

GRAPH AUTOMORPHISM SHUFFLES FROM PILE-SCRAMBLE SHUFFLES

KENGO MIYAMOTO AND KAZUMASA SHINAGAWA

ABSTRACT. A pile-scramble shuffle is one of the most effective shuffles in card-based cryptography. Indeed, many card-based protocols are constructed from pile-scramble shuffles. This article aims to study the power of pile-scramble shuffles. In particular, for any directed graph G , we introduce a new protocol called “a graph shuffle protocol for G ”, and show that it can be implemented by using pile-scramble shuffles only. Our proposed protocol requires $2(n + m)$ cards, where n and m are the numbers of vertices and arrows of G , respectively. The number of pile-scramble shuffles is $k + 1$, where $1 \leq k \leq n$ is the number of distinct degrees of vertices of G . As an application, a random cut for n cards, which is also an important shuffle, can be realized by $3n$ cards and two pile-scramble shuffles.

Secure computation; Card-based cryptography; Pile-scramble shuffles; Graph automorphisms

1. INTRODUCTION

1.1. Background. Let X, Y be finite sets, $n \in \mathbb{N}$ a natural number, and $f : X^n \rightarrow Y$ a function. Suppose that n players each having $x_i \in X$ as input wish to know an output value $f(x_1, x_2, \dots, x_n) \in Y$ without revealing anything about their own inputs beyond the output value to other players. Secure computation protocols can solve this kind of situation. Secure computation, which was formalized by Yao [56, 57], plays an important role in cryptography (cf. see the survey on secure computation by Lindell [19]).

Card-based cryptography [6, 7] is a kind of secure computation, which uses a deck of physical cards. Given a sequence of face-down cards (which is typically an encoding of input $(x_1, x_2, \dots, x_n) \in X^n$), a card-based protocol transforms it to an output sequence (which is typically an encoding of output $f(x_1, x_2, \dots, x_n) \in Y$) by a bunch of physical operations on cards. One of the features of card-based cryptography is that it allows us to understand intuitively the correctness and security of a protocol, since we can actually perform the protocol by hands. For this reason, it is expected to be used as an educational material. Indeed, some universities [5, 20, 25] have actually used card-based cryptography as an educational material.

In card-based protocols, a *shuffle*, which is a probabilistic rearrangement, is allowed to apply to a sequence of cards. It is considered as the most crucial operation in card-based protocols since randomness from shuffles is the primary tool to obtain the security of protocols. Among shuffles, a (*pile*) *random cut (RC)*, a *random bisection cut (RBC)*, and a *pile-scramble shuffle (PSS)* are the most effective shuffles¹ in card-based cryptography. Indeed, most card-based protocols are constructed with these shuffles only (cf. protocols with RCs only [2, 6, 7, 12, 15, 17, 22, 23, 32, 34, 35, 40, 44, 47, 51, 55], protocols with RBCs only [27–29, 31, 36–38, 43, 48], protocols with PSSs only [3, 10, 14, 33, 39, 41, 45, 46, 49], protocols with RCs and RBCs only [1, 16, 24, 54], protocols with RCs and PSSs only [4, 8, 18, 42, 52, 53], and protocols with RBCs and PSSs only [11, 13, 26, 50]). With this background, it is essential to study further what can be done by these shuffles.

1.2. Contribution. In this paper, we show that *graph shuffles* can be implemented with PSSs. Let G be a directed graph². A graph shuffle for G is a shuffle that arranges a sequence of cards according to an automorphism of G chosen uniformly at random. Our main contribution is to construct a card-based protocol that achieves a graph shuffle for any graph G . We call this a *graph shuffle protocol for G* . The number of cards in our protocol is $2(n + m)$, where n and m are the numbers of vertices and edges of G , respectively. The number of shuffles (i.e., PSSs) in our protocol is $|\text{Deg}_G| + 1$, where Deg_G is the set of vertex degree of G (see Section 3.1). We remark that our protocol has one drawback: it requires to compute a graph isomorphism between G and its isomorphic graph G' . In general, computing a graph isomorphism is

¹Our classification focuses on the group structure of permutations: the cyclic groups (RCs) and the symmetric groups (PSSs). Since RBCs are historically important shuffles and the intersection of RCs and PSSs, we classify them as RC, RBC, and PSS.

²We regard undirected graphs as directed graphs by identifying each undirected edge with two directed edges with opposite directions.

a complex computational task (see also Remark 3.3). We conjecture that computing a graph isomorphism is inherent in implementing a graph shuffle. We left it as an open problem whether computing a graph isomorphism can be removed or not.

A class of graph shuffles includes many interesting shuffles (see Section 3.6). Indeed, RCs, RBCs, and PSSs are special cases of graph shuffles. In particular, a RC is a graph shuffle for a directed cycle graph. A straightforward corollary of our main result is that a RC can be implemented with PSSs. Since a PSS can be implemented with RCs (cf. see Crépeau and Kilian [6]’s idea for generating a random fixed-free permutation), PSSs and RCs are essentially equivalent from the viewpoint of feasibility. It is worthwhile to mention the importance of the fact that RCs are implementable by PSSs. From the theoretical viewpoint, this shows that every protocol with RCs is transformed into a protocol with PSSs and vice versa. From the practical viewpoint, you can choose whether to use RCs or PSSs as shuffles in a protocol execution. In order to execute a RC by hand, we need to ensure that everyone must be able to verify that the rearrangement is indeed a cyclic shift while hiding the rearrangement itself. On the other hand, a PSS can be done by a rearrangement of piles in a completely randomly fashion although it requires physical envelopes as an additional tool. Which shuffle can be easily executable depends on a situation and thus there should be some cases that PSSs are more desirable than RCs.

Due to the importance of the result of RC, we improve a graph shuffle protocol for a directed cycle graph. In particular, for the directed cycle graph with n vertices, we design a graph shuffle protocol with $3n$ cards while the general protocol requires $4n$ cards.

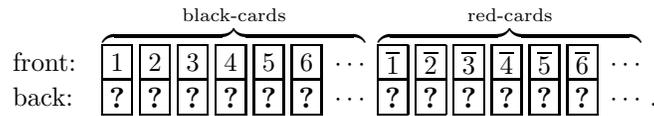
We also improve a graph shuffle protocol for an undirected cycle graph. A graph shuffle for the undirected cycle graph is equivalent to the *dihedral shuffle*, which is introduced by Niemi and Renvall [34]. For the undirected cycle graph with n vertices, we design a graph shuffle protocol with $3n$ cards while the general protocol requires $6n$ cards.

1.3. Related works. Koch and Walzer [15] showed that *uniform closed shuffles* (see Definition 2.1) can be implemented with RCs only. It is an essential milestone for implementing uniform closed shuffles. Since graph shuffles are uniformly closed, Koch and Walzer’s method allows that every graph shuffle can be done by RCs. However, we point out that their protocol requires each party somehow to generate a uniformly random element of a given group in the party’s head. This action is not allowed in the Mizuki-Shizuya model [30] which is known as the standard computational model of card-based cryptography. From this viewpoint, our protocol for graph shuffles and their protocol are based on different models of card-based cryptography. Our motivation is to implement a subclass of uniform closed shuffles in the Mizuki-Shizuya model. Besides the theoretical aspect, it is worthwhile to note that removing a randomness generation in the head brings a practical benefit for security because it is not clear how close the distribution of random elements generated in the head will be to the distribution of truly random elements.

2. PRELIMINARIES

In this section, we collect some fundamentals in card-based cryptography; see [30] for example.

2.1. Cards. Throughout this paper, we deal with physical *cards* with the symbol “?” on the backs. We use two collections of cards: *black-cards* $\boxed{1} \boxed{2} \boxed{3} \cdots$ and *red-cards* $\overline{\boxed{1}} \overline{\boxed{2}} \overline{\boxed{3}} \cdots$ as follows:



We distinguish between the natural number \overline{i} (written in red) and the natural number i (written in black). We denote by \mathbb{N}^{red} the set of all natural numbers written in red, i.e., $\mathbb{N}^{\text{red}} = \{\overline{1}, \overline{2}, \overline{3}, \dots\}$. The set \mathbb{N}^{red} is a totally ordered set by using the natural order on \mathbb{N} . We define a totally order \preceq on $\mathbb{N} \cup \mathbb{N}^{\text{red}}$ by $\alpha \preceq \beta$ if and only if

- $x, y \in \mathbb{N}$ and $x \leq y$, where $\alpha = x$ and $\beta = y$,
- $\overline{x}, \overline{y} \in \mathbb{N}^{\text{red}}$ and $x \leq y$, where $\alpha = \overline{x}$ and $\beta = \overline{y}$, or
- $\overline{x} \in \mathbb{N}^{\text{red}}$ and $y \in \mathbb{N}$, where $\alpha = \overline{x}$ and $\beta = y$.

A *deck* D is a non-empty multiset such that $\{?\} \cap D = \emptyset$. Let D be a deck. An expression $\frac{x}{?}$ (resp. $\frac{?}{x}$) with $x \in D$ is said to be a *face-up card* (resp. a *face-down card*) of D . A *lying card* y of D is the face-up card $y = \frac{x}{?}$ of D or the face-down card $y = \frac{?}{x}$ of D , and in this case, we set $\text{atom}(y) = x$. A *card-sequence* from D is a list of lying cards of D , say (x_1, \dots, x_n) , such that $\{\text{atom}(x_i) \mid i = 1, 2, \dots, n\} = D$ as multisets. For a card-sequence \mathbf{x} , we write x_i for the i -th term. A face-up card $\frac{x}{?}$ is represented by \boxed{x} , and a face-down card $\frac{?}{x}$ is represented by $\boxed{?}$. Given a card x with the expression $\frac{y}{z}$, we write $\text{front}(x) = y$, $\text{back}(x) = z$, and $\text{swap}(x) = \frac{z}{y}$. For a card-sequence $\mathbf{x} = (x_1, \dots, x_n)$ and a subset $T \subseteq \{1, 2, \dots, n\}$, we define an operator $\text{turn}_T(-)$ by

$$\text{turn}_T(\mathbf{x}) = (y_1, \dots, y_n), \quad y_i = \begin{cases} \text{swap}(x_i) & \text{if } i \in T, \\ x_i & \text{if } i \notin T. \end{cases}$$

The card-sequence $\text{front}(\mathbf{x}) = (\text{front}(x_1), \dots, \text{front}(x_n))$ is called *the visible sequence* of \mathbf{x} . Let $(\mathcal{T}, \mathcal{G})$ be a pair of a collection of subsets of $\{1, 2, \dots, n\}$ (i.e., $\mathcal{T} \subseteq 2^{\{1, 2, \dots, n\}}$) and a probability distribution on \mathcal{T} . Now, we also define an operation $\text{rflip}_{(\mathcal{T}, \mathcal{G})}(-)$ associated with the pair $(\mathcal{T}, \mathcal{G})$ by

$$\text{rflip}_{(\mathcal{T}, \mathcal{G})}(\mathbf{x}) = \text{turn}_T(\mathbf{x}),$$

where T is chosen from \mathcal{T} depending on the probability distribution \mathcal{G} . Note that if $\mathcal{T} = \{T\}$ with a subset $T \subseteq \{1, 2, \dots, n\}$, then $\text{rflip}_{(\mathcal{T}, \mathcal{G})}(-) = \text{turn}_T(-)$.

