テントアンドツリーに対するカードベースゼロ知識証明

藤原 愛斗[1,a)]　池田 昇太[1,b)]　品川 和雅[2,3,c)]

概要：テントアンドツリーとは、木が配置された盤面上で、各木に対応するテントの位置を論理的に推定するパズルである。本稿では、テントアンドツリーに対するカードベースゼロ知識証明プロトコルを 2 つ提案する。1 つ目は直感的な理解がしやすいシンプルなプロトコルであり、そのシャッフル回数は 5 回である。2 つ目は 1 つ目の手順をベースに、より多くのカードを使用する代わりにシャッフル回数を 3 回に減らしたプロトコルである。

キーワード：カードベース暗号、ゼロ知識証明、テントアンドツリー

Card-Based Zero-Knowledge Proofs for Tents and Trees

Aito Fujiwara[1,a)]　Shota Ikeda[1,b)]　Kazumasa Shinagawa[2,3,c)]

Abstract: Tents and Trees is a logic puzzle played on a grid with pre-placed trees, where the goal is to deduce the correct positions of tents corresponding to each tree. In this paper, we propose two card-based zero-knowledge proof protocols for the Tents and Trees puzzle. The first protocol is simple and easy to understand intuitively, requiring five shuffles. The second protocol builds upon the first, reducing the number of shuffles to three by using a greater number of cards.

Keywords: Card-based cryptography, Zero-knowledge proof, Tents and Trees

1. はじめに

1.1 テントアンドツリー

テントアンドツリー（テントサイト、テント）[7] とは、オランダの Leon Balmaekers によって考案された論理パズルである。木の配置された盤面が与えられ、プレイヤーは以下の条件[*1]を全て満たすようなテントの配置を考える。

対応条件　全ての木に対して 1 つのテントが対応している。

隣接条件　テントは対応している木の上下左右 1 マスのいずれかに位置する。

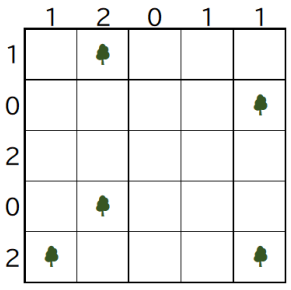


図 1　例題

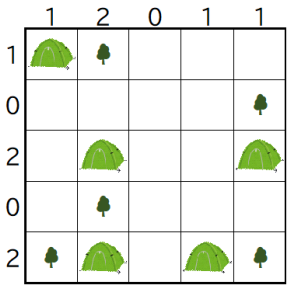


図 2　例題の解

周囲条件　どの 2 つのテントも縦・横・斜めで隣り合うことはない。

行列条件　各行および各列に位置するテントの数は盤外の数字に等しい。

テントアンドツリーの例題とその解を図 1, 2 に示す。図 2 において、上記の 4 つの条件が満たされていることが確認できる。

[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)] 22t4066l@vc.ibaraki.ac.jp
[b)] 25nm709x@vc.ibaraki.ac.jp
[c)] shinagawa@cs.tsukuba.ac.jp

[*1] 対応条件・隣接条件・周囲条件・行列条件という名称は本稿で便宜的に使用するものであることに注意する。

表 1 提案プロトコル 1 と提案プロトコル 2 の比較

	カード枚数	シャッフル回数
提案プロトコル 1	$(2n_{\mathsf{max}}+4)\|E\|-n_{\mathsf{sum}}$	5
提案プロトコル 2	$(3n_{\mathsf{max}}+5)\|E\|-n_{\mathsf{sum}}$	3

E：候補マスと木マスの両方に隣接する辺の集合（2.3 節）
n_{max} と n_{sum}：周辺の辺の個数の最大値と総和（2.3 節）

1.2 ゼロ知識証明

ゼロ知識証明とは、Goldwasser, Micali, Rackoff [3] によって提案された暗号技術であり、証明したい事柄以外の情報を何も明かすことのない証明を指す。ゼロ知識証明は以下の 3 つの性質からなる。

完全性 証明者 P が知識 w を知っている場合、検証者 V は常に証明を受理する。

健全性 証明者 P が知識 w を知らない場合、検証者 V は常に証明を棄却する。

ゼロ知識性 検証者 V は証明の過程で知識 w に関する情報を得ることができない。

パズルに対するゼロ知識証明とは、「パズルに解が存在する」という主張に対するゼロ知識証明である。本稿では、テントアンドツリー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。すなわち、具体的なテントの位置を明かさずに、対応条件・隣接条件・周囲条件・行列条件の全てを満たすテントの配置が存在することを証明する。

1.3 貢献

本稿の貢献は、3,4 節にて提案するカードベースゼロ知識証明プロトコル (提案プロトコル 1, 提案プロトコル 2) により、テントアンドツリーに対する物理的なゼロ知識証明を構成したことである。

