Vol.2019-CSEC-85 No.16
Vol.2019-IOT-45 No.16
2019/5/24

シンプルなカードベース置換生成に関する一考察

齋藤 敬宏[1,a)]　千田 栄幸[2]　水木 敬明[3]

概要：不動点を持たない置換はプレゼント交換の際に有益であり，カード組を用いて秘匿したまま生成するプロトコルがこれまで数多く考案されている．著者らは CSS 2018 において，既存の手法を組み合わせ改良し，シンプルな置換生成プロトコルを提案した．すなわち，各プレーヤーごとに識別カードとして 2 つの同一なものを用意し，Pile-Scramble シャッフルを適用後，巡回シフトを 1 回行い，最後にもう一度 Pile-Scramble シャッフルを行うことで，最長サイクルを一様ランダムに生成できることを指摘した．本稿では，前述の手法の中の巡回シフトの代わりに，別な置換やシャッフルを適用することで，いろいろな分布の置換を生成できることを示す．

キーワード：カードベース暗号，プレゼント交換，不動点, ランダム置換

A Note on Simple Card-Based Generation of Random Permutation

Takahiro Saito[1,a)]　Eikoh Chida[2]　Takaaki Mizuki[3]

1. はじめに

n 人のプレイヤーがいて，プレゼント交換を行いたい場面を考える．各々のプレイヤーが自分自身の贈り主になるのは避けたいので，不動点 (fixed point) を持たない置換 $\pi \in S_n$ をランダムに生成したい．ここで S_n は, n 次対称群を意味する．カード組を用いると，そのようなランダム置換を秘匿したままで生成できるとともに, 置換そのものを明らかにすることなく, 不動点を持たないことの証明が可能であることが次に示すとおり知られている．

1.1 既存プロトコル

Crépeau と Kilian は, 裏面が同一の模様 ? である 4 色のカードを $2n^2$ 枚用いて不動点をもたない置換 $\pi \in S_n$ を一様ランダムに生成するプロトコルを与えた [2]. Ishikawa らは，Crépeau と Kilian のプロトコルを改良し, 各プレーヤーに対応するカード束の表現を工夫することなどで, 2 色のカードを $n\lceil \log n \rceil + 6$ 枚[*1]用いて不動点を持たない置換を生成できることを示した [6]. また, Ibaraki と Manabe は, Pile-Scramble シャッフルと巡回シャッフルにより, 2 色のカードを $4n\lceil \log n \rceil + 6$ 枚を用いて, ランダム置換を生成するプロトコルを提案した [5].

さらに, 著者らは CSS 2018 において, Ibaraki と Manabe のプロトコルをベースに巡回シャッフルの代わりに巡回シフトを用いて, 不動点を持たない長さ n のサイクルをランダムに生成するプロトコルを提案した [9] ．また，Hashimoto らは主に数字カードを用いて任意の型の置換をランダムに生成する手法を与えている [4].

1.2 本稿の貢献

著者らが CSS 2018 において提案したプロトコル [9] が

[1] 一関高専 専攻科生産工学専攻
Advanced Course of Prouduction Engineering, National Institute of Technology,Ichinoseki College, Takanashi, Hagisyo, Ichinoseki-shi, Iwate 021-8511, Japan
[2] 一関高専 未来創造工学科情報・ソフトウェア系
Department of Engineering for Future Inovation, Division of Computer Engineering and Informatics, Takanashi, Hagisyo, Ichinoseki-shi, Iwate 021-8511, Japan
[3] 東北大学サイバーサイエンスセンター
Cyberscience Center Tohoku University, 6-3, Aoba, Aramaki, Aoba-ku, Sendai-shi, Miyagi 980-8578, Japan
[a)] a18613@g.ichinoseki.ac.jp

[*1] 本文で用いる log の底は全て 2 とする．

生成する置換は巡回置換となっており，具体的に生成される巡回置換のサイクル長はプレイヤー人数 n と等しい最長サイクルに固定されている．本稿の 3 節では CSS 2018 で提案したプロトコルが, 長さ n のサイクルを一様ランダムに生成することを明示的に証明する．4 節では著者らが提案したプロトコルの中の巡回シフトの部分に対し，別な置換を適用することで最長サイクルに限らず，様々な長さのサイクルを含む巡回置換を生成できることを示す．さらに 5 節ではそれに伴うサイクル長の変化等について解析を行う．6 節では置換の型を秘匿にしたまま置換生成を行うプロトコルとして，Permutation Randomizing Protocol [4] を紹介する．その後，同様な置換生成が可能であるプロトコルを提案する．

