\begin{array}{c|c|c|c} [u_0] & [u_1] & \cdots & [u_n] \\ [u_1] & [u_2] & \cdots & [u_0] \\ ? & ? & \cdots & ? \\ \vdots & \vdots & \ddots & \vdots \\ ? & ? & \cdots & ? \end{array}\right]$$
$$V_1^{(k)} \quad V_2^{(k)} \quad \quad V_{n+1}^{(k)}$$

シャッフル適用後のカード行列を $\widetilde{V}^{(k)}$ とする。$\widetilde{V}^{(k)}$ の ℓ 列目（$1 \le \ell \le n+1$）を $\widetilde{V}_\ell^{(k)}$ とおく。この操作は 3 節手順 5 に該当する。

(g′) すべてのカードを表向きにし、$\widetilde{V}^{(k)}$ の 1 段目と 2 段目を確認し、3 節手順 6-b にならって $\widetilde{V}_\ell^{(k)}$ のカード束ごと並び替える。

(h′) $\widetilde{V}^{(k)}$ の $n+1$ 個の列のうち ♣ を含む列が b_k 個連続して隣り合っていることを確認する。ただし、$n+1$ 列目と 1 列目は隣り合った列であるとみなす。連続していない場合は、証明を棄却する。

5.2 バッチングについて

提案プロトコルは 3 回の PSS で実行できることを示す。カード枚数が異なるカード束に対してまとめて PSS を行う場合、ダミーカードを追加することで、バッチングを適用できるようにする。ダミーカードはカード枚数を揃えるためのものであり、カードの種類は問わない。シャッフル操作後はダミーカードを回収する。

(a) 4.3 節手順 1-a のコピープロトコルにおけるシャッフルは、(i, j, ℓ) の組（ℓ は対象のコミットメントの数え番号）を識別情報とするバッチングにより、$n|\mathcal{W}_{\mathsf{blank}}|$ 個のコミットメントに対して 1 回の PSS でまとめて実行できる。

(b) 以下の PSS はバッチングによりまとめて適用することができる。

- $C^{(i)}$（$1 \le i \le n$）に関する 4.5 節手順 2-a での $C_1^{(i)}, \ldots, C_n^{(i)}$（合計 n^2 束）に対する PSS
- $R^{(j)}$（$1 \le j \le n$）に関する 4.5 節手順 3 での $R_1^{(j)}, \ldots, R_n^{(j)}$（合計 n^2 束）に対する PSS
- $B^{(k)}$（$1 \le k \le m$）に関する 4.6 節手順 2-c での $B_1^{(k)}, \ldots, B_{b_k}^{(k)}$（合計 $2|\mathcal{W}|$ 束）に対する PSS
- $U^{(k)}$（$1 \le k \le m$）に関する 5.1 節手順 d′ での $U_1^{(k)}, \ldots, U_{n+1}^{(k)}$（合計 $m(n+1)$ 束）に対する PSS

$C^{(i)}, R^{(j)}, k$ ごとにバッチングの識別情報を区別する。なお、$C_\ell^{(i)}, R_\ell^{(j)}, U_\ell^{(k)}$ はそれぞれ $n+1-w_i,\ n+1-w_j, n-1$ 枚のカードが不足しているので、シャッフル操作の直前にダミーカードを追加して $n+2$ 枚（1 枚の識別情報を含む）のカード束にする。

(c) 以下の PSS はバッチングによりまとめて適用することができる。

- $V^{(k)}$（$1 \leq k \leq m$）に関する 5.1 節手順 f′ での $V_1^{(k)}, \ldots, V_{n+1}^{(k)}$（合計 $m(n+1)$ 束）に対する PSS
  k ごとにバッチングの識別情報を区別する。なお、$V_\ell^{(k)}$ は $n+2-b_k$ 枚のカードが不足しているので、シャッフル操作の直前にダミーカードを追加して $n+3$ 枚（1 枚の識別情報を含む）のカード束にする。