提案プロトコル 1 は各条件を 1 つずつ検証する基本的なプロトコルであるのに対して、提案プロトコル 2 は提案プロトコル 1 よりも使用カード枚数を増やす代わりにシャッフル回数を減らしたプロトコルとなっている。提案プロトコル 1 と提案プロトコル 2 の比較表を表 1 に示す。使用カード枚数は提案プロトコル 1 が $(2n_{\mathsf{max}}+4)|E|-n_{\mathsf{sum}}$ 枚、提案プロトコル 2 が $(3n_{\mathsf{max}}+5)|E|-n_{\mathsf{sum}}$ 枚となり、提案プロトコル 1 の方が $(n_{\mathsf{max}}+1)|E|$ 枚少ない。なお、E は候補マスと木マスの両方に隣接する辺の集合、n_{max} は周辺の辺の個数の最大値、n_{sum} は周辺の辺の個数の総和を指す（2.3 節）。シャッフル回数は提案プロトコル 1 が 5 回、提案プロトコル 2 が 3 回となり、提案プロトコル 2 の方が 2 回少ない。

1.4 関連研究

カードベース暗号 [1, 2] とは、物理的なカードを用いてゼロ知識証明や秘密計算といった技術を実現する分野である。カードベース暗号のプロトコルが評価される際、指標としてシャッフル回数が用いられることがある。同じ機能を持つプロトコルがある場合、より少ない、かつ定数回のシャッフルで動作できる方が良いとされている。シャッフル定数回のカードベースゼロ知識証明プロトコルの既存研究として、数独に対するシャッフル 2 回のプロトコル [6]、グラフ同型問題に対するシャッフル 3 回のプロトコル [5]、ハミルトン閉路問題に対するシャッフル 3 回のプロトコル [8] などが知られている。本研究と同じく、これらのプロトコルで用いられているシャッフルは、すべてパイルスクランブルシャッフルである。

2. 準備

2.1 使用カード

3,4 節では以下のカードを使用する。

2 色カード 2 種類から構成され、それぞれの表面が ♡ ♣ であるカード。

数字カード 1 2 3 ⋯ のように表面に数字の書かれたカード。

ダミーカード □ のように表面にシンボルを持たないカード。提案プロトコル 2 で使用される。

全てのカードの裏面は共通して ? である。そのため、カードの裏面から表面を推測することはできない。

2.2 パイルスクランブルシャッフル

3,4 節では、Ishikawa, Chida, Mizuki [4] によって提案されたパイルスクランブルシャッフルを使用する。これは、同じ枚数からなる複数のカード束の列に対して、完全ランダムに並べ替えるシャッフル操作である。例えば、同じ枚数からなる 3 個のカード束 A, B, C がこの順に並んでいるとする。これらにパイルスクランブルシャッフルを行うと、A, B, C の位置がランダムに並べ変えられる。シャッフル後の並び順は $(A,B,C),(A,C,B),(B,A,C),(B,C,A),(C,A,B),(C,B,A)$ の内のいずれかになり、各結果の生起確率は等しい。また、パイルスクランブルシャッフルの前後で各カード列の中身が並べ替えられることはない。例えばカード束 A の中身が 1 2 3 4 であり、この順で構成されているとする。パイルスクランブルシャッフルによりカード束 A の位置が変わったとしても、カード束 A 内の並び順は変わらず 1 2 3 4 の順である。

パイルスクランブルシャッフルを適用するとき、本稿では $[\,\cdot\,|\,\cdot\,|\cdots|\,\cdot\,]$ で表す。例えば、4 枚からなる 3 個のカード束の列に対するパイルスクランブルシャッフルは、以下のように表される。

$$\left[\ ?\,?\,?\,?\ \middle|\ ?\,?\,?\,?\ \middle|\ ?\,?\,?\,?\ \right]$$

2.3 テントアンドツリーの用語の定義

マスの種類 各マスを「木のあるマス (木マス)」、「テントを置く候補となるマス (候補マス)」、「テントを置けな

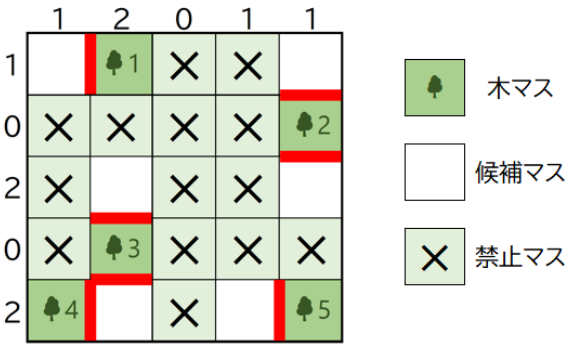


図 3　テントアンドツリーの用語の定義

いマス (禁止マス)」の 3 種類に分類する。分類法は以下の通りである。なお、一度分類されたマスは以降の手順で再分類されない。

(1) そのマスに木が描かれているとき、そのマスを木マスとする。
(2) そのマスが含まれる行または列の指定されたテント数が 0 である場合、そのマスを禁止マスとする。
(3) そのマスの上下左右に木マスがあるとき、そのマスを候補マスとする。
(4) そのマスがまだ分類されていない場合、そのマスを禁止マスとする。