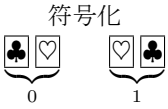
2. 準備

本節では本稿で取り扱うカードの性質とカード列に対するシャッフルについて説明する．

2.1 カード

本稿において, カードは表面に 10 進数の数値が書かれたもの ($\boxed{1}\,\boxed{2}\,\boxed{3}\,\cdots$) を用いる．また裏面の模様はいずれのカードも $\boxed{?}$ とし，裏面のみでは表面の数値がどれかわからないものとする．ここで, プロトコルに用いるカードは全て同じサイズと重さで傷等はついておらず模様以外ではカードの特定は不可能であるとする．また, 裏返されたカードを**コミットメント**と呼ぶ.

いくつかの既存プロトコルのおいては, 表面が ♣ と ♡ の 2 種類のものを用い, 2 枚の組のカードに対して，2 進数の 0 と 1 を対応させるため次のように**符号化 (エンコーディング)** を行っているものがある [6].



2.2 シャッフル

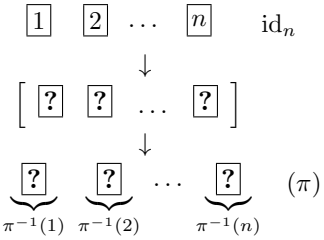
以下のとおり, 本稿で紹介するプロトコルにおいて利用するシャッフルについて説明する.

2.2.1 Pile-Scramble シャッフル [6]

Pile-Scramble シャッフルは, 同じ枚数で構成されるカード束が ℓ 個ある時，$\ell!$ 通りの束の並びからランダムに 1 つの並びが選ばれるシャッフルのことを指す．クリップや封筒を用いることで容易に実装することができる．

各束が 1 枚の数字カードである場合，Pile-Scramble シャッフルはいわゆる通常のシャッフルとなる．すなわち ℓ 個のカード束は ℓ 枚の数字カードに対応する．数字カードの系列が $(x_1, x_2, x_3, ..., x_\ell)$ であるとき, $(x_{\pi^{-1}(1)}, x_{\pi^{-1}(2)}, x_{\pi^{-1}(3)}, ..., x_{\pi^{-1}(\ell)})$ を得るようなシャッフルとなる．ここで $\pi \in S_m$ はランダム置換を表す.

ここで，id_n を恒等置換とし，数字カードの系列 $(1, 2, \cdots, n)$ に対して，Pile-Scramble シャッフルを実行したときに得られる置換を以下のとおりに表現する.



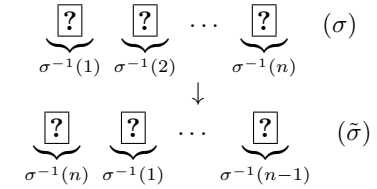
2.2.2 巡回シフトと巡回シャッフル

ℓ 個のカードの束を 1 つ巡回させるのが巡回シフトである．さらに巡回シフトを任意の回数だけ実行し，その回数がランダムであり，かつ誰もその回数を知らないとき，巡回シャッフルとなる．巡回シャッフルの結果に対応する置換の総数は ℓ 通りである.

例えば, 6 枚の数字カードの系列が $(x_1, x_2, x_3, x_4, x_5, x_6)$ であるとき, 巡回シャッフルの置換のパターンは次の 6 通りである.

$(x_1, x_2, x_3, x_4, x_5, x_6)$, $(x_2, x_3, x_4, x_5, x_6, x_1)$,
$(x_3, x_4, x_5, x_6, x_1, x_2)$, $(x_4, x_5, x_6, x_1, x_2, x_3)$,
$(x_5, x_6, x_1, x_2, x_3, x_4)$, $(x_6, x_1, x_2, x_3, x_4, x_5)$.

ここで, ある置換 σ について, 数字カードの系列 $(\sigma(1), \sigma(2), \cdots, \sigma(n))$ に対して, 巡回シフトを 1 回のみ実行したときに得られる置換 $\tilde{\sigma}$ は以下のとおりとなる.



2.3 巡回置換の表現方法

任意の置換は互いに素な巡回置換の積で表せることが知られている [1].