2.2. Shuffles. For a natural number $n \in \mathbb{N}$, we denote by \mathfrak{S}_n the symmetric group of degree n , that is, the group whose elements are all bijective maps from $\{1, 2, \dots, n\}$ to itself, and whose group multiplication is the composition of functions. An element of the symmetric group is called a *permutation*.

Given a card-sequence $\mathbf{x} = (x_1, \dots, x_n)$ and $\sigma \in \mathfrak{S}_n$, we have a card-sequence $\sigma(\mathbf{x})$ in the natural way:

$$\sigma(\mathbf{x}) = (x_{\sigma^{-1}(1)}, \dots, x_{\sigma^{-1}(n)}).$$

Now, we recall an operation on a card-sequence which is called a “shuffle”. Roughly speaking, a shuffle is a probabilistic reordering operation on a card-sequence. Let (Π, \mathcal{F}) be a pair of a subset of \mathfrak{S}_n and a probability distribution on Π . For a card-sequence $\mathbf{x} = (x_1, \dots, x_n)$, an operation $\text{shuffle}_{(\Pi, \mathcal{F})}(-)$ associated with the pair (Π, \mathcal{F}) is defined by

$$\text{shuffle}_{(\Pi, \mathcal{F})}(\mathbf{x}) = \sigma(\mathbf{x}).$$

Here, σ is chosen according to the probability distribution \mathcal{F} on Π . Note that when we apply a shuffle to a card-sequence, no one knows which permutation was actually chosen. We also note that if $\Pi = \{\sigma\}$ for some $\sigma \in \mathfrak{S}_n$, then $\text{shuffle}_{(\Pi, \mathcal{F})}(-) = \sigma(-)$.

Definition 2.1. A shuffle $\text{shuffle}_{(\Pi, \mathcal{F})}$ is said to be *uniform closed* if Π is closed under the multiplication of the symmetric group, and \mathcal{F} is the uniform distribution on Π .

All shuffles dealt with this paper are uniform closed shuffles.

Example 2.2. (1) For a sequence of ℓ cards, suppose that a subsequence of the sequence is divided into n piles of m cards. (It holds $\ell \geq nm$.) A *pile-scramble shuffle* (PSS for short) is a uniform closed shuffle that completely randomly permutes n piles. The following shuffle is an example of a PSS:

$$\text{PSS}_{(3,2)} : \left(\boxed{1} \boxed{2}, \boxed{3} \boxed{4}, \boxed{5} \boxed{6} \right) \xrightarrow{\sigma} \begin{cases} \left(\boxed{1} \boxed{2}, \boxed{3} \boxed{4}, \boxed{5} \boxed{6} \right) & \text{if } \sigma = \text{id}, \\ \left(\boxed{1} \boxed{2}, \boxed{5} \boxed{6}, \boxed{3} \boxed{4} \right) & \text{if } \sigma = (2\ 3), \\ \left(\boxed{3} \boxed{4}, \boxed{5} \boxed{6}, \boxed{1} \boxed{2} \right) & \text{if } \sigma = (1\ 3\ 2), \\ \left(\boxed{3} \boxed{4}, \boxed{1} \boxed{2}, \boxed{5} \boxed{6} \right) & \text{if } \sigma = (1\ 2), \\ \left(\boxed{5} \boxed{6}, \boxed{1} \boxed{2}, \boxed{3} \boxed{4} \right) & \text{if } \sigma = (1\ 2\ 3), \\ \left(\boxed{5} \boxed{6}, \boxed{3} \boxed{4}, \boxed{1} \boxed{2} \right) & \text{if } \sigma = (1\ 3). \end{cases}$$

We use $\text{PSS}_{(n,m)}$ to denote a PSS for n piles each having m cards. We remark that $\text{PSS}_{(n,m)}$ can be easily implemented by putting each pile into each physical envelope and then permute them.

(2) Let $\pi_k \in \mathfrak{S}_n$ be the permutation

$$\pi_k = \begin{pmatrix} 1 & 2 & \cdots & k & k+1 & \cdots & n \\ n-k+1 & n-k+2 & \cdots & n & 1 & \cdots & n-k \end{pmatrix},$$

and set $\Pi = \{\pi_k \mid k = 1, 2, \dots, n\}$. This uniform closed shuffle $\text{shuffle}_{(\Pi, \mathcal{F})}$ is called a *random cut* (RC for short).

2.3. Protocols. Mizuki and Shizuya [30] define the formal definition of a card-based protocol via an abstract machine. In this section, we recall the definition of a card-based protocol and introduce a shuffle protocol, which is a particular card-based protocol realizing a shuffle.

2.3.1. Card-based protocols. To put it briefly, a ‘‘protocol’’ is a Turing machine that chooses one of the following operations to be applied to a card-sequence \mathbf{x} : turning ($\mathbf{x} \mapsto \text{rflip}_{(\mathcal{T}, \mathcal{G})}(\mathbf{x})$) or shuffling ($\mathbf{x} \mapsto \text{shuffle}_{(\Pi, \mathcal{F})}(\mathbf{x})$).

For a deck D , the set of all card-sequences from D will be denoted by Seq^D . Then *the visible sequence set* Vis^D is defined as the set of all sequences $\text{front}(\mathbf{x})$ for $\mathbf{x} \in \text{Seq}^D$. We also define the sets of the actions:

$$\text{turn}^n = \{\text{turn}_T(-) \mid T \subseteq \{1, 2, \dots, n\}\},$$

$$\text{perm}^n = \{\sigma(-) \mid \sigma \in \mathfrak{S}_n\},$$

$$\text{SP}^n = \{\text{shuffle}_{(\Pi, \mathcal{F})}(-) \mid \mathcal{F} \text{ is a probability distribution on } \Pi \in 2^{\mathfrak{S}_n}\}, \text{ and}$$

$$\text{TP}^n = \{\text{rflip}_{(\mathcal{T}, \mathcal{G})}(-) \mid \mathcal{G} \text{ is a probability distribution on } \mathcal{T} \subseteq 2^{\{1, 2, \dots, n\}}\}.$$

A protocol is a Markov chain, that is, a stochastic model describing a sequence of possible actions in which the probability of each action depends only on the state attained in the previous event. Let Q be a finite set with two distinguished states, which are called *an initial state* q_0 and *a final state* q_f .

Definition 2.3. A *card-based protocol* is a quadruple $\mathcal{P} = (D, U, Q, A)$, where $U \subseteq \text{Seq}^D$ is an input set and A is a partial action function

$$A: \begin{array}{ccc} (Q \setminus \{q_f\}) \times \text{Vis}^D & \longrightarrow & Q \times (\text{turn}^n \cup \text{perm}^n \cup \text{SP}^n \cup \text{TP}^n), \\ (q, \mathbf{y}) & \longmapsto & (q', \text{act}_{q, \mathbf{y}}) \end{array}$$

which depends only on the current state and visible sequence, specifying the next state and an operation on the card-sequence from $(\text{turn}^n \cup \text{perm}^n \cup \text{SP}^n \cup \text{TP}^n)$, such that $A(q_0, \text{front}(\mathbf{x}))$ is defined if $\mathbf{x} \in U$. For a state $q \in Q \setminus \{q_f\}$ and a visible sequence $\mathbf{y} = \text{front}(\mathbf{x}) \in \text{Vis}^D$ such that $A(q, \mathbf{y}) = (q', \text{act}_{q, \mathbf{y}})$, we obtain the next state $(q', \text{front}(\text{act}_{q, \mathbf{y}}(\mathbf{x})))$. By the above process, if we have $(q_f, \mathbf{X}) \in Q \times \text{Vis}^D$ for some $\mathbf{X} \in \text{Vis}^D$, the protocol \mathcal{P} terminates.

Let $\mathcal{P} = (D, U, Q, A)$ be a card-based protocol. For an execution of \mathcal{P} with an input card-sequence $\mathbf{x}^{(0)} \in U$, we obtain a sequence of results of actions as follows:

$$(q_0, \mathbf{x}^{(0)}) \longmapsto (q_1, \mathbf{x}^{(1)}) \longmapsto (q_2, \mathbf{x}^{(2)}) \longmapsto (q_3, \mathbf{x}^{(3)}) \longmapsto \cdots,$$

where $\mathbf{x}^{(i)} = \text{act}_{q_{i-1}, \text{front}(\mathbf{x}^{(i-1)})}(\mathbf{x}^{(i-1)})$ for $i \geq 1$. Here, q_i ($i = 0, 1, 2, \dots$) are not necessarily distinct. If the action function value $A(q_i, \mathbf{x}^{(i)})$ is undefined for some $i \in \mathbb{N}$, we say that ‘‘ \mathcal{P} aborts at Step i in the execution’’. Note that even for the same input card-sequence $\mathbf{x}^{(0)}$, the obtained chains may be different for each execution. If the protocol \mathcal{P} terminates for an input card-sequence $\mathbf{x}^{(0)}$, then we have a chain of results as follows:

$$(q_0, \mathbf{x}^{(0)}) \longmapsto (q_1, \mathbf{x}^{(1)}) \longmapsto (q_2, \mathbf{x}^{(2)}) \longmapsto (q_2, \mathbf{x}^{(3)}) \longmapsto \cdots \longmapsto (q_f, \mathbf{x}^{(\ell)}).$$

In this case, $\mathbf{x}^{(0)}$ is called *an initial sequence*, $\mathbf{x}^{(\ell)}$ is called *a final sequence*, and the sequence

$$(\mathbf{y}^{(0)}, \mathbf{y}^{(1)}, \dots, \mathbf{y}^{(\ell)}),$$

where $\mathbf{y}^{(i)} = \text{front}(\mathbf{x}^{(i)})$, is called *a visible sequence-trace* of \mathcal{P} . We denote by $\text{Fin}(\mathcal{P})$ the set of all final sequences, which is obtained by \mathcal{P} .

Example 2.4. Let us consider the following. Take the deck $D = \{1, 2, 3, 4\}$, and hence use as follows:

$$\begin{array}{l} \text{front: } \boxed{1} \quad \boxed{2} \quad \boxed{3} \quad \boxed{4} \\ \text{back: } \boxed{?} \quad \boxed{?} \quad \boxed{?} \quad \boxed{?} . \end{array}$$

Now, we give a card-based protocol $\mathcal{P} = \left(D, \left\{ \left(\frac{1}{?}, \frac{2}{?}, \frac{3}{?}, \frac{4}{?} \right) \right\}, \{q_0, q_1, q_2, q_f\}, \mathbf{A} \right)$ such that

$$\begin{aligned} \mathbf{A}(q_0, 1234) &= (q_1, \text{turn}_{\{1,2,3,4\}}(-)), \\ \mathbf{A}(q_1, ????) &= (q_2, (1\ 3)(-)), \\ \mathbf{A}(q_2, ????) &= (q_f, \text{turn}_{\{3\}}(-)). \end{aligned}$$

In this case, the card-sequence $\boxed{1}\boxed{2}\boxed{3}\boxed{4}$ is changed by the protocol \mathcal{P} as follows:

$$\boxed{1}\boxed{2}\boxed{3}\boxed{4} \longrightarrow \overset{1}{\boxed{?}}\overset{2}{\boxed{?}}\overset{3}{\boxed{?}}\overset{4}{\boxed{?}} \longrightarrow \overset{3}{\boxed{?}}\overset{2}{\boxed{?}}\overset{1}{\boxed{?}}\overset{4}{\boxed{?}} \longrightarrow \overset{3}{\boxed{?}}\overset{2}{\boxed{?}}\boxed{1}\overset{4}{\boxed{?}}.$$

Thus, the final sequence is $\boxed{?}\boxed{?}\boxed{1}\boxed{?}$.

