How to Implement a Non-uniform or Non-closed Shuffle*

Takahiro Saito[1], Daiki Miyahara[1,3], Yuta Abe[1], Takaaki Mizuki[2], and Hiroki Shizuya[1]

[1] Graduate School of Information Sciences, Tohoku University
6–3–09 Aramaki-Aza-Aoba, Aoba, Sendai 980–8579, Japan
saito.takahiro.q7@dc.tohoku.ac.jp
[2] Cyberscience Center, Tohoku University
6–3 Aramaki-Aza-Aoba, Aoba, Sendai 980–8578, Japan
[3] National Institute of Advanced Industrial Science and Technology (AIST),
2–4–7 Aomi, Koto-ku, Tokyo 135–0064, Japan

Abstract. Card-based protocols allow to perform secure multiparty computations using a deck of physical cards, and rely on shuffle actions such as the (normal) shuffle, the random cut, and the random bisection cut. A shuffle action is mathematically defined by a pair of a permutation set (which is a subset of the symmetric group) and a probability distribution on it; while one can theoretically consider any shuffle action in mind, he or she may be unable to determine whether it can be easily implemented by human hands. As one of the most general results, Koch and Walzer showed that any uniform closed shuffle (meaning that its permutation set is a subgroup and its distribution is uniform) can be implemented by human hands with the help of additional cards. However, there are several existing protocols which use non-uniform and/or non-closed shuffles. To implement these specific shuffles, Nishimura et al. proposed an idea of using (special) physical cases that can store piles of cards as well as Koch and Walzer proposed an implementation of a specific non-closed shuffle with additional cards. Because their implementations handle a limited class of non-uniform and/or non-closed shuffles, it is still open to find a general method for implementing any shuffle. In this paper, we solve the above problem; we implement "any" shuffle with only additional cards, provided that every probability of its distribution is a rational number. Therefore, our implementation works for any non-closed or non-uniform shuffle (if the distribution is rational as above).

Keywords: Cryptography · Card-based protocols · Implementation of Shuffle · Unconventional computation

1 Introduction

In 1989, den Boer [1] proposed the first *card-based protocol* called the "five-card trick." This protocol performs a secure computation of the logical AND using

* This paper appears in Proceedings of TPNC 2020. The final authenticated version is available online at https://doi.org/10.1007/978-3-030-63000-3_9.

two black cards ♣♣ and three red cards ♡♡♡, whose backsides have the same pattern ?. This paper begins by introducing the five-card trick.

1.1 Five-card Trick

Assume that Alice has a private bit $a \in \{0,1\}$ and Bob has a private bit $b \in \{0,1\}$; the five-card trick takes a and b as inputs and outputs the value of $a \wedge b$ without revealing any information about a and b more than necessary, as follows.

1. Alice puts two face-down cards commiting to her bit a according to the encoding: $\clubsuit\heartsuit = 0, \heartsuit\clubsuit = 1$. Bob puts two face-down cards similarly, and they put an additional red card in the middle. Then, swap the positions of Alice's two cards so that we have a negated value $\overline{a}$ as well as turn over the middle red card:
$$\underbrace{??}_{a}\ \heartsuit\ \underbrace{??}_{b} \to \underbrace{??}_{\overline{a}}\ ?\ \underbrace{??}_{b}.$$
2. Apply a *random cut* (denoted by $\langle\cdot\rangle$) to the sequence of five cards:
$$\langle\ ?????\ \rangle \to ?????.$$
A random cut (RC), meaning a cyclic shuffling operation, uniformly and randomly shifts the positions of a sequence without changing its order. Mathematically, it uniformly chooses one permutation from the permutation set
$$\{\mathsf{id}, (1\ 2\ 3\ 4\ 5), (1\ 2\ 3\ 4\ 5)^2, (1\ 2\ 3\ 4\ 5)^3, (1\ 2\ 3\ 4\ 5)^4\}, \tag{1}$$
and the chosen permutation is applied to the sequence of five cards (but nobody knows which permutation is applied), where id denotes the identity permutation and $(i_1\ i_2 \cdots i_\ell)$ represents a cyclic permutation.
3. Reveal the five cards. If the resulting sequence has three consecutive red cards ♡♡♡ (apart from cyclic rotation), then $a \wedge b = 1$. Otherwise, $a \wedge b = 0$.