以上の 3 つのバッチングにより、提案プロトコル全体は 3 回の PSS でプロトコルを実行できる。

6. プロトコルの正当性

本節では、プロトコルがゼロ知識証明の条件を満たしていることを述べる。

6.1 完全性

各検証フェーズによって、正しい解を必ず検出できることを示す。

- コミットメントの検証
  正当なコミットメントを置いている場合、4.3 節手順 1 のコピープロトコルでコピーされる。
- 入力列の検証
  正しい入力列の場合、各行（列）の入力列には、$\boxed{\heartsuit}\boxed{\clubsuit}$ がただ 1 つだけ存在する。つまり、空白マスの値が定まるので、4.5 節手順 2-c の検証が通る。
- 制約黒マス条件の検証
  正しい解の場合、すべての制約黒マスの行・列に属する空白マスの数字が制約黒マスの数字と一致しないので、4.4 節手順 2 の検証が通る。
- ユニーク条件の検証
  正しい解の場合、すべての行（列）に属する空白マスに埋めた数字はその行（列）に関して重複していないので、4.5 節手順 2-d の検証が通る。
- ストレート条件の検証
  正しい解の場合、各ブロックの空白マスの数字の並びは連番であるので、4.6 節手順 2-e および 5.1 節手順 h′ の検証が通る。

6.2 健全性

P が空白マスに対して行う不正な解は、以下の 5 つの要素が考えられる。

(1) 無効なコミットメントを配置する。
(2) 数字を指定しない、または複数の数字を指定する。
(3) ユニーク条件を満たさない数字を指定する。
(4) 制約黒マス条件を満たさない数字を指定する。
(5) ストレート条件を満たさない数字を指定する。

これらの要素は、プロトコルの手順で棄却されることを示す。

- (1) のとき、$\boxed{\heartsuit}\boxed{\heartsuit}$ などが入力列に存在する。これは、4.2 節手順 1 で検出できる。
- (2) のとき、ある入力列は $\boxed{\heartsuit}\boxed{\clubsuit}$ が 1 つではない。これは、4.5 節手順 2-c で検出できる。
- (3) のとき、ある空白マス数字は、その空白マスの行列上に存在する制約黒マスの数字と一致する。これは、4.4 節手順 2 で検出できる。
- (4) のとき、ある行（列）上の空白マスの数字が、その行（列）に属する別の白マスの数字と一致する。これは、4.5 節手順 2-d で検出できる。
- (5) のとき、あるブロック上の空白マスの数字の並びが連続しない。これは、4.6 節手順 2-e および 5.1 手順 h′ で検出できる。

6.3 ゼロ知識性

カードの公開時に、解の情報が漏れないことを示す。

- ユニーク条件
  4.5 節手順 2-a の PSS により、$C^{(i)}$ の各列 $C_\ell^{(i)}$ の分布は一様かつ独立に定まる。$\widetilde{C}^{(i)}$ の各行を見ると、行に対応する白マスは特定できる。各列を見ると、空白マスの数字がユニークであることがわかる。しかし、いずれの場合もそれ以外の情報は漏れない。列に関しても同様に、条件を満たすこと以外の情報は漏れない。
- 制約黒マス条件
  4.4 節により、$E_n^{\heartsuit}(x_\alpha)$ の x_δ に対応するカードだけを表向きにするので、数字が一致していないこと以外の情報は漏れない。
- ストレート条件
  4.6 節手順 2-a の PSS により、$B^{(k)}$ の各行 $B_\ell^{(k)}$ の分布は一様かつ独立に定まる。よって、$\widetilde{B}^{(k)}$ の各行を見ても、行に対応する白マスは特定できなくなる。
  基本プロトコルでは、その後の 4.6 節手順 2-c の PShift により、$\widetilde{B}^{(k)}$ の各列の分布は一様かつ独立に定まる。PShift は $\widetilde{B}^{(k)}$ ごとに独立して行うので、盤面上の同じ行（列）に含まれるブロック同士の関係性などは漏れない。$\widehat{B}^{(k)}$ の各列から、ストレートであることはわかるが、それ以外の情報は漏れない。
  提案プロトコルでは、5.1 節手順 f′ の PSS により、$\widetilde{B}^{(k)}$ または $V^{(k)}$ の各列の分布は一様かつ独立に定まる。$\widetilde{V}^{(k)}$ の各列から、ストレートであることはわかるが、それ以外の情報は漏れない。