2.3.2. Shuffle protocols. A shuffle protocol³ is a card-based protocol realizing a shuffle operation. It takes a card-sequence $\mathbf{x} = (x_1, x_2, \dots, x_n)$ such that $\text{back}(\mathbf{x}) = (?, ?, \dots, ?)$ as input and outputs $\mathbf{y} = (y_1, y_2, \dots, y_n)$ such that $\mathbf{y} = \sigma(\mathbf{x})$ for a permutation σ is chosen from \mathfrak{S}_n depending on some probability distribution:

$$\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\mathbf{x}} \underbrace{\boxed{h_1}\boxed{h_2}\cdots\boxed{h_k}}_{\mathbf{h}} \longrightarrow \underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\mathbf{y}} \underbrace{\boxed{h_1}\boxed{h_2}\cdots\boxed{h_k}}_{\mathbf{h}},$$

where \mathbf{h} is a card-sequence of helping cards. Informally speaking, the correctness requires $\mathbf{y} = \text{shuffle}_{(\Pi, \mathcal{F})}(\mathbf{x})$ and the security requires that no one learns nothing about the chosen permutation $\sigma \in \Pi$.

Definition 2.5. Let $D_{\text{inp}}, D_{\text{help}}$ be decks, U_{inp} an input set from D_{inp} , and $\mathbf{h} \in \text{Seq}^{D_{\text{help}}}$ a card-sequence from D_{help} . We define an input set U from $D = D_{\text{inp}} \cup D_{\text{help}}$ by $U = \{(\mathbf{x}, \mathbf{h}) \mid \mathbf{x} \in U_{\text{inp}}\}$. A card-based protocol $\mathcal{P} = (D, U, Q, \mathbf{A})$ is said to be a *shuffle protocol* if the following conditions are satisfied:

- (a) \mathcal{P} always terminates within a fixed number of steps, i.e., it is a finite-runtime protocol;
- (b) for any input sequence $(\mathbf{x}, \mathbf{h}) \in U$ and for any final sequence $\mathbf{y} \in \text{Fin}(\mathcal{P})$ of the form $\mathbf{y} = (\mathbf{x}', \mathbf{h})$, there exists a permutation $\sigma \in \mathfrak{S}_{|D_{\text{inp}}|}$ such that $\mathbf{x}' = \sigma(\mathbf{x})$;
- (c) for any input sequence $(\mathbf{x}, \mathbf{h}) \in U$, any card contained in \mathbf{x} has not been turned at any step of a protocol execution.

We say that \mathcal{P} implements a shuffle $\text{shuffle}_{(\Pi, \mathcal{F})}$ if every permutation σ in (b) belongs to Π and it is chosen according to the distribution \mathcal{F} . We say that \mathcal{P} is secure if for any $\mathbf{x} \in U_{\text{inp}}$, a random variable of σ is stochastically independent of the random variable of the visible sequence-trace of \mathcal{P} .

3. GRAPH SHUFFLE PROTOCOLS

In this section, we construct a card-based protocol called the graph shuffle protocol for a directed graph. First, we introduce a graph shuffle in Subsection 3.1. Second, we construct the graph shuffle protocol, which is a shuffle protocol for any graph shuffle in Subsection 3.2. We note that our protocol requires PSSs only.

3.1. Graph shuffle. First, we recall some fundamentals from graph theory; for example, see [9].

A directed graph is a quadruple $G = (V_G, E_G, s_G, t_G)$ consisting of two sets V_G, E_G and two maps $s_G, t_G : E_G \rightarrow V_G$. Each element of V_G (resp. E_G) is called a vertex (resp. an edge). Note that there might be two or more edges from a to b for some $a, b \in V_G$, that is, G admits multiple edges. For an edge $e \in E_G$, we call $s_G(e)$ (resp. $t_G(e)$) the source (resp. the target) of e . We will commonly write $a \xrightarrow{e} b$ or $e : a \rightarrow b$ to indicate that an edge e has the source a and the target b , and identify e with a pair $(s_G(e), t_G(e))$. A directed graph G is finite if two sets V_G and E_G are finite sets. In this paper, a graph means a finite directed graph with n vertices and m edges.

³Koch and Walzer [15] considered a similar notion and proposed a protocol for any uniform closed shuffles. The main difference of their model and our model is that their model allows a randomness generation in the head (see Section 1.3 in Introduction).

Let G be a graph. For a vertex $v \in V_G$, we define the following three functions:

$$\text{in}(v) = |\{e \in E_G \mid v = t_G(e)\}|, \quad \text{out}(v) = |\{e \in E_G \mid v = s_G(e)\}|, \quad \text{and} \quad \text{deg}(v) = \text{in}(v) + \text{out}(v).$$

The number $\text{deg}(v)$ is called *the degree* of v . We set $\text{Deg}_G = \{\text{deg}(v) \mid v \in V_G\}$.

For graphs G and G' , a pair $f = (f_0, f_1) : G \rightarrow G'$ consisting of maps $f_0 : V_G \rightarrow V_{G'}$ and $f_1 : E_G \rightarrow E_{G'}$ is a morphism of graphs if $(f_0 \times f_0) \circ (s_G \times t_G) = (s_{G'} \times t_{G'}) \circ f_1$ holds. In addition, if f_0 and f_1 are bijective, f is called *an isomorphism* of graphs. In this case, we say that G and G' are isomorphic as graphs. In other words, two graphs G and G' are isomorphic as graphs when $x \xrightarrow{e} y$ in G exists if and only if $f_0(x) \xrightarrow{f_1(e)} f_0(y)$ exists in G' . We denote by $\text{Iso}(G, G')$ the set of all isomorphisms from G to G' , and $\text{Iso}_0(G, G')$ the set of all f_0 such that $(f_0, *) \in \text{Iso}(G, G')$. For a graph G , an isomorphism from G to itself is called *an automorphism*. We denote by $\text{Aut}(G)$ the set of all automorphisms of G , and $\text{Aut}_0(G)$ the set of all f_0 such that there exists $(f_0, f_1) \in \text{Aut}(G)$. Then it is obvious that $\text{Aut}(G)$ is a group by the composition of maps. Furthermore, the group structure of $\text{Aut}(G)$ induces the group structure on $\text{Aut}_0(G)$. Note that, if G has no multiple edges, then an automorphism $f = (f_0, f_1) \in \text{Aut}(G)$ is determined by f_0 . In the case that G is an undirected graph, one can transform G into the following directed graph \vec{G} :

$$V_{\vec{G}} = V_G, \quad E_{\vec{G}} = \{i \rightarrow j, j \rightarrow i \mid (i, j) \in E_G\}.$$

Definition 3.1. Let G be a graph. The uniform closed shuffle $\text{shuffle}_{(\text{Aut}_0(G), \mathcal{F})}$ is called *the graph shuffle* for G over n cards. (Recall that G has n vertices.)

3.2. Graph shuffle protocols. In this subsection, we construct a graph shuffle protocol, which is a shuffle protocol of the graph shuffle for a graph $G = (V_G, E_G, s_G, t_G)$. We set $V_G = \{1, 2, \dots, n\}$. Let $D_{\text{inp}} = \{x_1, x_2, \dots, x_n\}$ be any deck, and U_{inp} any input set from D_{inp} . We set a card-sequence h of helping cards as follows:

$$h = \boxed{1} \boxed{2} \boxed{3} \cdots \boxed{n} \overbrace{\boxed{1} \cdots \boxed{1}}^{\text{deg}(1)} \overbrace{\boxed{2} \cdots \boxed{2}}^{\text{deg}(2)} \overbrace{\boxed{3} \cdots \boxed{3}}^{\text{deg}(3)} \cdots \overbrace{\boxed{n} \cdots \boxed{n}}^{\text{deg}(n)}.$$

Thus the deck of helping cards is $D_{\text{help}} = \{\overline{1}, \overline{2}, \dots, \overline{n}, 1^{\text{deg}(1)}, 2^{\text{deg}(2)}, \dots, n^{\text{deg}(n)}\}$, where the superscript denotes the number of the symbol in the deck D_{help} . The deck D is the union of D_{inp} and D_{help} as multisets and it consists of $2(n + m)$ symbols.

For an input card-sequence $x \in U_{\text{inp}}$, our protocol proceeds as follows:

- (1) Place the cards as follows.

$$\underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_x \underbrace{\boxed{1} \boxed{2} \boxed{3} \cdots \boxed{n} \boxed{1} \cdots \boxed{1} \boxed{2} \cdots \boxed{2} \boxed{3} \cdots \boxed{3} \cdots \cdots \boxed{n} \cdots \boxed{n}}_h.$$

- (2) For each i , we define $\text{pile}[i]$ by

$$\text{pile}[i] = \left(\overbrace{\frac{?}{i}, \frac{?}{i}, \dots, \frac{?}{i}}^{\text{deg}(i)} \right) = \overbrace{\boxed{\frac{?}{i}} \boxed{\frac{?}{i}} \cdots \boxed{\frac{?}{i}}}^{\text{deg}(i)}.$$

Arrange the card-sequence as $(x, \text{pile}[1], \text{pile}[2], \text{pile}[3], \dots, \text{pile}[n])$, that is:

$$\underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_x \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[1]} \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[2]} \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[3]} \cdots \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[n]}.$$

- (3) For each $d \in \text{Deg}_G$, we set $V_G^{(d)} = \{v_1^{(d)}, v_2^{(d)}, \dots, v_{\ell_d}^{(d)}\}$ for all vertices with degree d , and apply $\text{PSS}_{(\ell_d, d+1)}$ to the card-sequence $(\text{pile}[v_1^{(d)}], \text{pile}[v_2^{(d)}], \dots, \text{pile}[v_{\ell_d}^{(d)}])$. Then we obtain a card-sequence

$$\underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_x \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[\alpha_1]} \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[\alpha_2]} \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[\alpha_3]} \cdots \underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_{\text{pile}[\alpha_n]}.$$

Let $\sigma \in \mathfrak{S}_n$ be the chosen permutation such that $\alpha_i = \sigma^{-1}(i)$.

- (4) For each $i \in V_G$ and $j \rightarrow k \in E_G$, we set $\text{vertex}[i] = \left(\frac{?}{\alpha_i}, x_i \right)$ and $\text{edge}[j \rightarrow k] = \left(\frac{?}{\alpha_j}, \frac{?}{\alpha_k} \right)$, respectively. Arrange the card-sequence⁴ as follows:

$$\underbrace{\boxed{?} \boxed{?}}_{\text{vertex}[1]} \underbrace{\boxed{?} \boxed{?}}_{\text{vertex}[2]} \cdots \underbrace{\boxed{?} \boxed{?}}_{\text{vertex}[n]} \underbrace{\boxed{?} \boxed{?}}_{\text{edge}[e_1]} \underbrace{\boxed{?} \boxed{?}}_{\text{edge}[e_2]} \cdots \underbrace{\boxed{?} \boxed{?}}_{\text{edge}[e_m]},$$

where $E_G = \{e_1, e_2, \dots, e_m\}$.