To implement an RC, Alice (or Bob) quickly repeats cutting a sequence of cards until nobody (even including Alice) can trace the offsets. Note that this implementation was experimentally shown to be secure by Ueda et al. [22].

1.2 Unconventional Shuffles

As seen above, the five-card trick [1] elegantly performs a secure computation of AND by using an RC, which is implementable. On the other hand, this paper deals with unconventional shuffles: To understand them, let us consider the following permutation set, which is similar to the permutation set (1) that represents an RC used in the five-card trick:

$$\{\mathsf{id}, (1\ 2\ 3\ 4\ 5), (1\ 2\ 3\ 4\ 5)^2, (1\ 2\ 3\ 4\ 5)^3\}; \tag{2}$$

namely, $(1\ 2\ 3\ 4\ 5)^4$ is excluded from (1). Compared to implementing an RC, it seems difficult for human hands to implement such a shuffle action of uniformly

choosing one permutation from the above set and applying that permutation. This is because the permutation set (2) cannot be generated by a single element, i.e., it is not a cyclic group.

As another example, let us consider a shuffle action that divides a sequence of four cards into two halves and randomly swaps them. Mathematically, it uniformly and randomly chooses one permutation from the permutation set

$$\{\mathsf{id}, (1\ 3)(2\ 4)\},$$

and applies that permutation to the sequence. This kind of a shuffle is called a *random bisection cut* (RBC) and is used in several card-based protocols (e.g., [12,13,20]). Ueda et al. [21,22] showed the secure implementation of an RBC called the "spinning throw" as well as a more secure way using a polystyrene ball. Now, let us consider a "modified RBC" with a non-uniform probability distribution. For example, how could we implement a shuffle where id is chosen with a probability of 1/3 and $(1\ 3)(2\ 4)$ is chosen with a probability of 2/3? Surprisingly, such a non-uniform RBC is required for executing the card-based AND protocols with the minimal number of cards [7]. Implementing such a non-uniform shuffle seems difficult by only human hands.

Hereinafter, assume that, for a sequence of m (≥ 2) face-down cards, we want to apply a shuffle action. Formally, a *shuffle* is defined as a set of permutations $\Pi \subseteq S_m$ along with a probability distribution $\mathcal{F}$ on Π [10,11], where S_m denotes the symmetric group of degree m: We use the notation $(\mathsf{shuf}, \Pi, \mathcal{F})$ to mean that a permutation $\pi \in \Pi$ is drawn according to the distribution $\mathcal{F}$ and π is applied to the sequence of m cards.

There are three important categories of a shuffle.

Uniform shuffle. A shuffle $(\mathsf{shuf}, \Pi, \mathcal{F})$ is said to be *uniform* if the distribution $\mathcal{F}$ is uniform. That is, any $\pi \in \Pi$ is drawn with the equal probability. We sometimes write it as (shuf, Π) if $\mathcal{F}$ is uniform and $(\mathsf{shuf}, \mathcal{F})$ if $\mathcal{F}$ is non-uniform for simplicity.

Cyclic shuffle. A shuffle $(\mathsf{shuf}, \Pi, \mathcal{F})$ is said to be *cyclic* if the permutation set Π is a cyclic group, i.e., there exists a permutation π in Π such that $\Pi = \langle \pi \rangle = \{\pi^i \mid i \text{ is an integer}\}$.

Closed shuffle. A shuffle $(\mathsf{shuf}, \Pi, \mathcal{F})$ is said to be *closed* if Π is a subgroup of the symmetric group.

As shown in the above examples, one can consider any shuffle in mind although it is not so easy to determine whether such a (possibly, complex) shuffle is implementable by humans. One of the most general results is as follows: Koch and Walzer [6] in 2017 showed that any uniform closed shuffles can be implemented with additional cards whose back pattern is different from the pattern of cards to be shuffled.