木の番号　全ての木に対して、番号を 1 から順に割り振る。

木マスと候補マスの境界　木マスと候補マスの境界となる辺を、それぞれ (i,j,k) の 3 つの数字で表して 1 つの集合 E とする。なお、i は対応する候補マスの行、j は対応する候補マスの列、k は対応する木の番号を指す。図 1 の場合だと、集合 E は $(1,1,1),(1,5,2),(3,2,3),(3,5,2),(5,2,3),(5,2,4),(5,4,5)$ という 7 組から構成される。図 3 において、E に該当する辺は太い赤色で表現されている。

周辺の辺　$(i,j,k) \in E$ に対して、$i-1 \leq i' \leq i+1$ と $j-1 \leq j' \leq j+1$ と $k' \neq k$ を満たす $(i',j',k') \in E$ の全体集合を $N(i,j,k)$ で表す。すなわち $N(i,j,k)$ は、i 行 j 列のマスを中心とした 3×3 マスに含まれる E の要素の中で、木 k とは異なる木 k' の境界を集めた集合である。例えば、図 1 において $N(5,2,3) = (5,2,4)$ となる。また、$N(i,j,k)$ の要素数を $|N(i,j,k)|$ で表す。図 1 において $|N(5,2,3)| = 1$ である。また、n_{max} を以下のように定義する。

$$n_{\mathsf{max}} \coloneqq \max_{(i,j,k)\in E} |N(i,j,k)|$$

図 1 の場合は $n_{\mathsf{max}} = 1$ であり、$|N(i,j,k)| = 1$ を実現する (i,j,k) は $(5,2,3)$ と $(5,2,4)$ である。また、n_{sum} を以下のように定義する。

$$n_{\mathsf{sum}} \coloneqq \sum_{(i,j,k)\in E} |N(i,j,k)|$$

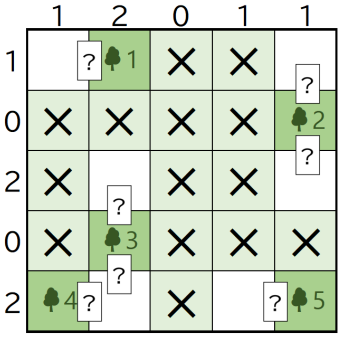


図 4　♡ と ♣ の入力

図 1 の場合は、$|N(5,2,3)| = |N(5,2,4)| = 1$ であり、それ以外の (i,j,k) では $|N(i,j,k)| = 0$ であるため、$n_{\mathsf{sum}} = 2$ となる。

3. 提案プロトコル 1

提案プロトコル 1 は「隣接条件を満たす盤面上へのカード配置」、「対応条件の検証」、「行列条件の検証」、「周囲条件の検証」から構成され、この順で実行される。

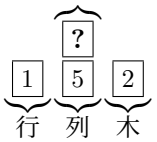
3.1 隣接条件を満たす盤面上へのカード配置

(1) 証明者はパズルの答えとなるテントの置かれる各マスに対して、そのテントに対応する木との境界に ♡ を裏向きにして置く。同時に、他の木マスと候補マスの境界に ♣ を裏向きにして置く。以降、この裏向きの各カードをコミットメントと呼ぶ。
(2) 検証者は木マスと候補マスの全ての境界だけにカードが 1 枚置かれていることを確認し、条件を満たさない場合は証明を棄却する。

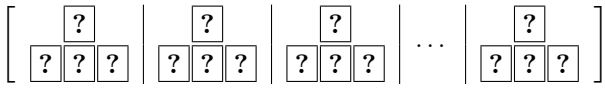
3.2 対応条件の検証

(1) 裏向きに置いた各カードの下に「対応する候補マスの位置する行」、「対応する候補マスの位置する列」、「対応する木の番号」を表す数字カードを並べる。図 1 において $(1,5,2)$ を例にすると、以下のようになる。