- (5) Apply $\text{PSS}_{(m+n,2)}$ to the card-sequence as follows:

$$\left| \underbrace{\boxed{?} \boxed{?}}_{\text{vertex}[1]} \right| \left| \underbrace{\boxed{?} \boxed{?}}_{\text{vertex}[2]} \right| \cdots \left| \underbrace{\boxed{?} \boxed{?}}_{\text{edge}[e_m]} \right| \rightarrow \boxed{?} \boxed{?} \boxed{?} \boxed{?} \cdots \boxed{?} \boxed{?} \boxed{?} \boxed{?}.$$

- (6) For each pile, turn over the left card, and if it is a black-card, turn over the right card. Then sort $n + m$ piles⁵ so that the left card is in ascending order via \preceq as follows:

$$\boxed{\bar{1}} \boxed{?} \boxed{\bar{2}} \boxed{?} \boxed{\bar{3}} \boxed{?} \cdots \boxed{\bar{n}} \boxed{?} \boxed{i_1} \boxed{j_1} \boxed{i_2} \boxed{j_2} \boxed{i_3} \boxed{j_3} \cdots \boxed{i_m} \boxed{j_m},$$

where $i_1 \preceq i_2 \preceq i_3 \preceq \cdots \preceq i_m$.

- (7) We define a graph G' by $V_{G'} = V_G$ and $E_{G'} = \{i_1 \rightarrow j_1, i_2 \rightarrow j_2, i_3 \rightarrow j_3, \dots, i_m \rightarrow j_m\}$.

- (8) Take an isomorphism $\psi : G \rightarrow G'$, and set $\beta_i := \psi_0^{-1}(i)$. Let y_i be the right next card of $\boxed{\beta_i}$ and $y = (y_1, y_2, \dots, y_n)$. Arrange the card-sequence as follows:

$$\underbrace{\boxed{?} \boxed{?} \cdots \boxed{?}}_y \underbrace{\boxed{\bar{1}} \boxed{\bar{2}} \boxed{\bar{3}} \cdots \boxed{\bar{n}} \boxed{1} \cdots \boxed{1} \boxed{2} \cdots \boxed{2} \boxed{3} \cdots \boxed{3} \cdots \cdots \boxed{n} \cdots \boxed{n}}_h.$$

The output card-sequence for the input x is y .

Remark 3.2. Regarding the number of cards, the number of cards in the proposed protocol is $2n + 2m$, of which $n + 2m$ are helping cards. As for the number of shuffles, it is $|\text{Deg}_G| + 1$, and all of them are PSSs. We remark that the PSSs in Step (3) can be executed in parallel.

Remark 3.3. In Step (8), given two isomorphic graphs G and G' , we need to solve the problem of finding one specific isomorphism between them. However, no polynomial-time algorithm for this problem has been found so far in general. On the other hand, there exist polynomial-time algorithms to find isomorphisms for some specific graph classes. In addition, for small specific examples, an isomorphism can be computed by using a mathematical library for graph computation (e.g., Nauty [21]).

3.3. Proof of correctness. Let $x = (x_1, x_2, \dots, x_n)$ be an input sequence and $y = (y_1, y_2, \dots, y_n)$ a random variable of an output sequence of the protocol when x is given as input. Fix a graph G' , which is defined in Step (7) following the opened result in Step (6). Let $\sigma \in \mathfrak{S}_n$ be a random variable of the permutation chosen by the PSSs in Step (3) such that $\alpha_i = \sigma^{-1}(i)$ for all $i \in V_G$. Since an edge $i \rightarrow j \in E_G$ of G corresponds to an edge $\alpha_i \rightarrow \alpha_j = \sigma^{-1}(i) \rightarrow \sigma^{-1}(j) \in E_{G'}$ of G' , there is an isomorphism $\phi = (\phi_0, \phi_1) \in \text{Iso}(G, G')$ such that $\phi_0 = \sigma^{-1}$. From the property of the PSSs, the permutation ϕ_0 is a uniform random variable on $\text{Iso}_0(G, G')$. Let $\psi = (\psi_0, \psi_1) \in \text{Iso}(G', G)$ be a random variable of the isomorphism chosen in Step (9).

We first claim that $y = \psi_0 \circ \phi_0(x)$. This is shown by observing a sequence of red cards $\boxed{\bar{1}} \boxed{\bar{2}} \boxed{\bar{3}} \cdots \boxed{\bar{n}}$.

Hereafter, for the sake of clarity, we do not distinguish face-up $\frac{\bar{i}}{?}$ and face-down $\frac{?}{\bar{i}}$ and use “ \bar{i} ” to denote the red card i . In Step (1), the sequence of red cards omitting other cards is $(\bar{1}, \bar{2}, \dots, \bar{n})$. In Steps (3), (6), and (8), it is arranged as follows:

$$\underbrace{(\bar{1}, \bar{2}, \dots, \bar{n})}_{\text{Step (1)}} \xrightarrow{\phi_0^{-1}} \underbrace{(\overline{\alpha_1}, \overline{\alpha_2}, \dots, \overline{\alpha_n})}_{\text{Step (3)}} \xrightarrow{\phi_0} \underbrace{(\bar{1}, \bar{2}, \dots, \bar{n})}_{\text{Step (6)}} \xrightarrow{\psi_0} \underbrace{(\overline{\beta_1}, \overline{\beta_2}, \dots, \overline{\beta_n})}_{\text{Step (8)}}.$$

Since the input sequence x is arranged as $((\overline{\alpha_1}, x_1), (\overline{\alpha_2}, x_2), \dots, (\overline{\alpha_n}, x_n))$ in Step (3), the permutation $\psi_0 \circ \phi_0$ is applied to x . Thus, it holds $y = \psi_0 \circ \phi_0(x)$. We note that $\psi \circ \phi_0$ is an automorphism of G .

⁴Note that this rearrangement is possible without looking under the cards since the subscripts of α_i are public information.

⁵It is not essential the order of pairs of helping cards.

It remains to prove that the distribution of $\psi_0 \circ \phi_0 \in \text{Aut}_0(G)$ is uniformly random. We note that given the graph G' , the distributions of ϕ_0 and ψ_0 are independent. This is because the choice of ψ_0 depends on the opened symbols in Step (6) only, and they are independent of ϕ_0 due to the PSS in Step (5). Thus, we can change the order of choice without harming the distributions of ϕ_0, ψ_0 : first, ψ_0 is chosen, and then ϕ_0 is chosen. Since the distribution of $\phi_0 \in \text{Iso}_0(G, G')$ is uniformly random, it is sufficient to show that the function

$$\begin{aligned} \Phi : \text{Iso}_0(G, G') &\longrightarrow \text{Aut}_0(G) \\ \phi_0 &\longmapsto \psi_0 \circ \phi_0 \end{aligned}$$

is bijective.

We first prove that Φ is injective. Suppose that $\Phi(\phi'_0) = \Phi(\phi''_0)$ for some $\phi'_0, \phi''_0 \in \text{Iso}_0(G, G')$, that is, $\psi_0 \circ \phi'_0 = \psi_0 \circ \phi''_0$. Since ψ_0 is a bijection, $\phi'_0 = \phi''_0$ holds. Thus Φ is injective. We next prove that Φ is surjective. For any $\tau \in \text{Aut}_0(G)$, we have

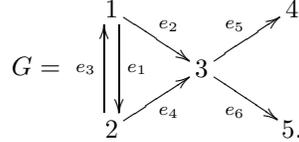
$$\tau = \psi_0 \circ \psi_0^{-1} \circ \tau = \Phi(\psi_0^{-1} \circ \tau).$$

It yields that Φ is surjective. Therefore, Φ is bijective.

This shows that the distribution of $\psi \circ \sigma^{-1}$ is uniformly random, and hence our protocol is correct.

3.4. Proof of security. In the proof of the correctness, we have already claimed that the distribution of the opened symbols in Step (6) is independent of σ due to the PSS in Step (5). Since cards are opened in Step (6) only, this shows a distribution of the permutation $\psi \circ \sigma^{-1} \in \text{Aut}(G)$ is independent of the distribution of the visible sequence-trace of our protocol. Therefore, our protocol is secure.

3.5. Example of our protocol for a graph. Let G be a directed graph with 5 vertices as follows:

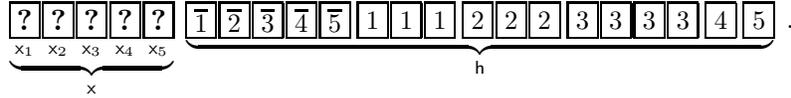


We perform our graph shuffle protocol for G . Let D_{inp} be an arbitrary deck with $D_{\text{inp}} = \{x_1, x_2, x_3, x_4, x_5\}$. The card-sequence h of helping cards is defined as follows:

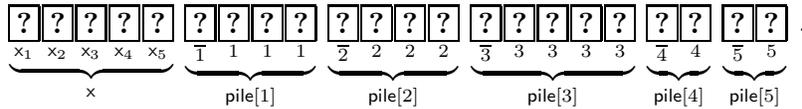
$$h = \boxed{1} \boxed{2} \boxed{3} \boxed{4} \boxed{5} \boxed{1} \boxed{1} \boxed{1} \boxed{2} \boxed{2} \boxed{2} \boxed{3} \boxed{3} \boxed{3} \boxed{3} \boxed{4} \boxed{5}.$$

Set $D_{\text{help}} = \{\bar{1}, \bar{2}, \bar{3}, \bar{4}, \bar{5}, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3, 4, 5\}$ and $D = D_{\text{inp}} \cup D_{\text{help}}$. For an input card-sequence $x = (x_1, x_2, x_3, x_4, x_5) \in U$, the graph shuffle protocol proceeds as follows:

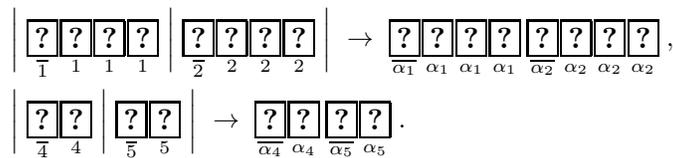
(1) Place the cards such as:



(2) Arrange the card-sequence as follows:



(3) Perform $\text{PSS}_{(2,4)}$ and $\text{PSS}_{(2,2)}$ as follows:



By setting $\alpha_3 = 3$, we have the following card-sequence:

$$\underbrace{\begin{array}{|c|c|c|c|c|} \hline ? & ? & ? & ? & ? \\ \hline x_1 & x_2 & x_3 & x_4 & x_5 \\ \hline \end{array}}_x \underbrace{\begin{array}{|c|c|c|c|} \hline ? & ? & ? & ? \\ \hline \alpha_1 & \alpha_1 & \alpha_1 & \alpha_1 \\ \hline \end{array}}_{\text{pile}[\alpha_1]} \underbrace{\begin{array}{|c|c|c|c|} \hline ? & ? & ? & ? \\ \hline \alpha_2 & \alpha_2 & \alpha_2 & \alpha_2 \\ \hline \end{array}}_{\text{pile}[\alpha_2]} \underbrace{\begin{array}{|c|c|c|c|c|} \hline ? & ? & ? & ? & ? \\ \hline \alpha_3 & \alpha_3 & \alpha_3 & \alpha_3 & \alpha_3 \\ \hline \end{array}}_{\text{pile}[\alpha_3]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_4 & \alpha_4 \\ \hline \end{array}}_{\text{pile}[\alpha_4]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_5 & \alpha_5 \\ \hline \end{array}}_{\text{pile}[\alpha_5]} .$$