例えば，次のような置換 $\tau \in S_6$ を考える.

$$(\tau(1), \cdots, \tau(6)) = (5, 1, 6, 4, 2, 3)$$

この置換 τ を互いに素な巡回置換の積で表すと,

$$\tau = (125)(4)(36)$$

となり，τ が 3 つの巡回置換に分解されている．本稿では置換を巡回置換の積で表したとき，構成要素である巡回置換をサイクルエリアと呼ぶ. 長さが 1 のサイクルエリアは不動点となることに注意する必要がある.

次に置換の型を考える．型とはサイクルエリアの組み合わせを表すものである．置換が長さが 1 のサイクルエリア m_1 個，2 のサイクルエリア m_2 個, $\cdots$, n のサイクルエリア m_n 個に分解される場合，置換の型は次のように表される.

$$\langle 1^{m_1}, 2^{m_2}, \cdots, n^{m_n} \rangle.$$

例えば $n = 6$ の場合, 不動点を持たない置換に限ると型は, 以下の 4 とおりである.

$$\langle 2^3 \rangle, \langle 2^1, 4^1 \rangle, \langle 3^2 \rangle, \langle 6^1 \rangle.$$

3. 最長サイクルの一様ランダム置換生成プロトコル

著者らは CSS 2018 にて不動点を持たない最長サイクルの巡回置換を生成するプロトコルを提案した. 以下, プロトコルを紹介するとともに類似した手法である Secret Santa を併せて紹介する.

3.1 プロトコル

(1) 各プレイヤー p_i に対して, 表面に i の数字が書かれたカードを 2 枚用意し, 次のように並べる.

プレイヤー ID : $\boxed{1}\ \boxed{2} \cdots\ \boxed{n}$
ギフト　ID : $\boxed{1}\ \boxed{2} \cdots\ \boxed{n}$

1 行目に置かれたカードをプレイヤー ID と呼び, 2 行目に置かれたカードをギフト ID と呼ぶ.

(2) プレイヤー ID とギフト ID を裏返し, ペアにして Pile-Scramble シャッフルを実行する. シャッフルによりランダムな置換 $r \in S_n$ によって並べ替えが起こったとする.

プレイヤー ID : $\underbrace{\boxed{?}}_{1}\ \underbrace{\boxed{?}}_{2} \cdots \underbrace{\boxed{?}}_{n}$
ギフト　ID : $\underbrace{\boxed{?}}_{1}\ \underbrace{\boxed{?}}_{2} \cdots \underbrace{\boxed{?}}_{n}$

$$\downarrow$$

$$\begin{bmatrix} \boxed{?} & \boxed{?} & \cdots & \boxed{?} \\ \boxed{?} & \boxed{?} & \cdots & \boxed{?} \end{bmatrix}$$

$$\downarrow$$

$$\underbrace{\boxed{?}}_{r^{-1}(1)}\ \underbrace{\boxed{?}}_{r^{-1}(2)} \cdots \underbrace{\boxed{?}}_{r^{-1}(n)}$$
$$\underbrace{\boxed{?}}_{r^{-1}(1)}\ \underbrace{\boxed{?}}_{r^{-1}(2)} \cdots \underbrace{\boxed{?}}_{r^{-1}(n)}$$

(3) プレイヤー ID とギフト ID のペアを切り離し, ギフト ID のみに対して巡回シフトを 1 回のみ実行する.

$$\underbrace{\boxed{?}}_{r^{-1}(1)}\ \underbrace{\boxed{?}}_{r^{-1}(2)} \cdots \underbrace{\boxed{?}}_{r^{-1}(n)}$$
$$\underbrace{\boxed{?}}_{r^{-1}(n)}\ \underbrace{\boxed{?}}_{r^{-1}(1)} \cdots \underbrace{\boxed{?}}_{r^{-1}(n-1)}$$

(4) もう一度プレイヤー ID とギフト ID をペアにして, Pile-Scramble シャッフルを実行する. 再び, シャッフルによりランダムな置換 $\pi \in S_n$ によって並べ替えが起こったとする.

$$\underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(1))}\ \underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(2))} \cdots \underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(n))}$$
$$\underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(n))}\ \underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(1))} \cdots \underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(n)-1))}$$

(5) プレイヤー ID のみを表側にする. プレイヤー ID に対応するプレイヤーがペアになっているギフト ID を確認する. 各プレイヤーは確認したギフト ID が示すプレイヤーへとギフトを贈る.