For implementing a non-uniform and/or non-closed shuffle, there are two existing works, each of which proposed implementations of specific non-uniform and/or non-closed shuffles, as follows. Nishimura et al. [14] in 2018 proposed implementations of two kinds of shuffles:

$$(\mathsf{shuf}, \{\mathsf{id}, \pi\}, \mathcal{F}_1) \text{ and } (\mathsf{shuf}, \Pi^{\langle s_1, s_2, \cdots, s_k \rangle}, \mathcal{F}_2),$$

using (special) physical card cases that can store piles of cards. Here, π is any permutation in the symmetric group and $\Pi^{(s_1,s_2,\cdots,s_k)}$ is the permutation set obtained by dividing a sequence of cards into k piles, where the i-th pile consists of s_i cards, and arbitrarily shifting them. $\mathcal{F}_1$ and $\mathcal{F}_2$ are any distributions where each probability is a rational number. In 2017, Koch and Walzer [6] showed an implementation of the uniform non-closed shuffle

$$(\mathsf{shuf}, \{\mathsf{id}, (1\ 2\ 3\ 4\ 5)^3\})$$

using additional cards (and they mentioned that its formalization is possible). To summarize, their proposed implementations focus on a limited class of non-uniform and/or non-closed shuffles: How to implement any shuffle including non-uniform or non-closed shuffles remains an open problem.

1.3 Contribution

In this paper, we solve the above problem. That is, we propose implementations of any shuffle

$$(\mathsf{shuf}, \{\pi_1, \pi_2, \ldots, \pi_k\}, \pi_1 \mapsto p_1, \pi_2 \mapsto p_2, \ldots, \pi_k \mapsto p_k),$$

for a positive integer $k\,(\geq 2)$, where π_i is a permutation in S_m and p_i is a rational number for every $i, 1 \leq i \leq k$. (Note that $\sum_{i=1}^{k} p_i = 1$.) Therefore, we can implement any shuffle even if it is non-uniform or non-closed (provided that the distribution is rational as above). Our proposed implementations require only additional cards (called "padding cards" in this paper), and hence, our result shows that the above problem can be solved without relying on any additional tool, such as special physical cases as proposed by Nishimura et al. [14].

Specifically, we propose two implementations. In Section 3, we show the first one that is specialized for $k = 2$: We implement a non-uniform RBC with padding cards and combine our implementation and the idea by Koch and Walzer [6]. In Section 4, we show the second one that is for any k. It is based on another idea where we regard a permutation as a sequence of number cards.

One might think that this paper considered a narrow problem and our proposed methods would have less applications. We note that our proposed techniques can be applied to the problem of uniformly generating a hidden random permutation without fixed points (also known as Secret Santa) [2] although we omit the details due to the page limit length.

1.4 Related work

The mainstream in the field of card-based protocols was to use a uniform and cyclic shuffle. In 2015, Koch et al. [7] first studied non-uniform and/or non-closed shuffles and proposed card-minimal AND protocols using such unconventional shuffles. In 2015, Nishimura et al. [15] showed that a non-closed shuffle can be used to construct a COPY protocol with the minimal number of cards (and the

general one was proposed in [16]) . In 2017, Kaster et al. [3] showed that using non-uniform and/or non-closed shuffles is related to the necessary and sufficient numbers of cards for constructing an AND protocol. In 2018, Ruangwises and Itoh [17] and Koch [4] independently showed a card-minimal AND protocol with non-closed shuffles. In 2018, Hashimoto et al. [2] showed that a non-uniform shuffle is useful for uniformly generating a random permutation without fixed points.

Recently, Koch et al. [5] proposed a card-minimal AND protocol with a standard deck of cards using an RC in 2019. Miyahara et al. [8] proposed efficient implementations of Yao's millionaire protocol using an RC in 2020. A uniform closed shuffle called a pile-scramble shuffle has been often used when constructing zero-knowledge proof protocols for puzzles [9,18] and secure ranking protocols [19].

2 Preliminaries

In this section, we formally explain a deck of cards and introduce the shuffle action called a pile-shifting scramble used in our implementations.

Card. In Section 1.1, we introduced the five-card trick that uses a deck of cards, each of whose front side is $\boxed{\clubsuit}$ or $\boxed{\heartsuit}$ and each of whose backside is $\boxed{?}$. Our implementations use such a two-colored deck of cards as well as *number cards* having numbers written on the front side:

$$\boxed{1}\boxed{2}\cdots\boxed{n}$$

for some natural number n. Those cards satisfy the following two conditions.

- All cards are of the same size and weight, and we cannot distinguish them from the backsides, e.g., there is no scratch on the backsides.
- The pattern on the backside is vertically asymmetric such as $\boxed{?}\boxed{¿}$.

The first condition is necessary for ensuring the security of card-based protocols: If this condition does not hold, a player may identify each face-down card and information about it will be leaked. The second is necessary when we use a two-colored deck of cards as padding cards in our proposed methods, as seen later.