(4) Arrange the card-sequence as follows:

$$\underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_1 & \alpha_1 \\ \hline \end{array}}_{\text{vertex}[1]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_2 & \alpha_2 \\ \hline \end{array}}_{\text{vertex}[2]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_3 & \alpha_3 \\ \hline \end{array}}_{\text{vertex}[3]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_4 & \alpha_4 \\ \hline \end{array}}_{\text{vertex}[4]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_5 & \alpha_5 \\ \hline \end{array}}_{\text{vertex}[5]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_1 & \alpha_2 \\ \hline \end{array}}_{\text{edge}[1 \rightarrow 2]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_1 & \alpha_3 \\ \hline \end{array}}_{\text{edge}[1 \rightarrow 3]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_2 & \alpha_1 \\ \hline \end{array}}_{\text{edge}[2 \rightarrow 1]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_2 & \alpha_3 \\ \hline \end{array}}_{\text{edge}[2 \rightarrow 3]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_3 & \alpha_4 \\ \hline \end{array}}_{\text{edge}[3 \rightarrow 4]} \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_3 & \alpha_5 \\ \hline \end{array}}_{\text{edge}[3 \rightarrow 5]} .$$

(5) Apply $\text{PSS}_{(11,2)}$ to the card-sequence as follows:

$$\left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_1 & \alpha_1 \\ \hline \end{array}}_{\text{vertex}[1]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_2 & \alpha_2 \\ \hline \end{array}}_{\text{vertex}[2]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_3 & \alpha_3 \\ \hline \end{array}}_{\text{vertex}[3]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_4 & \alpha_4 \\ \hline \end{array}}_{\text{vertex}[4]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline x_5 & \alpha_5 \\ \hline \end{array}}_{\text{vertex}[5]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_1 & \alpha_2 \\ \hline \end{array}}_{\text{edge}[1 \rightarrow 2]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_1 & \alpha_3 \\ \hline \end{array}}_{\text{edge}[1 \rightarrow 3]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_2 & \alpha_1 \\ \hline \end{array}}_{\text{edge}[2 \rightarrow 1]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_2 & \alpha_3 \\ \hline \end{array}}_{\text{edge}[2 \rightarrow 3]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_3 & \alpha_4 \\ \hline \end{array}}_{\text{edge}[3 \rightarrow 4]} \right| \left| \underbrace{\begin{array}{|c|c|} \hline ? & ? \\ \hline \alpha_3 & \alpha_5 \\ \hline \end{array}}_{\text{edge}[3 \rightarrow 5]} \right| .$$

(6) For each pile, turn over the left card, and if it is a black-card, turn over the right card. The following card-sequence is an example outcome:

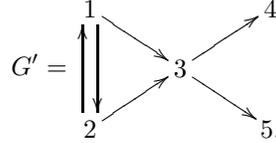
$$\overline{5} \ ? \ 1 \ 3 \ 2 \ 3 \ \overline{4} \ ? \ \overline{2} \ ? \ 2 \ 1 \ \overline{1} \ ? \ 3 \ 5 \ 3 \ 4 \ 1 \ 2 \ \overline{3} \ ? .$$

Sort 11 piles so that the left card is in ascending order via \prec as follows:

$$\overline{1} \ ? \ \overline{2} \ ? \ \overline{3} \ ? \ \overline{4} \ ? \ \overline{5} \ ? \ 1 \ 3 \ 1 \ 2 \ 2 \ 3 \ 2 \ 1 \ 3 \ 5 \ 3 \ 4 .$$

$$\begin{array}{cccccc} y'_2 & y'_1 & y'_3 & y'_4 & y'_5 & \end{array}$$

(7) Define a graph G' by $V_{G'} = \{1, 2, 3, 4, 5\}$ and $E_{G'} = \{1 \rightarrow 3, 1 \rightarrow 2, 2 \rightarrow 3, 2 \rightarrow 1, 3 \rightarrow 4, 3 \rightarrow 5\}$;



(8) Take an isomorphism $\psi : G \rightarrow G'$ defined by

$$1 \mapsto 2, \quad 2 \mapsto 1, \quad 3 \mapsto 3, \quad 4 \mapsto 4, \quad 5 \mapsto 5.$$

Arrange the above card-sequence as follows:

$$\underbrace{\begin{array}{|c|c|c|c|c|} \hline ? & ? & ? & ? & ? \\ \hline y'_2 & y'_1 & y'_3 & y'_4 & y'_5 \\ \hline \end{array}} \underbrace{\overline{1} \ \overline{2} \ \overline{3} \ \overline{4} \ \overline{5} \ 1 \ 1 \ 1 \ 2 \ 2 \ 2 \ 3 \ 3 \ 3 \ 3 \ 4 \ 5}_{\text{h}} .$$

The output card-sequence for the input x is $(y'_2, y'_1, y'_3, y'_4, y'_5)$.

3.6. Implication of our protocol.

In this subsection, we consider several interesting graph shuffles. We first observe that a RC for n cards are graph shuffles for the directed n -cycle graph \vec{C}_n (see Section 4.1). Since it holds $2n + 2m = 4n$ and $|\text{Deg}_G| + 1 = 2$, a RC can be done by $4n$ cards and two PSSs. In Section 4.1, the number of cards is improved to $3n$. We remark that our graph shuffle protocol works even for a sequence of piles each having equivalent number of face-down cards. Thus a pile-shifting shuffle (i.e., a pile-version of RC) can be done by the same number of helping cards. In particular, for a pile-shifting shuffle for n piles of m cards, it can be done by $nm + 3n$ cards and two PSSs. We note that PSSs and RBCs are graph shuffles for graphs with no edges in this sense.

A graph shuffle for the undirected n -cycle graph C_n is equivalent to the *dihedral shuffle*, which is introduced by Niemi and Renvall [34]. Since it holds $2n + 2m = 5n$ and $|\text{Deg}_G| + 1 = 2$, our result implies that a RC can be done by $5n$ cards and two PSSs. In Section 4.2, the number of cards is improved to $3n$, although the number of PSSs is increased to three.

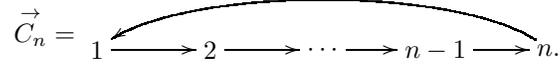
For a cyclic group $\Pi = \langle (1 \ 2)(3 \ 4 \ 5 \ 6) \rangle$, a uniform closed shuffle (shuffle, Π, \mathcal{F}) is a graph shuffle for G where $V_G = \{1, 2, 3, 4, 5, 6\}$ and $E_G = E_1 \cup E_2 \cup E_3$ with $E_1 = \{1 \rightarrow 2, 2 \rightarrow 1\}$, $E_2 = \{3 \rightarrow 4, 4 \rightarrow 5, 5 \rightarrow 6, 6 \rightarrow 4\}$, and $E_3 = \{1 \rightarrow 3, 1 \rightarrow 5, 2 \rightarrow 4, 2 \rightarrow 6\}$. Since it holds $\text{Aut}_0(G) = \langle (1 \ 2)(3 \ 4 \ 5 \ 6) \rangle$, we can conclude that a graph shuffle for G is equivalent to a uniform closed shuffle (shuffle, Π, \mathcal{F}). Since it holds $2n + 2m = 32$ and

$|\text{Deg}_G| + 1 = 3$, our result implies that it can be done by 32 cards and three PSSs. By generalizing this idea, for any cyclic group $\Pi = \langle \pi \rangle$, a uniform closed shuffle $(\text{shuffle}, \Pi, \mathcal{F})$ is a graph shuffle for some graph.

4. EFFICIENCY IMPROVEMENTS FOR GRAPH SHUFFLES FOR CYCLES

In this section, we implement efficient graph shuffle protocols for some specific graph classes. In particular, we improve the number of cards in our protocol.

4.1. **The n -cycle graph.** First, we consider the n -cycle graph \vec{C}_n :



The graph shuffle for \vec{C}_n is equivalent to a RC of n cards since the automorphism group $\text{Aut}(\vec{C}_n)$ is isomorphic to the cyclic group of degree n . If we apply our graph shuffle protocol for \vec{C}_n proposed in Section 3, we need $4n$ cards. In this subsection, we propose a graph shuffle protocol for \vec{C}_n with $3n$ cards only.

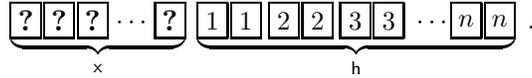
Before describing the improved protocol, we shortly mention how to improve the number of cards. The idea⁶ is to remove the red cards by making a pile of $(x_i, \alpha_i, \alpha_{i+1})$ instead of a pile of $(x_i, \overline{\alpha_i})$ and a pile of (α_i, α_{i+1}) in the previous protocol. Since all vertices of \vec{C}_n have the same degree, all piles of $(x_i, \alpha_i, \alpha_{i+1})$ have the same number of cards and thus the final randomization (corresponding to Step (5) in the previous protocol) can be done by a single PSS.

Let $D_{\text{inp}} = \{x_1, x_2, \dots, x_n\}$ be an arbitrary deck and $D_{\text{help}} = \{1, 1, 2, 2, 3, 3, \dots, n, n\}$ a deck of the symbols of $2n$ helping cards. The sequence of helping cards h is defined as follows:

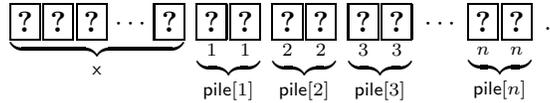
$$h = \boxed{1} \boxed{1} \boxed{2} \boxed{2} \boxed{3} \boxed{3} \cdots \boxed{n} \boxed{n}.$$

For $i = 1, 2, \dots, n$, we set $\text{pile}[i] = \left(\frac{?}{i}, \frac{?}{i} \right)$.

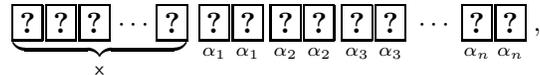
- (1) Place the $3n$ cards as follows:



- (2) Arrange the card-sequence as follows:

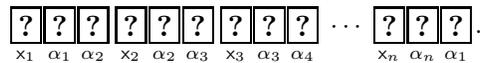


Apply $\text{PSS}_{(n,2)}$ to $(\text{pile}[1], \text{pile}[2], \dots, \text{pile}[n])$ and then we obtain the card-sequence as follows:



where $\{\alpha_1, \alpha_2, \dots, \alpha_n\} = \{1, 2, \dots, n\}$.

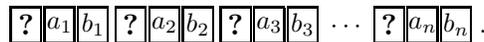
- (3) Arrange the card-sequence as follows:



- (4) Apply $\text{PSS}_{(n,3)}$ to the card-sequence as follows:



- (5) For all piles, turn over the second and third cards. Let $a_i, b_i \in \{1, 2, \dots, n\}$ be the opened symbols of the second and third cards, respectively, in the i -th pile as follows:



⁶We remark that this idea works for every graphs such that all vertices have the same degree.