プレイヤー ID:
$\boxed{r^{-1}(\pi^{-1}(1))}\ \boxed{r^{-1}(\pi^{-1}(2))} \ldots \boxed{r^{-1}(\pi^{-1}(n))}$

ギフト ID:
$$\underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(n))}\ \underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(1))} \cdots \underbrace{\boxed{?}}_{r^{-1}(\pi^{-1}(n)-1))}$$

3.2 プロトコルの正当性

このプロトコルでは, ステップ (2) においてランダムな置換 $r \in S_n$ が発生し, ステップ (3) の巡回シフト後の時点で

$$\begin{pmatrix} r^{-1}(1) & r^{-1}(2) & \ldots & r^{-1}(n) \\ r^{-1}(n) & r^{-1}(1) & \ldots & r^{-1}(n-1) \end{pmatrix},$$

すなわち

$$(r^{-1}(1)\ r^{-1}(2)\ \cdots\ r^{-1}(n))$$

という巡回置換が生成されている. これは長さ n の最長サイクルになっており, この置換を $gen(r)$ と書くことにする.

こののち, ステップ (4) において Pile-Scramble シャッフルが適用されるが, 以下の Lemma 3.1 により最終的に生成される巡回置換そのものには寄与しないことに注意する.

Lemma 3.1
任意の置換

$$\sigma = \begin{pmatrix} i_1 & i_2 & \ldots & i_n \\ j_1 & j_2 & \ldots & j_n \end{pmatrix}$$

を考える. このとき, 任意の置換 $\pi \in S_n$ に対して,

$$\begin{pmatrix} i_{\pi^{-1}(1)} & i_{\pi^{-1}(2)} & \ldots & i_{\pi^{-1}(n)} \\ j_{\pi^{-1}(1)} & j_{\pi^{-1}(2)} & \ldots & j_{\pi^{-1}(n)} \end{pmatrix} = \sigma$$

である.

したがって, ステップ (2) で生成される置換を $r \in S_n$ とするとき, $gen(r)$ が最終的に出力される置換となる.

また, 置換

$$r = \begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ r(1) & r(2) & r(3) & \ldots & r(n) \end{pmatrix}$$

の 2 行目を“シフトさせた”置換

$$r' = \begin{pmatrix} 1 & 2 & 3 & \ldots & n \\ r(n) & r(1) & r(2) & \ldots & r(n-1) \end{pmatrix}$$

を考えると, プロトコルの最終的な出力は同一の最長サイクルである. すなわち, $gen(r') = gen(r)$ である.

したがって, $gen(r)$ を生成するようなシャッフル源の置換はちょうど n 個存在する. 各シャッフル源の置換は $1/n!$ の確率で発生するので, $gen(r)$ は $1/(n-1)!$ の確率で生成される. 一方, 最長サイクルは, $(n-1)!$ 個存在するので, このプロトコルは, 一様ランダムな最長サイクルの置換を

生成していることになる．

3.3 Secret Santa [3]

YouTube にて，最長サイクルの置換生成プロトコルに類似した手法を紹介する動画が公開されている [3]．以下に，CSS 2018 で提案したプロトコルとの相違点を挙げる．

- 再利用不可能な紙片を利用している
- 名前リストを作る必要がある
- 生成された巡回置換をそのまま配布するため秘匿性に問題がある

著者らの手法と組み合わせると次のように改良できる．まず, 紙片に数字ではなく直接名前を書くことによって，名前と数字を対応させた名前リストを作る必要がなくなる．また, 紙片を切り巡回シフトを実行した後に, 再びテープで紙片をとめてシャッフルをすることで巡回置換の秘匿性が確保される．

4. 任意の型の置換生成プロトコル

本節では，著者らが CSS 2018 にて提案した最長サイクルの生成プロトコルを一般化し，任意の型の巡回置換を生成するプロトコルを提案する．

4.1 プロトコル

(1) 各プレイヤー p_i に対して，表面に i の数字が書かれたカードを 2 枚用意し次のように並べる．

$$\text{プレイヤー ID}: \boxed{1}\ \boxed{2}\cdots\ \boxed{n}$$
$$\text{ギフト　ID}: \boxed{1}\ \boxed{2}\cdots\ \boxed{n}$$

(2) プレイヤー ID とギフト ID を裏返し，ペアにして Pile-Scramble シャッフルを実行する．$r \in S_n$ をランダムな置換とする．

$$\begin{bmatrix} ? & ? & \cdots & ? \\ ? & ? & \cdots & ? \end{bmatrix}$$
$$\downarrow$$
$$\underbrace{?}_{r^{-1}(1)}\ \underbrace{?}_{r^{-1}(2)}\ \cdots\ \underbrace{?}_{r^{-1}(n)}$$
$$\underbrace{?}_{r^{-1}(1)}\ \underbrace{?}_{r^{-1}(2)}\ \cdots\ \underbrace{?}_{r^{-1}(n)}$$