Pile-shifting scramble. Nishimura et al. [14] showed the notion of a pile-shifting scramble and use it for their implementation. It is a shuffle action denoted by $\langle\cdot|\cdot|\cdots|\cdot\rangle$ that randomly shifts a sequence of piles of cards $(A_1, \ldots, A_k)$:

$$\langle \underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{A_1} | \underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{A_2} | \cdots | \underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{A_k} \rangle,$$

where each pile consists of the same number of cards. We demonstrate that it can be implemented by using the vertically asymmetric property, as proposed by Ueda et al. [21].

1. Make the first card of each pile A_i upside down for every $i, 1 \leq i \leq k$:

$$A_i : \boxed{¿}\boxed{?} \cdots \boxed{?}.$$

2. Apply an RC to the sequence of piles:

$$\langle \boxed{¿}\boxed{?}\boxed{?} \cdots \boxed{?}\boxed{¿}\boxed{?}\boxed{?} \cdots \boxed{?} \cdots \boxed{¿}\boxed{?}\boxed{?} \cdots \boxed{?} \rangle.$$

Then, shift the sequence so that an arbitrary upside down card becomes the first one.

3 Our implementation for Two-state Shuffles

In this section, we show our implementation that performs $(\mathsf{shuf}, \{\pi_1, \pi_2\}, \mathcal{F})$ with padding cards, where $\mathcal{F}$ is rational. Note that our implementation does not require any additional tool such as the special physical cases used in [14]. We first show the idea behind our implementation in Section 3.1. Then, we present our methods in the succeeding subsections.

3.1 Idea

Assume a sequence of m face-down cards (to be shuffled). To implement

$$(\mathsf{shuf}, \{\pi_1, \pi_2\}, \mathcal{F}),$$

let us first transform it as follows.

1. To "virtually" transform π_1 to id, we apply a permutation π_1 (to a sequence of m cards), i.e., apply (perm, π_1), before applying the shuffle. Hence, we derive $(\mathsf{shuf}, \{\mathsf{id}, \pi_2\pi_1^{-1}\}, \mathcal{F})$.
2. To see $\pi_2\pi_1^{-1}$ in a different angle, we use the well-known fact that any permutation can be uniquely expressed as the product of disjoint cyclic permutations. That is, $\pi_2\pi_1^{-1}$ can be expressed as $\sigma_1\sigma_2\cdots\sigma_\ell$ for some natural number ℓ where σ_i is a cyclic permutation and any pair of σ_i and σ_j, $i \neq j$, is disjoint; we derive $(\mathsf{shuf}, \{\mathsf{id}, \sigma_1\sigma_2\cdots\sigma_\ell\}, \mathcal{F})$.
3. To make $\sigma_1\sigma_2\cdots\sigma_\ell$ simple, we use another well-known fact that any permutation π can be expressed as a conjugate $\tau\rho\tau^{-1}$ for some permutation τ where ρ is of the same type of π. Using this fact, for example, we can transform $(1\,3\,5\,7\,9)(2\,4\,6\,8\,10)$ to $\tau(1\,2\,3\,4\,5)(6\,7\,8\,9\,10)\tau^{-1}$ where $\tau = (2\,6\,8\,9\,5\,3)(4\,7)$. In this way, we transform $\sigma_1\sigma_2\cdots\sigma_\ell$ to $\tau\rho_1\rho_2\cdots\rho_\ell\tau^{-1}$ where

$$\rho_i = (1{+}\Sigma_{j=1}^{i-1}|\rho_j| \quad 2{+}\Sigma_{j=1}^{i-1}|\rho_j| \quad \cdots \quad \Sigma_{j=1}^{i}|\rho_j|), \tag{3}$$

and $|\pi|$ denotes the length of a cyclic permutation π. Note that ρ_i has the same length of σ_i.

To summarize, we transform $(\mathsf{shuf}, \{\pi_1, \pi_2\}, \mathcal{F})$ into a series of four actions:

$$(\mathsf{perm}, \pi_1)$$
$$(\mathsf{perm}, \tau^{-1})$$
$$(\mathsf{shuf}, \{\mathsf{id}, \rho_1\rho_2\cdots\rho_\ell\}, \mathcal{F})$$
$$(\mathsf{perm}, \tau).$$

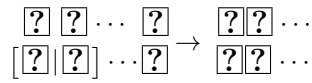
To implement $(\mathsf{shuf}, \{\mathsf{id}, \rho_1\rho_2\cdots\rho_\ell\}, \mathcal{F})$, we first show an implementation of a uniform shuffle $(\mathsf{shuf}, \{\mathsf{id}, \rho_1\rho_2\cdots\rho_\ell\})$ (which might formalize the implementation proposed by Koch and Walzer [6, Appendix B]) using the RC and RBC. Then, by implementating a non-uniform RBC with padding cards, we show our implementation of $(\mathsf{shuf}, \{\mathsf{id}, \rho_1\rho_2\cdots\rho_\ell\}, \mathcal{F})$.