(6) Arrange n piles so that $(c_1, d_1) = (a_1, b_1)$, $d_i = c_{i+1}$, $(1 \leq i \leq n-1)$, and $d_n = c_1$ as follows:

$$\begin{array}{ccccccc} \boxed{?} & \boxed{c_1} & \boxed{d_1} & \boxed{?} & \boxed{c_2} & \boxed{d_2} & \boxed{?} & \boxed{c_3} & \boxed{d_3} & \cdots & \boxed{?} & \boxed{c_n} & \boxed{d_n} \\ y_1 & & & y_2 & & & y_3 & & & & y_n & & \end{array}.$$

After that, we arrange the card-sequence as follows:

$$\begin{array}{ccccccccccc} \boxed{?} & \boxed{?} & \boxed{?} & \cdots & \boxed{?} & \boxed{1} & \boxed{1} & \boxed{2} & \boxed{2} & \boxed{3} & \boxed{3} & \cdots & \boxed{n} & \boxed{n} \\ y_1 & y_2 & y_3 & & y_n & \underbrace{\hspace{10em}} & & & & & & & & \\ & & & & & & & & & & \text{h} & & & \end{array}.$$

Then the output card-sequence is $y = (y_1, y_2, \dots, y_n)$.

We show the correctness of the protocol. Let $x = (x_1, \dots, x_n)$ be an input sequence. Assume that the protocol outputs the sequence $y = (y_1, \dots, y_n)$ when x is given as input. First, we see that $y = \sigma(x)$ for some σ in the cyclic group of degree n . For $i = 1, \dots, n$, we set $P_i = (x_i, \alpha_i, \alpha_{i+1})$, where $\alpha_{n+1} = \alpha_1$, in Step (3) and put $P = (P_1, P_2, \dots, P_n)$. Let $Q = (Q_1, \dots, Q_n) = (P_{\sigma^{-1}(1)}, \dots, P_{\sigma^{-1}(n)})$ for some $\sigma \in \mathfrak{S}_n$. Then, Q is obtained by σ in the cyclic group of degree n if and only if the third entry of Q_i and the second entry of Q_{i+1} are same for any $1 \leq i \leq n-1$. It follows that the components of obtained sequence in Step (6) are sorted in a cyclic fashion of P . Therefore, y is equal to $\sigma(x)$ for some σ in the cyclic group of degree n . Note that each element σ of the cyclic group is determined by $\sigma^{-1}(1)$, and it is determined by d_1 . For each $k \in \{1, 2, \dots, n\}$, the probability that $k = d_1$ is $\frac{1}{n}$ since d_1 is dependent on the PSS in Step (4) only. Thus the distribution of σ is uniformly random, and hence the protocol is correct.

We show the security of the protocol. Assume that $\sigma \in \mathfrak{S}_n$ and $\tau \in \mathfrak{S}_n$ are chosen in Steps (2) and (4), respectively. Then the first card in the i -th pile in Step (5) is $x_{\tau^{-1}(i)}$. On the other hand, the second and third cards in the i -th pile in Step (5) are $a_i = \tau^{-1}\sigma^{-1}(i)$ and $b_i = \tau^{-1}\sigma^{-1}(i+1)$. Here, we consider $n+1$ as 1. This implies that these opened symbols a_1, \dots, a_n and b_1, \dots, b_n do not allow us to guess the first card of any pile since σ is chosen uniformly at random in Step (4). Therefore, the protocol is secure.

4.2. The undirected n -cycle. Next, we consider the undirected n -cycle graph C_n :

$$C_n = 1 \overset{\text{arc}}{\curvearrowright} 2 \text{ --- } \cdots \text{ --- } n-1 \text{ --- } n.$$

Recall that we regard undirected edge as two directed edges with opposite directions (see the paragraph just before Definition 3.1). The automorphism group $\text{Aut}(C_n)$ is isomorphic to the dihedral group of degree n . For example, the graph shuffle for C_n when $n = 4$ is given as follows:

$$\boxed{1} \boxed{2} \boxed{3} \boxed{4} \mapsto \left\{ \begin{array}{cccc} \boxed{1} & \boxed{2} & \boxed{3} & \boxed{4} \\ \boxed{2} & \boxed{3} & \boxed{4} & \boxed{1} \\ \boxed{3} & \boxed{4} & \boxed{1} & \boxed{2} \\ \boxed{4} & \boxed{1} & \boxed{2} & \boxed{3} \\ \boxed{4} & \boxed{3} & \boxed{2} & \boxed{1} \\ \boxed{3} & \boxed{2} & \boxed{1} & \boxed{4} \\ \boxed{2} & \boxed{1} & \boxed{4} & \boxed{3} \\ \boxed{1} & \boxed{4} & \boxed{3} & \boxed{2} \end{array} \right\},$$

where each sequence is obtained with probability $1/8$. If we apply the graph shuffle for C_n , we need $6n$ cards. In this subsection, we propose a graph shuffle protocol for C_n with $3n$ cards only.

For an undirected n -cyclic graph, even though it has $2n$ edges, the number of cards corresponding to edges is reduced to n pairs of cards using the symmetry of the graph. This improvement is done by the pile-scramble shuffle in Step (3) in the below protocol.

Let $D_{\text{inp}} = \{x_1, x_2, \dots, x_n\}$ be an arbitrary deck and $D_{\text{help}} = \{1, 1, 2, 2, 3, 3, \dots, n, n\}$ a deck of the symbols of $2n$ helping cards. The sequence of helping cards h is defined as follows:

$$h = \boxed{1} \boxed{1} \boxed{2} \boxed{2} \boxed{3} \boxed{3} \cdots \boxed{n} \boxed{n}.$$

For $i = 1, 2, \dots, n$, we set $\text{pile}[i] = \left(\frac{?}{i}, \frac{?}{i} \right)$.

- (1) Place the $3n$ cards as follows:

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \cdots \boxed{?}}_x \underbrace{\boxed{1} \boxed{1} \boxed{2} \boxed{2} \boxed{3} \boxed{3} \cdots \boxed{n} \boxed{n}}_h .$$

- (2) Arrange the card-sequence as follows:

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \cdots \boxed{?}}_x \underbrace{\boxed{?} \boxed{?}}_{\substack{1 \\ \text{pile}[1]}} \underbrace{\boxed{?} \boxed{?}}_{\substack{2 \\ \text{pile}[2]}} \underbrace{\boxed{?} \boxed{?}}_{\substack{3 \\ \text{pile}[3]}} \cdots \underbrace{\boxed{?} \boxed{?}}_{\substack{n \\ \text{pile}[n]}} .$$

Apply $\text{PSS}_{(n,2)}$ to $(\text{pile}[1], \text{pile}[2], \dots, \text{pile}[n])$ and then we obtain the card-sequence as follows:

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \cdots \boxed{?}}_x \underbrace{\boxed{?} \boxed{?}}_{\alpha_1 \alpha_1} \underbrace{\boxed{?} \boxed{?}}_{\alpha_2 \alpha_2} \underbrace{\boxed{?} \boxed{?}}_{\alpha_3 \alpha_3} \cdots \underbrace{\boxed{?} \boxed{?}}_{\alpha_n \alpha_n} ,$$

where $\{\alpha_1, \alpha_2, \dots, \alpha_n\} = \{1, 2, \dots, n\}$.

- (3) Arrange the card-sequence as follows:

$$\underbrace{\boxed{?} \boxed{?} \boxed{?} \cdots \boxed{?}}_x \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{\alpha_1 \alpha_2 \alpha_3} \cdots \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{\alpha_{n-1} \alpha_n \alpha_2 \alpha_3 \alpha_4} \cdots \underbrace{\boxed{?} \boxed{?}}_{\alpha_n \alpha_1} .$$

Apply $\text{PSS}_{(2,n)}$ to the rightmost card-sequence of $2n$ cards as follows:

$$\left| \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{\alpha_1 \alpha_2 \alpha_3} \cdots \underbrace{\boxed{?} \boxed{?}}_{\alpha_{n-1} \alpha_n} \right| \left| \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{\alpha_2 \alpha_3 \alpha_4} \cdots \underbrace{\boxed{?} \boxed{?}}_{\alpha_n \alpha_1} \right| .$$

Then we obtain the following card-sequence:

$$\underbrace{\boxed{?} \boxed{?}}_x \cdots \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{\beta_1 \beta_2 \beta_3} \cdots \underbrace{\boxed{?} \boxed{?} \boxed{?} \boxed{?}}_{\beta_n \gamma_1 \gamma_2 \gamma_3} \cdots \underbrace{\boxed{?}}_{\gamma_n} ,$$

where $\{(\alpha_1, \alpha_2, \dots, \alpha_n), (\alpha_2, \dots, \alpha_n, \alpha_1)\} = \{(\beta_1, \beta_2, \dots, \beta_n), (\gamma_1, \gamma_2, \dots, \gamma_n)\}$.

- (4) Arrange the card-sequence as follows:

$$\underbrace{\boxed{?} \boxed{?} \boxed{?}}_{x_1 \beta_1 \gamma_1} \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{x_2 \beta_2 \gamma_2} \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{x_3 \beta_3 \gamma_3} \cdots \underbrace{\boxed{?} \boxed{?} \boxed{?}}_{x_n \beta_n \gamma_n} .$$

- (5) Apply $\text{PSS}_{(n,3)}$ to the card-sequence as follows:

$$\left| \boxed{?} \boxed{?} \boxed{?} \right| \left| \boxed{?} \boxed{?} \boxed{?} \right| \left| \boxed{?} \boxed{?} \boxed{?} \right| \cdots \left| \boxed{?} \boxed{?} \boxed{?} \right| .$$

- (6) For all piles, turn over the second and third cards. Let $a_i, b_i \in \{1, 2, \dots, n\}$ be the opened symbols of the second and third cards, respectively, in the i -th pile as follows:

$$\boxed{?} \boxed{a_1} \boxed{b_1} \boxed{?} \boxed{a_2} \boxed{b_2} \boxed{?} \boxed{a_3} \boxed{b_3} \cdots \boxed{?} \boxed{a_n} \boxed{b_n} .$$

- (7) Arrange n piles so that $(c_1, d_1) = (a_1, b_1)$, $d_i = c_{i+1}$, $(1 \leq i \leq n-1)$, and $d_n = c_1$ as follows:

$$\underbrace{\boxed{?} \boxed{c_1} \boxed{d_1}}_{y_1} \underbrace{\boxed{?} \boxed{c_2} \boxed{d_2}}_{y_2} \underbrace{\boxed{?} \boxed{c_3} \boxed{d_3}}_{y_3} \cdots \underbrace{\boxed{?} \boxed{c_n} \boxed{d_n}}_{y_n} .$$

Then arrange the card-sequence as follows:

$$\underbrace{\boxed{?} \boxed{?} \boxed{?}}_{y_1 \ y_2 \ y_3} \cdots \underbrace{\boxed{?}}_{y_n} \underbrace{\boxed{1} \boxed{1} \boxed{2} \boxed{2} \boxed{3} \boxed{3} \cdots \boxed{n} \boxed{n}}_h .$$

Then the output card-sequence is (y_1, y_2, \dots, y_n) .