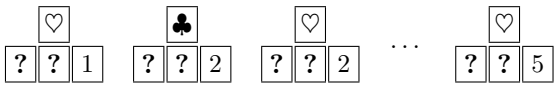
$(1,5,2)$ に対応するコミットメント



以降、この 4 枚で 1 組として扱う。
(2) 表のカードを全て裏向きにして、以下のようにパイルスクランブルシャッフルを行う。



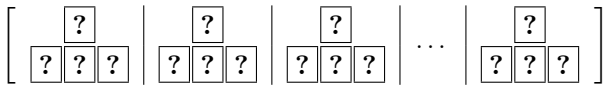
(3) 全組の 1 段目のカードと木を表す数字カードを表にし、木の数字カードが昇順になるように並び替える。



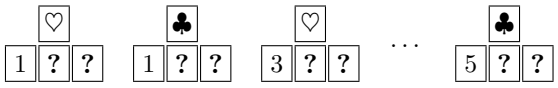
(4) 検証者は各木の番号ごとに ♡ が 1 枚だけ存在し、残りが全て ♣ であることを確認する。条件を満たさない場合、検証者は証明を棄却する。

3.3 行列条件の検証

(1) 先ほどの検証の終了時点で 1 段目のカードと木を表す数字カードが表になっているため、それらを全て裏向きにしてパイルスクランブルシャッフルを行う。

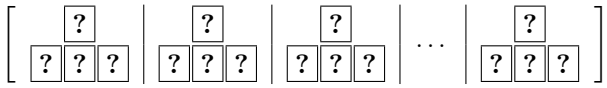


(2) 全組の 1 段目のカードと行を表す数字カードを表にし、行の数字カードが昇順になるように並び替える。

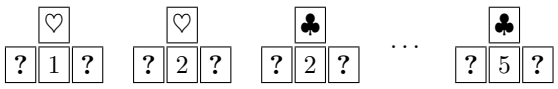


(3) 検証者は各行に存在する ♡ の枚数が指定されたテントの数だけであることを確認し、条件を満たさない場合は証明を棄却する。

(4) 表のカードを全て裏向きにしてパイルスクランブルシャッフルを行う。



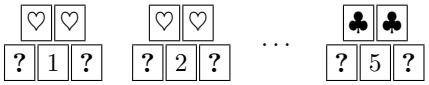
(5) 全組の 1 段目のカードと列を表す数字カードを表にし、列の数字カードが昇順になるように並び替える。



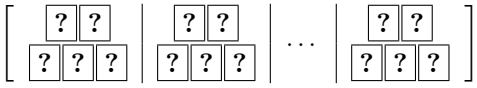
(6) 検証者は各列に存在する ♡ の枚数が指定されたテントの数だけであることを確認し、条件を満たさない場合は証明を棄却する。

3.4 周囲条件の検証

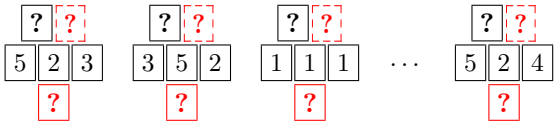
(1) 3.3 節の終了時点で 1 段目のカードと列の数字カードが表になっている。そこで、全組の 1 段目のカードに対して、同じマークのカードを表向きに $n_{\max}$ 枚だけ右に並べる。



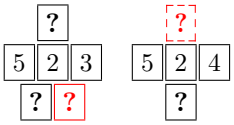
(2) 表になっているカードを全て裏向きにしてパイルスクランブルシャッフルを行う。



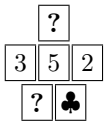
(3) 全組の行・列・木の数字カードを表にし、1 段目のカードから 1 枚取って 3 段目に置く。



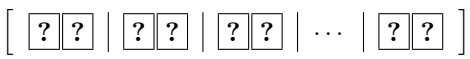
(4) 各候補マス $c = (i, j, k)$ に対して以下を実行する：$N(i, j, k)$ の各要素に対応している各組から 1 段目のカードを 1 枚ずつ取り、それを c の 3 段目の右に付け加える。以下は $c = (5, 2, 3)$ および $N(5, 2, 3) = \{(5, 2, 4)\}$ のとき（図 1）の場合の例である。$(5, 2, 4)$ の 1 段目のカードを、$(5, 2, 3)$ の 3 段目のカードの右に移動させている。



この操作を全ての候補マスについて実行した後、3 段目のカードが $n_{\max} + 1$ 枚未満の組に対しては、ちょうど $n_{\max} + 1$ 枚になるように ♣ を右に付け加える。以下は $c = (3, 5, 2)$ および $N(3, 5, 2) = \emptyset$ のとき（図 1）の場合の例である。



(5) 全組の 1, 2 段目を取り除き、表のカードを全て裏向きにしてパイルスクランブルシャッフルを行う。



(6) 各組の一番左のカードを表にする。

(7) 表になったカードが ♡ である全ての組について、その組の残りのカードを全て表にする。検証者はそれらが ♣ だけである場合に証明を受理し、♣ だけでない場合は証明を棄却する。