3.2 Uniform and Single Cycle Case

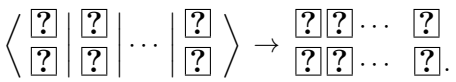
As mentioned before, Koch and Walzer [6] showed an implementation of $(\mathsf{shuf}, \{\mathsf{id}, (1\,2\,3\,4\,5)^3\})$ and stated that its formalization is possible. Let us show that given a single cycle ρ_i in (3), implementing $(\mathsf{shuf}, \{\mathsf{id}, \rho_i\})$ is possible, which is almost the same as the implementation in [6], as follows.

1. Suppose that we have a sequence of $|\rho_i|$ face-down cards (to which we want to apply the shuffle). Place a sequence of additional $|\rho_i|$ face-down cards as padding cards below the sequence:

   The first card in the additional sequence is ♡ and the remaining cards are ♣s.
2. Apply an RBC as follows:



   That is, ♡ is either in the first or the second with the equal probability in the resulting sequence.
3. Considering the cards in the same column as a pile, apply a pile-shifting scramble to the sequence of piles:

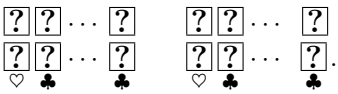


   Remember that this shuffle can be reduced to an RC as shown in Section 2.
4. Reveal the sequence of padding cards; one ♡ should appear.
5. Shift the sequence of piles so that the revealed ♡ becomes in the first position. The resulting upper sequence is either the original one or the shifted one to the right with a probability of 1/2, i.e., the desired shuffle has been applied.

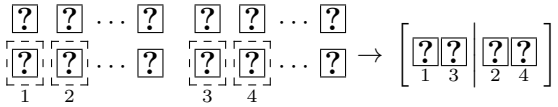
3.3 Uniform and Multiple Cycles Case

The previous subsection implies that applying $(\mathsf{shuf}, \{\mathsf{id}, \rho_i\})$ and $(\mathsf{shuf}, \{\mathsf{id}, \rho_j\})$, $i \neq j$, sequentially is possible. We now show that these two can be applied simultaneously, i.e., $(\mathsf{shuf}, \{\mathsf{id}, \rho_i\rho_j\})$ is possible by performing an RBC once. (Remember that ρ_i and ρ_j are disjoint.)

1. Suppose that we have a sequence of $|\rho_i|$ face-down cards and a sequence of $|\rho_j|$ ones. Place a sequence of $|\rho_i|$ padding cards and a sequence of $|\rho_j|$ padding ones below the two sequences, as follows:



   The first card in each sequence is ♡ and the remaining cards are ♣s.
2. Apply an RBC as follows:



   That is, ♡ is either in the first or the second with a probability of 1/2 in each of the resulting sequences.
3. The remaining steps are exactly the same as the implementation described in the previous subsection. That is, we apply a pile-shifting scramble to each sequence of piles, reveal every additional sequence, and then shift each sequence according to the positions of each of revealed ♡s.

Similarly, $(\mathsf{shuf}, \{\mathsf{id}, \rho_1\rho_2\cdots\rho_\ell\})$ is possible by performing an RBC and pile-shifting scrambles.

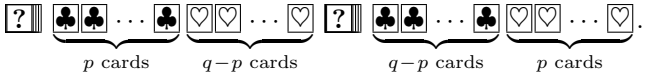
3.4 Non-uniform Case

Here, we consider non-uniform shuffles $(\mathsf{shuf}, \mathsf{id} \mapsto p/q,\ \rho_1\rho_2\cdots\rho_\ell \mapsto 1-p/q)$. Extending the above discussion, it suffices to deal with a non-uniform RBC:

$$(\mathsf{shuf}, \mathsf{id}\mapsto p/q,\ (1\ \ell{+}1)(2\ \ell{+}2)\cdots(\ell\ 2\ell)\mapsto 1-p/q),$$

where the probability p/q is a non-zero rational number and p is relatively prime to q. We show that it can be implemented with $2q$ padding cards, as follows.