We first show the correctness of the protocol. Let $x = (x_1, \dots, x_n)$ be an input sequence. Assume that the protocol outputs the sequence $y = (y_1, \dots, y_n)$ when x is given as input. Observe that if we apply a graph shuffle for C_n to x , the output sequence is one of the following sequences

$$(x_k, x_{k+1}, \dots, x_n, x_1, x_2, \dots, x_{k-1}), \quad (x_k, x_{k-1}, \dots, x_1, x_n, x_{n-1}, \dots, x_{k+1})$$

for some $k \in \{1, 2, \dots, n\}$. We denote by $\text{Cyc}(k)$ and $\text{Rev}(k)$ the former sequence and the latter sequence, respectively. To show the correctness of the protocol, we see that y is one of $\text{Cyc}(k)$ and $\text{Rev}(k)$ for some $k = 1, \dots, n$. For $i = 1, \dots, n$, we set $P_i = (x_i, \beta_i, \gamma_i)$ and put $P = (P_1, \dots, P_n)$. Suppose that $(\alpha_1, \dots, \alpha_n)$ is

equal to $(\beta_1, \dots, \beta_n)$ in Step (3). In this case, it holds $\gamma_i = \beta_{i+1}$ for any $i \in \{1, 2, \dots, n\}$, where $\beta_{n+1} = \beta_1$. It follows from the above equations and the argument in the proof of the correctness of the protocol in Subsection 4.1 that $\mathbf{y} = \text{Cyc}(k)$ for some k . Similarly, if $(\alpha_1, \dots, \alpha_n) = (\gamma_1, \dots, \gamma_n)$ in Step (3), the equations $\gamma_i = \beta_{n-i+1}$ hold for any $i \in \{1, 2, \dots, n\}$. This implies that $\mathbf{y} = \text{Rev}(k)$ for some k .

Next, we show that the distribution of \mathbf{y} is uniform. Assume that $n = 2$. We note that $\text{Cyc}(1) = \text{Rev}(1)$ and $\text{Cyc}(2) = \text{Rev}(2)$. Then the candidates appearing as a result of Step (4) are:

$$\begin{array}{cccccc} \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ x_1 & 1 & 2 & x_2 & 2 & 1 \end{array}, \quad \begin{array}{cccccc} \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} & \boxed{?} \\ x_1 & 2 & 1 & x_2 & 1 & 2 \end{array},$$

and these each have a probability of $\frac{1}{2}$. Thus, the probabilities that $\mathbf{y} = (x_1, x_2)$ and $\mathbf{y} = (x_2, x_1)$ are same. Now, we assume that $n \geq 3$. In this case, for any $k = 1, \dots, n$, all sequences $\text{Cyc}(k)$ and $\text{Rev}(k)$ are distinct. In order to get $\mathbf{y} = \text{Cyc}(k)$, it requires that $(\alpha_1, \dots, \alpha_n) = (\beta_1, \dots, \beta_n)$ in Step (3) and $\sigma^{-1}(1) = k$, where σ is the chosen permutation in Step (5). Hence, the probability that $\mathbf{y} = \text{Cyc}(k)$ is $\frac{1}{2k}$. Similarly, the probability that $\mathbf{y} = \text{Rev}(k)$ is also $\frac{1}{2k}$. This shows that the protocol is correct.

We show the correctness of the protocol. Assume that $\sigma \in \mathfrak{S}_n$ and $\tau \in \mathfrak{S}_n$ are chosen in Step (2) and Step (5), respectively. Then the first card in the i -th pile in Step (6) is $x_{\tau^{-1}(i)}$. On the other hand, the second and third cards in the i -th pile in Step (5) are depending on the result of Step (3), and they are determined as follows. If $(\alpha_1, \dots, \alpha_n) = (\beta_1, \dots, \beta_n)$, then $a_i = \tau^{-1}\sigma^{-1}(i)$ and $b_i = \tau^{-1}\sigma^{-1}(i+1)$, otherwise, $a_i = \tau^{-1}\sigma^{-1}(i+1)$ and $b_i = \tau^{-1}\sigma^{-1}(i)$. Here, we consider $n+1$ as 1. In either case, these open symbols a_1, \dots, a_n and b_1, \dots, b_n do not allow us to guess the first card of any pile since σ is chosen uniformly at random in Step (5). Therefore, the protocol is secure.

5. CONCLUSIONS AND FUTURE WORKS

In this paper, we show that any graph shuffle can be done by PSSs. In particular, we need $2(n+m)$ cards and $|\text{Deg}_G| + 1$ PSSs, where n and m are the numbers of vertices and arrows of G , respectively. We left as open problems (1) to remove the computation of an isomorphism between two isomorphic graphs in a graph shuffle protocol keeping everything efficient and (2) to find another interesting applications for our graph shuffle protocol. We hope that this research direction (i.e., constructing a nontrivial shuffle from the standard shuffles such as RCs, RBCs, and PSSs) will attract the interest of researchers on card-based cryptography and new shuffle protocols will be proposed in future work.

REFERENCES

- [1] Y. Abe, Y. Hayashi, T. Mizuki, and H. Sone. Five-card AND protocol in committed format using only practical shuffles. In K. Emura, J. H. Seo, and Y. Watanabe, editors, *Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop, APKC@AsiaCCS, Incheon, Republic of Korea, June 4, 2018*, pages 3–8. ACM, 2018.
- [2] Y. Abe, Y. Hayashi, T. Mizuki, and H. Sone. Five-card AND computations in committed format using only uniform cyclic shuffles. *New Gener. Comput.*, 39(1):97–114, 2021.
- [3] X. Bultel, J. Dreier, J. Dumas, and P. Lafourcade. Physical zero-knowledge proofs for akari, takuzu, kakuro and kenken. In E. D. Demaine and F. Grandoni, editors, *8th International Conference on Fun with Algorithms, FUN 2016, June 8-10, 2016, La Maddalena, Italy*, volume 49 of *LIPICs*, pages 8:1–8:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.
- [4] X. Bultel, J. Dreier, J. Dumas, P. Lafourcade, D. Miyahara, T. Mizuki, A. Nagao, T. Sasaki, K. Shinagawa, and H. Sone. Physical zero-knowledge proof for makaro. In T. Izumi and P. Kuznetsov, editors, *Stabilization, Safety, and Security of Distributed Systems - 20th International Symposium, SSS 2018, Tokyo, Japan, November 4-7, 2018, Proceedings*, volume 11201 of *Lecture Notes in Computer Science*, pages 111–125. Springer, 2018.
- [5] E. Cheung, C. Hawthorne, and P. Lee. Cs 758 project: Secure computation with playing cards, 2013. https://cdchawthorne.com/writings/secure_playing_cards.pdf.
- [6] C. Crépeau and J. Kilian. Discreet solitary games. In D. R. Stinson, editor, *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 319–330. Springer, 1993.
- [7] B. den Boer. More efficient match-making and satisfiability: *The Five Card Trick*. In J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings*, volume 434 of *Lecture Notes in Computer Science*, pages 208–217. Springer, 1989.

- [8] J. Dumas, P. Lafourcade, D. Miyahara, T. Mizuki, T. Sasaki, and H. Sone. Interactive physical zero-knowledge proof for norinori. In D. Du, Z. Duan, and C. Tian, editors, *Computing and Combinatorics - 25th International Conference, COCOON 2019, Xi'an, China, July 29-31, 2019, Proceedings*, volume 11653 of *Lecture Notes in Computer Science*, pages 166–177. Springer, 2019.
- [9] L. L. G. Chartrand and P. Zhang. *Graphs & Digraphs (six edition)*. CRC Press, 2015.
- [10] R. Gradwohl, M. Naor, B. Pinkas, and G. N. Rothblum. Cryptographic and physical zero-knowledge proof systems for solutions of sudoku puzzles. In *Fun with Algorithms, 4th International Conference, FUN 2007, Castiglioncello, Italy, June 3-5, 2007, Proceedings*, pages 166–182, 2007.
- [11] Y. Hashimoto, K. Shinagawa, K. Nuida, M. Inamura, and G. Hanaoka. Secure grouping protocol using a deck of cards. In J. Shikata, editor, *Information Theoretic Security - 10th International Conference, ICITS 2017, Hong Kong, China, November 29 - December 2, 2017, Proceedings*, volume 10681 of *Lecture Notes in Computer Science*, pages 135–152. Springer, 2017.
- [12] J. Heather, S. Schneider, and V. Teague. Cryptographic protocols with everyday objects. *Formal Asp. Comput.*, 26(1):37–62, 2014.
- [13] 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, editors, *Unconventional Computation and Natural Computation - 14th International Conference, UCNC 2015, Auckland, New Zealand, August 30 - September 3, 2015, Proceedings*, volume 9252 of *Lecture Notes in Computer Science*, pages 215–226. Springer, 2015.
- [14] A. Koch and S. Walzer. Private function evaluation with cards. *IACR Cryptology ePrint Archive*, 2018:1113, 2018.
- [15] A. Koch and S. Walzer. Foundations for actively secure card-based cryptography. In M. Farach-Colton, G. Prencipe, and R. Uehara, editors, *10th International Conference on Fun with Algorithms, FUN 2021, May 30 to June 1, 2021, Favignana Island, Sicily, Italy*, volume 157 of *LIPICs*, pages 17:1–17:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.
- [16] H. Koyama, D. Miyahara, T. Mizuki, and H. Sone. A secure three-input AND protocol with a standard deck of minimal cards. In R. Santhanam and D. Musatov, editors, *Computer Science - Theory and Applications - 16th International Computer Science Symposium in Russia, CSR 2021, Sochi, Russia, June 28 - July 2, 2021, Proceedings*, volume 12730 of *Lecture Notes in Computer Science*, pages 242–256. Springer, 2021.
- [17] H. Koyama, K. Toyoda, D. Miyahara, and T. Mizuki. New card-based copy protocols using only random cuts. In K. Emura and Y. Wang, editors, *Proceedings of the 8th on ASIA Public-Key Cryptography Workshop, APKC@AsiaCCS 2021, Virtual Event Hong Kong, 7 June, 2021*, pages 13–22. ACM, 2021.
- [18] P. Lafourcade, D. Miyahara, T. Mizuki, T. Sasaki, and H. Sone. A physical ZKP for slitherlink: How to perform physical topology-preserving computation. In S. Heng and J. López, editors, *Information Security Practice and Experience - 15th International Conference, ISPEC 2019, Kuala Lumpur, Malaysia, November 26-28, 2019, Proceedings*, volume 11879 of *Lecture Notes in Computer Science*, pages 135–151. Springer, 2019.
- [19] Y. Lindell. Secure multiparty computation (mpc). *Cryptology ePrint Archive*, Report 2020/300, 2020. <https://ia.cr/2020/300>.
- [20] A. Marcedone, Z. Wen, and E. Shi. Secure dating with four or fewer cards. *Cryptology ePrint Archive*, Report 2015/1031, 2015.
- [21] B. McKay and A. Piperno. The nauty traces page.
- [22] D. Miyahara, Y. Hayashi, T. Mizuki, and H. Sone. Practical and easy-to-understand card-based implementation of yao’s millionaire protocol. In D. Kim, R. N. Uma, and A. Zelikovsky, editors, *Combinatorial Optimization and Applications - 12th International Conference, COCOA 2018, Atlanta, GA, USA, December 15-17, 2018, Proceedings*, volume 11346 of *Lecture Notes in Computer Science*, pages 246–261. Springer, 2018.
- [23] D. Miyahara, Y. Hayashi, T. Mizuki, and H. Sone. Practical card-based implementations of yao’s millionaire protocol. *Theor. Comput. Sci.*, 803:207–221, 2020.
- [24] 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 M. Farach-Colton, G. Prencipe, and R. Uehara, editors, *10th International Conference on Fun with Algorithms, FUN 2021, May 30 to June 1, 2021, Favignana Island, Sicily, Italy*, volume 157 of *LIPICs*, pages 20:1–20:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.
- [25] T. Mizuki. Applications of card-based cryptography to education. *IEICE Technical Report*, 116(289):13–17, 2016. (In Japanese).
- [26] T. Mizuki. Card-based protocols for securely computing the conjunction of multiple variables. *Theor. Comput. Sci.*, 622:34–44, 2016.
- [27] T. Mizuki. Efficient and secure multiparty computations using a standard deck of playing cards. In S. Foresti and G. Persiano, editors, *Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings*, volume 10052 of *Lecture Notes in Computer Science*, pages 484–499, 2016.
- [28] T. Mizuki, I. K. Asiedu, and H. Sone. Voting with a logarithmic number of cards. In G. Mauri, A. Denuzio, L. Manzoni, and A. E. Porreca, editors, *Unconventional Computation and Natural Computation - 12th International Conference, UCNC 2013, Milan, Italy, July 1-5, 2013, Proceedings*, volume 7956 of *Lecture Notes in Computer Science*, pages 162–173. Springer, 2013.
- [29] T. Mizuki, M. Kumamoto, and H. Sone. The five-card trick can be done with four cards. In X. Wang and K. Sako, editors, *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012, Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 598–606. Springer, 2012.