3.5 提案プロトコル 1 の効率性

カード枚数 3.1 節の (1) で $|E|$ 枚、3.2 節の (1) で $3|E|$ 枚、3.4 節の (1) で $n_{\max}|E|$ 枚、3.4 節の (4) で $n_{\max}|E| - n_{\mathsf{sum}}$ 枚のカードを追加している。よって、カード枚数は合計で $(2n_{\max} + 4)|E| - n_{\mathsf{sum}}$ 枚である。

シャッフル回数 3.2 節の (2)、3.3 節の (1) と (4)、3.4 節の (2) と (5) にて 1 回ずつパイルスクランブルシャッフルを行っている。よって、シャッフル回数は合計で 5 回である。

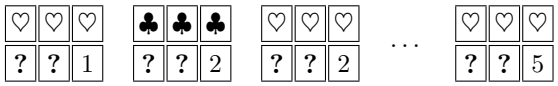
4. 提案プロトコル 2

4.1 実行手順

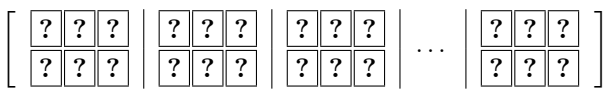
提案プロトコル 2 では、提案プロトコル 1 で行った行列条件の検証 (3.3 節) と周囲条件の検証 (3.4 節) を並列的に実行することにより、シャッフル回数を削減する。このプロトコルは、提案プロトコル 1 では使用していなかったダミーカード □ を使用する。

(1) 提案プロトコル 1 の対応条件の検証 (3.2 節) までの操作を行う。

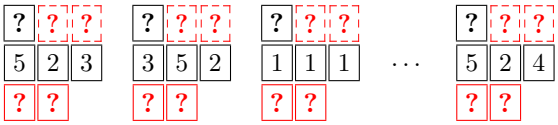
(2) (1) の終了時点で 1 段目のカードと木の数字カードが表になっている。そこで、全組の 1 段目のカードに対して、同じマークのカードを表向きに $n_{\max}+1$ 枚だけ右に並べる。



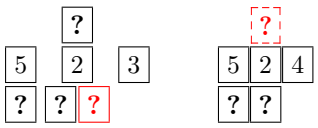
(3) 表のカードを裏向きにしてパイルスクランブルシャッフルを行う。



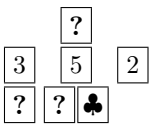
(4) 全組の行・列・木の数字カードを表にし、1 段目のカードから 2 枚取って、3 段目の行及び列の数字カードの下に 1 枚ずつ置く。



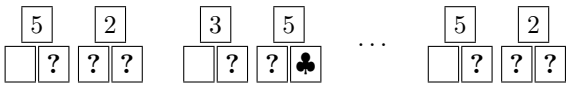
(5) 各候補マス $c=(i,j,k)$ に対して以下を実行する：$N(i,j,k)$ の各要素に対応している各組から 1 段目のカードを 1 枚ずつ取り、それを c の列の数字カードの下にある 3 段目のカードの右に付け加える。以下は $c=(5,2,3)$ および $N(5,2,3)=\{(5,2,4)\}$ のとき（図 1）の場合の例である。$(5,2,4)$ の 1 段目のカードを、$(5,2,3)$ の列の数字カードの下にある 3 段目のカードの右に移動させている。



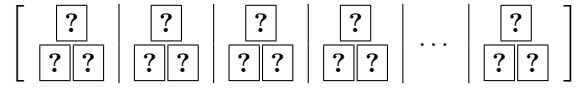
この操作を全ての候補マスについて実行した後、3 段目のカードが $n_{\max}+2$ 枚未満の組に対しては、ちょうど $n_{\max}+2$ 枚になるように ♣ を列の数字カードの下にある 3 段目のカードの右に付け加える。以下は $c=(3,5,2)$ および $N(3,5,2)=\emptyset$ のとき（図 1）の場合の例である。



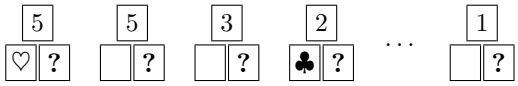
(6) 全組の 1 段目と木の数字カードを取り除き、行の数字カードの下に □ を表向きで $n_{\max}$ 枚だけ左に追加する。



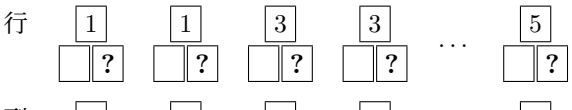
(7) 表のカードを裏向きにし、以下のように 1 組を 2 列に区切って、全体でパイルスクランブルシャッフルを行う。