7. カード枚数の効率性評価

本節では、カード枚数に着目してプロトコルを評価する。

7.1 各プロトコルの評価

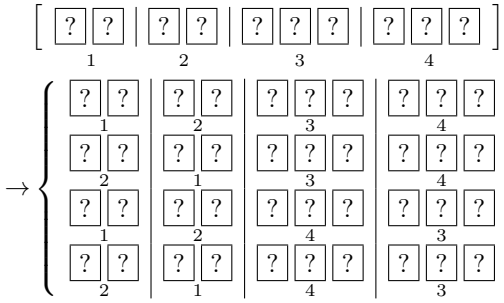
- 基本プロトコル
  使用するカードは以下に示す検証用カード列の準備過程でのみ発生する。

– 4.2 節の $2n|\mathcal{W}_{\mathsf{blank}}|$ 枚
– 4.3 節手順 1-a の $4n|\mathcal{W}_{\mathsf{blank}}|$ 枚
– 4.3 節手順 2 の $4n|\mathcal{W}_{\mathsf{hint}}|$ 枚
– 4.6 節手順 2-b の $|\mathcal{W}|$ 枚

- 提案プロトコル
  基本プロトコルのカード枚数（$(6n+1)|\mathcal{W}_{\mathsf{blank}}|+(4n+1)|\mathcal{W}_{\mathsf{hint}}|$ 枚）に加えて、グラフシャッフルによる追加カードを要する。
  – 5.1 節手順 c′ の $2(n+1)|\mathcal{W}|$ 枚
  また、バッチングを利用する場合、識別用のカードを要する。バッチングでの識別情報は 1 枚のカードで表現することにする。
  – 5.2 節 a の $2n|\mathcal{W}_{\mathsf{blank}}|$ 枚
  – 5.2 節 b の $2n^2+2|\mathcal{W}|+m(n+1)$ 枚
  なお、5.2 節 c については、5.2 節 b で使った $m(n+1)$ 枚を流用することで、追加カードは必要ない。
  さらに、バッチングにはダミーカードも要する。
  – 5.2 節 b の $2n(n^2-|\mathcal{W}|)+m(n+1)(n-1)$ 枚
  – 5.2 節 c の $(n+1)(nm-2|\mathcal{W}|)$ 枚

7.2 一般化 PSS を用いた効率化

石崎–品川 [14] は異なる枚数からなるカード列に対して、同時に PSS を行う操作を提案した。2 枚からなる 2 個のカード束と 3 枚からなる 2 個のカード束に対する一般化 PSS は以下の通りである。



上記のように、4 通りのカード束の並びが考えられる。

したがって、一般化 PSS を許せば、提案プロトコルのダミーカード（ $2n(n^2-|\mathcal{W}|)+m(n+1)(n-1)+(n+1)(nm-2|\mathcal{W}|)$ 枚 ）をすべて省くことができる。

8. おわりに

本稿では、Str8ts に対するカードベースゼロ知識証明プロトコルを提案した。黒マスがない行・列ではユニーク条件を満たせばストレート条件も満たすため、重ねて検証する必要はない。また、ブロック内の空白マスが 1 つしかない場合には、ストレート条件を検証する必要はない。