1. Suppose that we have a sequence of $m = 2\ell$ face-down cards [?][?] (to be shuffled), each of which consists of ℓ cards. Place q padding cards in the middle of the sequence and also q padding cards next to the resulting sequence, as follows:



   Then, turn over all the face-up cards.

2. Apply an RBC to the sequence:

$$[\boxed{?}\!\!\boxed{?}\cdots\boxed{?}|\boxed{?}\!\!\boxed{?}\cdots\boxed{?}].$$

3. Apply an RC to the left padding cards:

$$\boxed{?}\!\!\boxed{?}\ \langle\boxed{?}\cdots\boxed{?}\rangle\ \boxed{?}\!\!\boxed{?}\cdots\boxed{?}\to\boxed{?}\!\!\boxed{?}\cdots\boxed{?}\,\boxed{?}\!\!\boxed{?}\cdots\boxed{?}.$$

   Note that if the two halves were not swapped by the RBC in step 2, the first card in the left padding cards is ♣ with a probability of p/q. If the two halves were swapped, the first card in the left additional cards is ♣ with a probability of $1-p/q$.
4. Reveal the first card of the left padding cards. If it is ♣, do nothing; otherwise, swap the two halves.[4]

4 Our Implementation for General Case

In this section, let us implement a shuffle

$$(\mathsf{shuf}, \pi_1 \mapsto p_1/q, \pi_2 \mapsto p_2/q, \ldots, \pi_k \mapsto p_k/q), \tag{4}$$

where each probability is a non-zero rational number and the greatest common divisor of $(p_1, \ldots, p_k)$ is relatively prime to q. Note that $\sum_{i=1}^{k} p_i = q$. The following implementation generalizes our implementation of a non-uniform RBC proposed in Section 3.

4.1 Applying a Permutation with Number Cards

Hereinafter, we express a permutation π by a sequence of number cards. For example, using five number cards, express $\pi = (3\ 4\ 1\ 2\ 5)$ as follows:

$$\pi : \boxed{3}\boxed{4}\boxed{1}\boxed{2}\boxed{5}.$$

We show that applying π to a sequence of cards is possible without revealing any information about π using the above sequence, as follows [2]:

1. Place the sequence representing π below to a sequence to which we want to apply π:

$$\begin{array}{c}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\\ \pi : \boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}.\end{array}$$

2. Considering the cards in the two same column as a pile, apply a pile-shifting scramble to the sequence of piles:

$$\left\langle \begin{array}{c}\boxed{?}\\ \boxed{?}\end{array} \middle| \begin{array}{c}\boxed{?}\\ \boxed{?}\end{array} \middle| \cdots \middle| \begin{array}{c}\boxed{?}\\ \boxed{?}\end{array} \right\rangle \to \begin{array}{cc}\boxed{?}\boxed{?}\cdots & \boxed{?}\\ \boxed{?}\boxed{?}\cdots & \boxed{?}.\end{array}$$

[4] After swapping them, the padding $2q$ cards are discarded.

3. Reveal all the cards in the second row. Then, rearrange the two sequences together so that the second row becomes id.

$$\begin{array}{c} \overset{1}{\boxed{?}}\overset{2}{\boxed{?}}\overset{3}{\boxed{?}}\overset{4}{\boxed{?}}\overset{5}{\boxed{?}} \to \overset{5}{\boxed{?}}\overset{1}{\boxed{?}}\overset{2}{\boxed{?}}\overset{3}{\boxed{?}}\overset{4}{\boxed{?}} \\ \boxed{2}\boxed{3}\boxed{4}\boxed{5}\boxed{1} \to \boxed{1}\boxed{2}\boxed{3}\boxed{4}\boxed{5}. \end{array}$$

Thus, to implement the shuffle (4), it suffices to choose a sequence of number cards representing π_i with a probability of p_i/q.

4.2 Description

We are now ready to show our implementation for the general case.