(3) ギフト ID のみに対して巡回シフトを実行する．ただし, 作りたい置換の型に合わせ，各サイクルエリアごとに巡回シフトを実行する．ℓ をサイクルの数, $c_i (1 \le i \le \ell)$ を各サイクルの長さとする．

$$\underbrace{\underbrace{?}_{r^{-1}(1)}\ \cdots\ \underbrace{?}_{r^{-1}(c_1)}}_{\text{サイクルエリア 1}}\ \underbrace{\underbrace{?}_{r^{-1}(c_1+1)}\ \cdots\ \underbrace{?}_{r^{-1}(c_1+c_2)}}_{\text{サイクルエリア 2}}$$
$$\cdots\quad \underbrace{\underbrace{?}_{r^{-1}(c_1+\cdots+c_{\ell-1}+1)}\ \cdots\ \underbrace{?}_{r^{-1}(c_1+\cdots+c_\ell)}}_{\text{サイクルエリア } \ell}$$
$$\downarrow$$
$$\underbrace{\underbrace{?}_{r^{-1}(c_1)}\ \cdots\ \underbrace{?}_{r^{-1}(c_1-1)}}_{\text{サイクルエリア 1}}\ \underbrace{\underbrace{?}_{r^{-1}(c_1+c_2)}\ \cdots\ \underbrace{?}_{r^{-1}(c_1+c_2-1)}}_{\text{サイクルエリア 2}}$$
$$\cdots\quad \underbrace{\underbrace{?}_{r^{-1}(c_1+\cdots+c_\ell)}\ \cdots\ \underbrace{?}_{r^{-1}(c_1+\cdots+c_\ell-1)}}_{\text{サイクルエリア } \ell}$$

(4) もう一度プレイヤー ID とギフト ID をペアにして，Pile-Scramble シャッフルを実行する．$\pi \in S_n$ をランダムな置換とする．

$$\begin{bmatrix} ? & ? & \cdots & ? \\ ? & ? & \cdots & ? \end{bmatrix}$$

(5) プレイヤー ID のみを表側にする．プレイヤー ID に対応するプレイヤーがペアになっているギフト ID を確認する．各プレイヤーは確認したギフト ID が示すプレイヤーへとギフトを贈る．

このプロトコルでは

$$(r^{-1}(1)\cdots r^{-1}(c_1))\ (r^{-1}(c_1+1)\cdots r^{-1}(c_1+c_2))\cdots (r^{-1}(c_1+\cdots+c_{\ell-1}+1)\cdots r^{-1}(c_1+\cdots+c_\ell))$$

という置換が生成される．

ステップ (3) に関して, 例えば $n=6$ の時に $\langle 2^1, 4^1\rangle$ の型の置換を作りたい場合は 1 つめのサイクルエリア (長さ 2), 2 つめのサイクルエリア (長さ 4) に対して，それぞれ巡回シフトを実行する．巡回シフトを実行する範囲，すなわちサイクルエリアを変えることで任意の型の巡回置換を生成することができる．

4.2 巡回シフトの量を変化させた場合

4.1 節のプロトコルでは，各サイクルエリア内で出力される置換の長さは必ず各サイクルエリア長と等しくなっている．なぜならば各サイクルエリア内で行う巡回シフトのシフト量が 1 であるからである．一方，巡回シフトのシフト量を増やすと，サイクルエリアが分解されサイクルエリア長よりも短いサイクルが複数出力されることがある．その時，サイクルエリア長とシフト量の最大公約数の値だけ分解される．

具体例として，長さが 6 のサイクルエリアに対して巡回シフトを 3 回行った場合を考える．長さとシフト量の最大公約数 $\gcd(3,6)=3$ であり，生成される置換の型は

$$\langle 2^3\rangle$$

となる．元のサイクルエリアが 3 つに分解され, 長さ 2 のサイクルが 3 つ生成される．

ここで巡回シフトの性質をまとめると，サイクルエリアの長さを c，シフト量を s としたときに，生成されるサイクルの長さと個数は

$$\text{サイクルの長さ}: \frac{c}{\gcd(s,c)}$$

$$\text{サイクルの個数}: \gcd(s,c)$$

で表される．特にサイクルエリアが分解されず生成される巡回置換がサイクルエリア長と等しくなるための条件は

$$\gcd(s, c) = 1 \tag{3}$$

であり，サイクルエリア長とシフト量が互いに素である必要がある．

本節で提案した任意の型の置換生成プロトコルに改めて注目すると，各サイクルエリア内での巡回シフト量は 1 である．すなわちサイクルエリア長に関わらず常に式 (3) を満たすため，各サイクルエリア内で出力される巡回置換の長さは各サイクルエリアにおいて必ずサイクルエリア長と等しくなる．