今後の課題としては、プロトコルのシャッフル操作を 2 回以内に抑えることができるかについて挙げられる。また、別のパズルに対する巡回バッチングの応用方法についても考えられる。

謝辞 本研究は JSPS 科研費 JP23H00479、JP21K17702 と JST CREST JPMJCR22M1 の支援を受けた。

参考文献

[1] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pp. 291–304, 1985.
[2] R. Gradwohl, M. Naor, B. Pinkas, and G. N. Rothblum. Cryptographic and physical zero-knowledge proof systems for solutions of Sudoku puzzles. In P. Crescenzi, G. Prencipe, and G. Pucci eds., *Fun with Algorithms*, Vol. 4475 of *LNCS*, pp. 166–182, Berlin, Heidelberg, 2007. Springer.
[3] R. Ishikawa, E. Chida, and T. Mizuki. Efficient card-based protocols for generating a hidden random permutation without fixed points. In C. S. Calude and M. J. Dinneen eds., *Unconventional Computation and Natural Computation*, Vol. 9252 of *LNCS*, pp. 215–226, Cham, 2015. Springer.
[4] W. Jeff and S. Andrew. Str8ts.com, 2008.
[5] W. Jeff and S. Andrew. Str8ts 9x9 Samples Document. 8, 2010.
[6] K. Miyamoto and K. Shinagawa. Graph automorphism shuffles from pile-scramble shuffles. *New Gener. Comput.*, 40:199–223, 2022.
[7] T. Mizuki and H. Sone. Six-card secure AND and four-card secure XOR. In X. Deng, J. E. Hopcroft, and J. Xue eds., *Frontiers in Algorithmics*, Vol. 5598 of *LNCS*, pp. 358–369, Berlin, Heidelberg, 2009. Springer.
[8] T. Ono, S. Ruangwises, Y. Abe, K. Hatsugai, and M. Iwamoto. Single-shuffle physical zero-knowledge proof for sudoku using interactive inputs. In K. Emura and H. Morita eds., *ACM ASIA Public-Key Cryptography Workshop*, New York, 2025. ACM.
[9] S. Ruangwises. Two standard decks of playing cards are sufficient for a ZKP for Sudoku. In C.-Y. Chen, W.-K. Hon, L.-J. Hung, and C.-W. Lee eds., *Computing and Combinatorics*, Vol. 13025 of *LNCS*, pp. 631–642, Cham, 2021. Springer.
[10] T. Sasaki, T. Mizuki, and H. Sone. Card-based zero-knowledge proof for Sudoku. In H. Ito, S. Leonardi, L. Pagli, and G. Prencipe eds., *Fun with Algorithms*, Vol. 100 of *LIPIcs*, pp. 29:1–29:10, Dagstuhl, Germany, 2018. Schloss Dagstuhl.
[11] K. Shinagawa, T. Mizuki, J. Schuldt, K. Nuida, N. Kanayama, T. Nishide, G. Hanaoka, and E. Okamoto. Card-based protocols using regular polygon cards. *IEICE Trans. Fundam.*, E100.A(9):1900–1909, 2017.
[12] K. Tanaka and T. Mizuki. Two uno decks efficiently perform zero-knowledge proof for Sudoku. In H. Fernau and K. Jansen eds., *Fundamentals of Computation Theory*, Vol. 14292 of *LNCS*, pp. 406–420, Cham, 2023. Springer.
[13] K. Tanaka, S. Sasaki, K. Shinagawa, and T. Mizuki. Only two shuffles perform card-based zero-knowledge proof for sudoku of any size. In *2025 Symposium on Simplicity in Algorithms (SOSA)*, pp. 94–107. SIAM, 2025.
[14] 石崎悠斗, 品川和雅. 追加カード 2 枚の多入力 AND 計算におけるシャッフル回数の新しい削減方法. 2024 年暗号と情報セキュリティシンポジウム (SCIS2024) 3D1-1, pp. 1–8, 2024.