- [30] T. Mizuki and H. Shizuya. A formalization of card-based cryptographic protocols via abstract machine. *Int. J. Inf. Sec.*, 13(1):15–23, 2014.
- [31] T. Mizuki and H. Sone. Six-card secure AND and four-card secure XOR. In X. Deng, J. E. Hopcroft, and J. Xue, editors, *Frontiers in Algorithmics, Third International Workshop, FAW 2009, Hefei, China, June 20-23, 2009. Proceedings*, volume 5598 of *Lecture Notes in Computer Science*, pages 358–369. Springer, 2009.
- [32] T. Mizuki, F. Uchiike, and H. Sone. Securely computing XOR with 10 cards. *The Australasian Journal of Combinatorics*, 36:279–293, 2006.
- [33] S. Murata, D. Miyahara, T. Mizuki, and H. Sone. Efficient generation of a card-based uniformly distributed random derangement. In R. Uehara, S. Hong, and S. C. Nandy, editors, *WALCOM: Algorithms and Computation - 15th International Conference and Workshops, WALCOM 2021, Yangon, Myanmar, February 28 - March 2, 2021, Proceedings*, volume 12635 of *Lecture Notes in Computer Science*, pages 78–89. Springer, 2021.
- [34] V. Niemi and A. Renvall. Secure multiparty computations without computers. *Theor. Comput. Sci.*, 191(1-2):173–183, 1998.
- [35] V. Niemi and A. Renvall. Solitaire zero-knowledge. *Fundam. Informaticae*, 38(1-2):181–188, 1999.
- [36] T. Nishida, Y. Hayashi, T. Mizuki, and H. Sone. Card-based protocols for any boolean function. In R. Jain, S. Jain, and F. Stephan, editors, *Theory and Applications of Models of Computation - 12th Annual Conference, TAMC 2015, Singapore, May 18-20, 2015, Proceedings*, volume 9076 of *Lecture Notes in Computer Science*, pages 110–121. Springer, 2015.
- [37] T. Nishida, Y. Hayashi, T. Mizuki, and H. Sone. Securely computing three-input functions with eight cards. *IEICE Transactions*, 98-A(6):1145–1152, 2015.
- [38] T. Nishida, T. Mizuki, and H. Sone. Securely computing the three-input majority function with eight cards. In A. Dediu, C. Martín-Vide, B. Truthe, and M. A. Vega-Rodríguez, editors, *Theory and Practice of Natural Computing - Second International Conference, TPNC 2013, Cáceres, Spain, December 3-5, 2013, Proceedings*, volume 8273 of *Lecture Notes in Computer Science*, pages 193–204. Springer, 2013.
- [39] L. Robert, D. Miyahara, P. Lafourcade, and T. Mizuki. Physical zero-knowledge proof for suguru puzzle. In S. Devismes and N. Mittal, editors, *Stabilization, Safety, and Security of Distributed Systems - 22nd International Symposium, SSS 2020, Austin, TX, USA, November 18-21, 2020, Proceedings*, volume 12514 of *Lecture Notes in Computer Science*, pages 235–247. Springer, 2020.
- [40] L. Robert, D. Miyahara, P. Lafourcade, and T. Mizuki. Interactive physical ZKP for connectivity: Applications to nurikabe and hitori. In L. D. Mol, A. Weiermann, F. Manea, and D. Fernández-Duque, editors, *Connecting with Computability - 17th Conference on Computability in Europe, CiE 2021, Virtual Event, Ghent, July 5-9, 2021, Proceedings*, volume 12813 of *Lecture Notes in Computer Science*, pages 373–384. Springer, 2021.
- [41] S. Ruangwises and T. Itoh. Physical zero-knowledge proof for numberlink puzzle and k vertex-disjoint paths problem. *New Gener. Comput.*, 39(1):3–17, 2021.
- [42] S. Ruangwises and T. Itoh. Physical zero-knowledge proof for ripple effect. *Theor. Comput. Sci.*, 895:115–123, 2021.
- [43] S. Ruangwises and T. Itoh. Securely computing the n -variable equality function with $2n$ cards. *Theor. Comput. Sci.*, 887:99–110, 2021.
- [44] T. Saito, D. Miyahara, Y. Abe, T. Mizuki, and H. Shizuya. How to implement a non-uniform or non-closed shuffle. In C. Martín-Vide, M. A. Vega-Rodríguez, and M. Yang, editors, *Theory and Practice of Natural Computing - 9th International Conference, TPNC 2020, Taoyuan, Taiwan, December 7-9, 2020, Proceedings*, volume 12494 of *Lecture Notes in Computer Science*, pages 107–118. Springer, 2020.
- [45] T. Sasaki, D. Miyahara, T. Mizuki, and H. Sone. Efficient card-based zero-knowledge proof for sudoku. *Theor. Comput. Sci.*, 839:135–142, 2020.
- [46] T. Sasaki, T. Mizuki, and H. Sone. Card-based zero-knowledge proof for sudoku. In H. Ito, S. Leonardi, L. Pagli, and G. Prencipe, editors, *9th International Conference on Fun with Algorithms, FUN 2018, June 13-15, 2018, La Maddalena, Italy*, volume 100 of *LIPICs*, pages 29:1–29:10. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.
- [47] K. Shinagawa and T. Mizuki. The six-card trick: Secure computation of three-input equality. In K. Lee, editor, *Information Security and Cryptology - ICISC 2018 - 21st International Conference, Seoul, South Korea, November 28-30, 2018, Revised Selected Papers*, volume 11396 of *Lecture Notes in Computer Science*, pages 123–131. Springer, 2018.
- [48] K. Shinagawa and T. Mizuki. Secure computation of any boolean function based on any deck of cards. In Y. Chen, X. Deng, and M. Lu, editors, *Frontiers in Algorithmics - 13th International Workshop, FAW 2019, Sanya, China, April 29 - May 3, 2019, Proceedings*, volume 11458 of *Lecture Notes in Computer Science*, pages 63–75. Springer, 2019.
- [49] K. Shinagawa and K. Nuida. A single shuffle is enough for secure card-based computation of any boolean circuit. *Discret. Appl. Math.*, 289:248–261, 2021.
- [50] Y. Shinoda, D. Miyahara, K. Shinagawa, T. Mizuki, and H. Sone. Card-based covert lottery. In D. Maimut, A. Oprina, and D. Sauveron, editors, *Innovative Security Solutions for Information Technology and Communications - 13th International Conference, SecITC 2020, Bucharest, Romania, November 19-20, 2020, Revised Selected Papers*, volume 12596 of *Lecture Notes in Computer Science*, pages 257–270. Springer, 2020.
- [51] A. Stiglic. Computations with a deck of cards. *Theor. Comput. Sci.*, 259(1-2):671–678, 2001.
- [52] K. Takashima, Y. Abe, T. Sasaki, D. Miyahara, K. Shinagawa, T. Mizuki, and H. Sone. Card-based secure ranking computations. In Y. Li, M. Cardei, and Y. Huang, editors, *Combinatorial Optimization and Applications - 13th International Conference, COCOA 2019, Xiamen, China, December 13-15, 2019, Proceedings*, volume 11949 of *Lecture Notes in Computer Science*, pages 461–472. Springer, 2019.

- [53] K. Takashima, Y. Abe, T. Sasaki, D. Miyahara, K. Shinagawa, T. Mizuki, and H. Sone. Card-based protocols for secure ranking computations. *Theor. Comput. Sci.*, 845:122–135, 2020.
- [54] K. Takashima, D. Miyahara, T. Mizuki, and H. Sone. Card-based protocol against actively revealing card attack. In C. Martín-Vide, G. T. Pond, and M. A. Vega-Rodríguez, editors, *Theory and Practice of Natural Computing - 8th International Conference, TPNC 2019, Kingston, ON, Canada, December 9-11, 2019, Proceedings*, volume 11934 of *Lecture Notes in Computer Science*, pages 95–106. Springer, 2019.
- [55] K. Toyoda, D. Miyahara, T. Mizuki, and H. Sone. Six-card finite-runtime XOR protocol with only random cut. In K. Emura and N. Yanai, editors, *Proceedings of the 7th on ASIA Public-Key Cryptography Workshop, APKC@AsiaCCS 2020, Taipei, Taiwan, October 6, 2020*, pages 2–8. ACM, 2020.
- [56] A. C. Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 160–164. IEEE Computer Society, 1982.
- [57] A. C. Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*, pages 162–167. IEEE Computer Society, 1986.

DECLARATIONS

Funding. K. Miyamoto was partly supported by JSPS KAKENHI 20K14302. K. Shinagawa was partly supported by JSPS KAKENHI 21K17702.

(K. Miyamoto) IBARAKI UNIVERSITY, 4-12-1 NAKANARUSAWA, HITACHI, IBARAKI, 316-8511, JAPAN.
Email address: `kengo.miyamoto.uz63@vc.ibaraki.ac.jp`

(K. Shinagawa) IBARAKI UNIVERSITY, 4-12-1 NAKANARUSAWA, HITACHI, IBARAKI, 316-8511, JAPAN; NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST), TOKYO WATERFRONT BIO-IT RESEARCH BUILDING 2-4-7 AOMI, KOTO-KU, TOKYO, 135-0064, JAPAN.
Email address: `shinagawakazumasa@gmail.com`