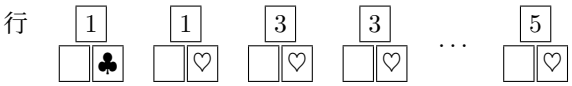
(8) 1 段目のカードおよび 2 段目の一番左のカードを表にする。



(9) 表になったカードに □ を含むグループと含まないグループの 2 種類に分けて、各グループで数字カードが昇順になるようにそれぞれ並び替える。なお、□ を含むグループは行の数字カード、□ を含まないグループは列の数字カードをそれぞれ持つ。

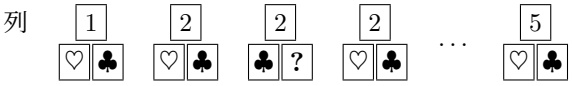


(10) □ を含むグループの各組の 2 段目に関して、一番右のカードを表にする。



(11) 検証者は各行・列に対応する表の ♡ の枚数が指定されたテントの数だけであることを確認し、条件を満たさない場合は証明を棄却する。

(12) □ を含まないグループの ♡ が表向きになっている全ての組について、その組の残りのカードを全て表にする。検証者はそれらが ♣ だけである場合に証明を受理し、♣ だけでない場合は証明を棄却する。



4.2 提案プロトコル 2 の効率性

カード枚数 (1) で $4|E|$ 枚、(2) で $(n_{\max}+1)|E|$ 枚、(5) で $n_{\max}|E|-n_{\sum}$ 枚、(6) で $n_{\max}|E|$ 枚のカードを追加している。よって、カード枚数は合計で $(3n_{\max}+5)|E|-n_{\sum}$ 枚である。

シャッフル回数 (1)、(3)、(7) にて 1 回ずつパイルスクランブルシャッフルを行っている。よって、シャッフル回数は合計で 3 回である。

5. 提案プロトコルの正当性

提案プロトコル 1 と提案プロトコル 2 がゼロ知識証明の 3 つの性質を満たしていることを証明する。

5.1 完全性

提案プロトコル 1 プロトコル通りに証明するとき、木マスと候補マスの境界にカードが 1 枚置かれることになる。そのため 3.1 節の (2) にて棄却されることはない。3.2 節の (4) では、各木の番号に対して ♡ が 1 枚だけ対応していれば棄却されない。証明者が正しい解に基づいたコミットメントの入力をした時、そのコミットメントは対応条件を満たしているので、この条件を満たしている。3.3 節の (3) では、指定されたテント数と同じ数だけ各行に ♡ があれば棄却されない。同様に 3.3 節の (6) では、指定されたテント数と同じ数だけ各列に ♡ があれば棄却されない。証明者が正しい解に基づいたコミットメントの入力をした時、そのコミットメントは行列条件を満たしているので、これらの条件を満たしている。3.4 節の (7) では、♡ の入力された辺に対応する E の要素 $(i,j,k) \in E$ に対して、$N(i,j,k)$ に属する要素に対応するコミットメントが全て ♣ になっていれば証明は受理される。証明者が正しい解に基づいたコミットメントの入力をした時、そのコミットメントは周囲条件を満たしているので、この条件を満たしている。よって、証明者が正しい解を知っている場合、検証者は常に証明を受理する。

提案プロトコル 2 4.1 節の (11) は 3.3 節の (3),(6) と、4.1 節の (12) は 3.4 節の (7) と同等の条件で証明が受理または棄却される。また、4.1 節の (1) で 3.1 節の (2) と 3.2 節の (4) も行われている。よって提案プロトコル 2 は提案プロトコル 1 と同等の条件で証明を受理及び棄却するため、提案プロトコル 1 の完全性が満たされていれば提案プロトコル 2 も完全性が満たされる。提案プロトコル 1 の完全性は既に証明済みなので、提案プロトコル 2 において証明者が正しい解を知っている場合、検証者は常に証明を受理する。

5.2 健全性

提案プロトコル 1 証明者が正しい解に基づいた入力をしていない場合、対応条件・隣接条件・行列条件・周囲条件のいずれかを満たしていない。なお、図 5 のようにテントが重なるような入力は周囲条件を満たしていない入力として扱う。

対応条件を満たしていない場合 3.2 節の (4) では、コミットメントと木の数字カードが共に表向きになっているため、各木の番号に対して何枚の ♡ があるかを知ることができる。そのため、ある木 T に対応しているテントが $t(\neq 1)$ 個であるとすると、木 T に対して t 枚の ♡ が対応していることが検証者に公開される。これにより、検証者は対応条件を満たしていない入力による証明を棄却する。