1. Let n be the greatest number among $|\pi_1|, \ldots, |\pi_k|$. Place a sequence of face-down (number) cards that represents π_i, $1 \le i \le k$; we denote it by A_i: $\boxed{?\!\|}$:
$$\pi_i : \overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{n \text{ cards}} \to \ A_i : \boxed{?\!\|}.$$
2. Place q additional cards next to A_i according to the probability of π_i as follows:
$$A_i : \boxed{?\!\|}\ \underbrace{\boxed{1}\boxed{1}\cdots\boxed{1}}_{p_{i+1 \bmod k}}\ \underbrace{\boxed{2}\boxed{2}\cdots\boxed{2}}_{p_{i+2 \bmod k}}\cdots\underbrace{\boxed{k}\boxed{k}\cdots\boxed{k}}_{p_{i+k \bmod k}} \to A'_i : \boxed{?\!\|}.$$
Regard A_i and the q additional cards as a pile denoted by A'_i.
3. Apply a pile-shifting scramble to the sequence of piles $A'_1, A'_2, ..., A'_k$:
$$\langle \boxed{?\!\|} | \boxed{?\!\|} | \cdots | \boxed{?\!\|} \rangle \to \boxed{?\!\|}\ \boxed{?\!\|} \cdots \boxed{?\!\|}.$$
4. Take one pile among the resulting piles; we denote it by A'_j. Apply an RC to its additional cards:
$$A'_j : \boxed{?\!\|}\ \langle \boxed{?}\boxed{?}\cdots\boxed{?} \rangle .$$
5. Reveal any card of the resulting additional cards. Let the revealed card be $\boxed{\ell}$. Then $A'_{j+\ell \bmod k}$ is the desired sequence. That is, the ℓ-th pile to the right of A'_j (up to rotation) is chosen with the desired probability.

5 Conclusion

In this paper, we proposed implementations of any shuffles (including non-uniform and non-cyclic shuffle) $(\mathsf{shuf}, \{\pi_1, \pi_2, \ldots, \pi_k\}, \mathcal{F})$ for a positive integer $k\,(\ge 2)$, where π_i is a permutation and $\mathcal{F}$ is any distribution where every probability is a rational number. For $k = 2$, the main idea is to implement a non-uniform RBC. For $k > 2$, the main idea is to introduce number cards corresponding to permutations, among which we choose one according the distribution.

It is an interesting open problem to deal with non-uniform shuffles whose probabilities are not rational numbers. (This might be impossible.) Also, for $k > 2$, a large number of cards are required depending on the probability distribution in our implementation. To improve upon this would be an intriguing future work.

Acknowledgments.

This work was supported in part by JSPS KAKENHI Grant Numbers JP19J21153 and JP19K11956. We thank the anonymous referees, whose comments have helped us to improve the presentation of the paper.

References

1. Den Boer, B.: More efficient match-making and satisfiability the five card trick. In: Quisquater, J.J., Vandewalle, J. (eds.) Advances in Cryptology—EUROCRYPT '89. LNCS, vol. 434, pp. 208–217. Springer, Berlin, Heidelberg (1990), https://doi.org/10.1007/3-540-46885-4_23
2. Hashimoto, Y., Nuida, K., Shinagawa, K., Inamura, M., Hanaoka, G.: Toward finite-runtime card-based protocol for generating a hidden random permutation without fixed points. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E101.A**(9), 1503–1511 (2018), https://doi.org/10.1587/transfun.E101.A.1503
3. Kastner, J., Koch, A., Walzer, S., Miyahara, D., Hayashi, Y., Mizuki, T., Sone, H.: The minimum number of cards in practical card-based protocols. In: Takagi, T., Peyrin, T. (eds.) Advances in Cryptology—ASIACRYPT 2017. LNCS, vol. 10626, pp. 126–155. Springer, Cham (2017), https://doi.org/10.1007/978-3-319-70700-6_5
4. Koch, A.: The landscape of optimal card-based protocols. Cryptology ePrint Archive, Report 2018/951 (2018), https://eprint.iacr.org/2018/951
5. Koch, A., Schrempp, M., Kirsten, M.: Card-based cryptography meets formal verification. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology–ASIACRYPT 2019. LNCS, vol. 11921, pp. 488–517. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-34578-5_18
6. Koch, A., Walzer, S.: Foundations for actively secure card-based cryptography. Cryptology ePrint Archive, Report 2017/423 (2017), https://eprint.iacr.org/2017/423
7. Koch, A., Walzer, S., Härtel, K.: Card-based cryptographic protocols using a minimal number of cards. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology—ASIACRYPT 2015. LNCS, vol. 9452, pp. 783–807. Springer, Berlin, Heidelberg (2015), https://doi.org/10.1007/978-3-662-48797-6_32
8. Miyahara, D., ichi Hayashi, Y., Mizuki, T., Sone, H.: Practical card-based implementations of Yao's millionaire protocol. Theor. Comput. Sci. **803**, 207–221 (2020), https://doi.org/10.1016/j.tcs.2019.11.005
9. Miyahara, D., Robert, L., Lafourcade, P., Takeshige, S., Mizuki, T., Shinagawa, K., Nagao, A., Sone, H.: Card-based ZKP protocols for Takuzu and Juosan. In: Farach-Colton, M., Prencipe, G., Uehara, R. (eds.) Fun with Algorithms. Leibniz International Proceedings in Informatics (LIPIcs), vol. 157, pp. 20:1–20:21. Schloss Dagstuhl, Dagstuhl, Germany (2020), https://doi.org/10.4230/LIPIcs.FUN.2021.20
10. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. Int. J. Inf. Secur. **13**(1), 15–23 (2014), https://doi.org/10.1007/s10207-013-0219-4
11. Mizuki, T., Shizuya, H.: Computational model of card-based cryptographic protocols and its applications. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E100.A**(1), 3–11 (2017), https://doi.org/10.1587/transfun.E100.A.3

12. Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: Deng, X., Hopcroft, J.E., Xue, J. (eds.) Frontiers in Algorithmics. LNCS, vol. 5598, pp. 358–369. Springer, Berlin, Heidelberg (2009), https://doi.org/10.1007/978-3-642-02270-8_36
13. Nishida, T., Mizuki, T., Sone, H.: Securely computing the three-input majority function with eight cards. In: Dediu, A.H., Martín-Vide, C., Truthe, B., Vega-Rodríguez, M.A. (eds.) Theory and Practice of Natural Computing. LNCS, vol. 8273, pp. 193–204. Springer, Berlin, Heidelberg (2013), https://doi.org/10.1007/978-3-642-45008-2_16
14. Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: Pile-shifting scramble for card-based protocols. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **101**(9), 1494–1502 (2018), https://doi.org/10.1587/transfun.E101.A.1494
15. Nishimura, A., Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Five-card secure computations using unequal division shuffle. In: Dediu, A.H., Magdalena, L., Martín-Vide, C. (eds.) Theory and Practice of Natural Computing. LNCS, vol. 9477, pp. 109–120. Springer, Cham (2015), https://doi.org/10.1007/978-3-319-26841-5_9
16. Nishimura, A., Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Card-based protocols using unequal division shuffles. Soft Comput. **22**, 361–371 (2018), https://doi.org/10.1007/s00500-017-2858-2
17. Ruangwises, S., Itoh, T.: AND protocols using only uniform shuffles. In: van Bevern, R., Kucherov, G. (eds.) Computer Science–Theory and Applications. LNCS, vol. 11532, pp. 349–358. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-19955-5_30
18. Sasaki, T., Miyahara, D., Mizuki, T., Sone, H.: Efficient card-based zero-knowledge proof for Sudoku. Theor. Comput. Sci. **839**, 135–142 (2020), https://doi.org/10.1016/j.tcs.2020.05.036
19. Takashima, K., Abe, Y., Sasaki, T., Miyahara, D., Shinagawa, K., Mizuki, T., Sone, H.: Card-based secure ranking computations. In: Li, Y., Cardei, M., Huang, Y. (eds.) Combinatorial Optimization and Applications. LNCS, vol. 11949, pp. 461–472. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-36412-0_37
20. Takashima, K., Miyahara, D., Mizuki, T., Sone, H.: Card-based protocol against actively revealing card attack. In: Martín-Vide, C., Pond, G., Vega-Rodríguez, M.A. (eds.) Theory and Practice of Natural Computing. LNCS, vol. 11934, pp. 95–106. Springer, Cham (2019), https://doi.org/10.1007/978-3-030-34500-6_6
21. Ueda, I., Miyahara, D., Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: Secure implementations of a random bisection cut. Int. J. Inf. Secur. **19**(4), 445–452 (2020), https://doi.org/10.1007/s10207-019-00463-w
22. Ueda, I., Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: How to implement a random bisection cut. In: Martín-Vide, C., Mizuki, T., Vega-Rodríguez, M.A. (eds.) Theory and Practice of Natural Computing. LNCS, vol. 10071, pp. 58–69. Springer, Cham (2016), https://doi.org/10.1007/978-3-319-49001-4_5