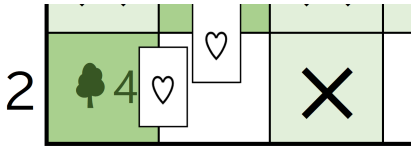


図 5 テントが重なるような入力

隣接条件を満たしていない場合 証明者が隣接条件を満たさない入力をしたとき、すなわち正しい解に基づいた場合と異なるカード配置の入力をしたとき、3.1 節の (2) にて検証者はその証明を棄却する。なぜならば、2.3 節で行ったマスの種類の分類は検証者に公開されている情報のみが使われているため、検証者はどの位置にカードが置かれるべきかを知っているからである。

行列条件を満たしていない場合 3.3 節の (3),(6) では、コミットメントと行及び列の数字カードが共に表向きになっているため、どの行・列に何枚の ♡ があるかを知ることができる。そのため、ある行 L に指定されたテント数が l 個、その行に位置するテントが $t_l(\neq l)$ 個であるとすると、行 L に対して t_l 枚の ♡ が対応していることが検証者に公開される。同様に、ある列 C に指定されたテント数が c 個、その列に位置するテントが $t_c(\neq c)$ 個であるとすると、列 C に対して t_c 枚の ♡ が対応していることが検証者に公開される。これにより、検証者は行列条件を満たしていない入力による証明を棄却する。

周囲条件を満たしていない場合 ある E の要素 $(i,j,k) \in E$ と $(i',j',k') \in N(i,j,k)$ に対応するコミットメントが共に ♡ であるとする。3.4 節の (4) にて、(i',j',k') に対応する組の 1 段目にあった ♡ が (i,j,k) に対応する組の 3 段目に移動する。また、(i,j,k) に対応する組の 1 段目にあった ♡ が (i',j',k') に対応する組の 3 段目に移動する。このとき、(i,j,k) 及び (i',j',k') の 3 段目には元々置いてある ♡ の右側に ♡ が少なくとも 1 枚以上並ぶことになる。よって、3.4 節の (7) において、表が ♡ である組の残りのカードを全て表にするとき、(i,j,k) 及び (i',j',k') に対応している組から ♡ が表になる。これにより、検証者は周囲条件を満たしていない入力による証明を棄却する。

以上より、対応条件・隣接条件・行列条件・周囲条件のいずれかを満たしていない入力による証明は棄却されるため、証明者が正しい解を知らない場合、検証者

は常にその証明を棄却する。

提案プロトコル 2　提案プロトコル 2 は提案プロトコル 1 と同等の条件で証明を受理及び棄却するため、提案プロトコル 1 の健全性が満たされていれば提案プロトコル 2 も健全性が満たされる。提案プロトコル 1 の健全性は既に証明済みなので、提案プロトコル 2 において証明者が正しい解を知らない場合、検証者は常にその証明を棄却する。

5.3　ゼロ知識性

証明者が正しい解に基づいた入力をした場合に関して、カードが裏から表になる操作について述べる。

提案プロトコル 1

対応条件の検証 (3)　3.2 節の (3) でコミットメントと木の数字カードが同時に表向きになる。直前の手順である 3.2 節の (2) にてカードを裏向きにしてパイルスクランブルシャッフルを行っているため、検証者は「各木の番号に対してテントが何個存在するか」以上の情報を得ることができない。検証者は対応条件として各木の番号につきそれぞれテントが 1 個存在すればよいことを知っているため、テントの位置に関する情報を新たに得ることは出来ない。

行列条件の検証 (2)　3.3 節の (2) でコミットメントと行の数字カードが同時に表向きになる。直前の手順である 3.3 節の (1) にてカードを裏向きにしてパイルスクランブルシャッフルを行っているため、検証者は「どの行にテントが何個存在するか」以上の情報を得ることができない。この数は行列条件として公開されているため、検証者はテントの位置に関する情報を新たに得ることは出来ない。

行列条件の検証 (5)　3.3 節の (5) でコミットメントと列の数字カードが同時に表向きになる。直前の手順である 3.3 節の (4) にてカードを裏向きにしてパイルスクランブルシャッフルを行っているため、検証者は「どの列にテントが何個存在するか」以上の情報を得ることができない。この数は行列条件として公開されているため、検証者はテントの位置に関する情報を新たに得ることは出来ない。

周囲条件の検証 (3)　3.4 節の (3) で行・列・木の数字カードが同時に表向きになる。直前の手順である 3.4 節の (2) にてカードを裏向きにしてパイルスクランブルシャッフルを行っているため、検証者は「どの位置にコミットメントが置かれていたか」以上の情報を得ることができない。これは 3.2 節の (1) と同じ状態であるため、検証者はテントの位置に関する情報を新たに得ることは出来ない。

周囲条件の検証 (6)　3.4 節の (6) で表になる ♡ と ♣ は、コミットメントとして入力したそれぞれの枚数と等しい。直前の手順である 3.4 節の (5) にてカードを裏向きにしてパイルスクランブルシャッフルを行っているため、検証者は「全体でテントが何個存在するか」以上の情報を得ることができない。検証者はテントの個数が木の本数と等しいことを知っているため、テントの位置に関する情報を新たに得ることは出来ない。

周囲条件の検証 (7)　3.4 節の (7) では、1 枚の ♡ につき n_{max} 枚の ♣ が表向きになる。そのため、検証者がこの状態から得られる情報は、♡ の入力された辺に対応する E の要素 $(i,j,k) \in E$ に対して、$N(i,j,k)$ に属する要素に対応するコミットメントは全て ♣ になっていることである。検証者はテントがあるマスを中心とする 3×3 の範囲に他のテントが存在しないことを知っているため、テントの位置に関する情報を新たに得ることは出来ない。

以上より、検証者は証明の過程で正しい解に関する情報を得ることができない。

提案プロトコル 2　4.1 節の (1) でカードが裏から表になる操作は 3.2 節の (3) と同じである。

実行手順 (4)　4.1 節の (4) で行・列・木の数字カードが同時に表向きになる。直前の手順である 4.1 節の (3) にてカードを裏向きにしてパイルスクランブルシャッフルを行っているため、検証者は「どの位置にコミットメントが置かれていたか」以上の情報を得ることができない。これは 3.2 節の (1) と同じ状態であるため、検証者はテントの位置に関する情報を新たに得ることは出来ない。

実行手順 (8)　4.1 節の (8) で、行の数字カードと □ が同時に表向きになる組と、列の数字カードと ♡ または ♣ が同時に表向きになる組がある。直前の手順である 4.1 節の (7) にてカードを裏向きにしてパイルスクランブルシャッフルを行っているため、検証者は「どの列にテントが何個存在するか」以上の情報を得ることができない。この数は行列条件として公開されているため、検証者はテントの位置に関する情報を新たに得ることは出来ない。

実行手順 (10)　4.1 節の (10) では行の数字カードに対応するコミットメントが表向きになる。このとき、検証者は「どの行にテントが何個存在するか」以上の情報を得ることができない。この数は行列条件として公開されているため、検証者はテントの位置に関する情報を新たに得ることは出来ない。

実行手順 (12)　4.1 節の (12) では、□ を含まないグループの 1 枚の ♡ につき n_{max} 枚の ♣ が表向きになる。そのため、検証者がこの状態から得られる情報は、♡ の入力された辺に対応する E の要素 $(i,j,k) \in E$ に対して、$N(i,j,k)$ に属する要素に対応するコミッ

トメントは全て ♣ になっていることである。検証者はテントがあるマスを中心とする 3×3 の範囲に他のテントが存在しないことを知っているため、テントの位置に関する情報を新たに得ることは出来ない。

6. おわりに

本稿では、カードベース暗号の技術を用いて論理パズルのテントアンドツリーに対する物理ゼロ知識証明プロトコルを 2 つ提案した。それらはどちらも定数回のシャッフルで実装でき、特に提案プロトコル 2 はシャッフル回数を 3 回にまで減らしている。今後の課題としては、シャッフル 2 回以下でゼロ知識証明をすることは出来るのか、出来ないのならその証明をすることである。また、その他にもマインスイーパーのような類似パズルに対するゼロ知識証明プロトコルの構築も挙げられる。

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] B. D. Boer. More efficient match-making and satisfiability the five card trick. In J.-J. Quisquater and J. Vandewalle eds., *EUROCRYPT 1989*, Vol. 434 of *LNCS*, pp. 208–217, Heidelberg, 1990. Springer.

[2] C. Crépeau and J. Kilian. Discreet solitary games. In D. R. Stinson ed., *Advances in Cryptology—CRYPTO' 93*, Vol. 773 of *LNCS*, pp. 319–330, Berlin, Heidelberg, 1994. Springer.

[3] 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.

[4] 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.

[5] D. Miyahara, H. Haneda, and T. Mizuki. Card-based zero-knowledge proof protocols for graph problems and their computational model. In Q. Huang and Y. Yu eds., *Provable and Practical Security*, Vol. 13059 of *LNCS*, pp. 136–152, Cham, 2021. Springer.

[6] 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.

[7] M. White. *Amazing Puzzles: Tents and Trees Puzzles*. Independently published, 2018.

[8] 猪狩紫雲, 駒野雄一, 水木敬明. 巡回セールスマン問題の物理的ゼロ知識証明プロトコル. マルチメディア, 分散, 協調とモバイルシンポジウム 2024 論文集, 2024:1077–1086, 2024.