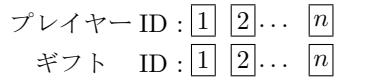
5. Ibaraki-Manabe プロトコルの解析

Ibaraki と Manabe [5] は 2016 年に巡回シャッフルを用いて，不動点を持たない巡回置換を生成するプロトコルを提案している．巡回シャッフルは, 巡回シフトをランダムな回数だけ実行することで実現できるため, 彼らが提案したプロトコルは, 4.1 節で提案したプロトコルの変形版とも言うことができる．

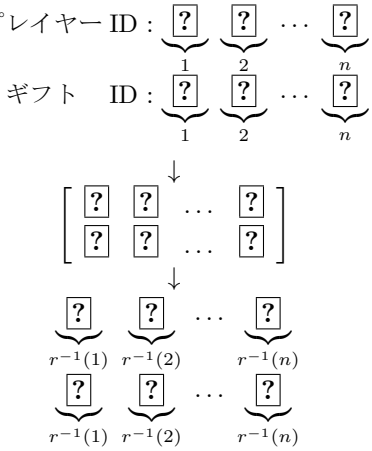
本節では, 彼らのプロトコルを紹介したあとに, 生成される置換のサイクル長について解析を行う．

5.1 プロトコル

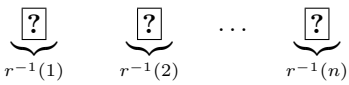
(1) 各プレイヤー p_i に対して，表面に i の数字が書かれたカードを 2 枚用意し，次のように並べる．

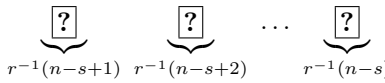


(2) ギフト ID とプレイヤー ID を裏返し，ペアにして Pile-Scramble シャッフルを実行する．シャッフルにより，次のようにランダムな置換 $r \in S_n$ で並べ替えが起こったとする．



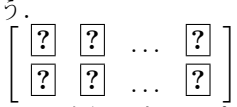
(3) プレイヤー ID とギフト ID のペアを切り離し, ギフト ID のみに対して巡回シャッフルを行う．ランダムなシフト量 s で並べ替えが起こったとする．





(4) プレイヤー ID $r^{-1}(1)$ とギフト ID $r^{-1}(n-s+1)$ を一組取り出し，不動点チェック (詳細は [5][6] を参照) を実行する．もし不動点だった場合にはステップ (3) の巡回シャッフルからやり直す.

(5) 不動点ではないことが確認できたら, もう一度プレイヤー ID とギフト ID をペアにして Pile-Scramble シャッフルを行う．



(6) プレイヤー ID のみを表側にする．プレイヤー ID に対応するプレイヤーがペアになっているギフト ID を確認する．各プレイヤーは確認したギフト ID が示すプレイヤーへとギフトを贈る．

以上のとおり, 巡回シャッフルを用いて, Ibaraki と Manabe が提案したプロトコルは不動点を持たない置換を生成している．

5.2 巡回シャッフルによる置換生成の解析

前述のとおり, 巡回シャッフルは巡回シフトをランダムな回数実行したものであり, 3 節で述べた巡回シフトによる最長サイクルのランダム置換生成プロトコルとは違い, 生成される置換が最長サイクルにならない場合がある．本節では, 各プレイヤー人数に対して，生成される置換の平均サイクル長がどのように変化するか解析する．

プレイヤーの人数を n とすると, 各シフト量 s は $1/(n-1)$ の確率で発生し, 生成される置換の平均サイクル長 $\overline{C_n}$ は次の式で求めることができる．

$$\overline{C_n} = \sum_{s=1}^{n-1} \frac{n}{(n-1) \cdot \gcd(s, n)} \tag{4}$$

式 (4) を用いて，プレイヤー人数が 3 人から 30 人の場合の平均サイクル長を求めると次の表とグラフのとおりとなる．

平均サイクル長を直線近似すると, もちろんプレーヤーの人数 n が大きくなればなるほど増加傾向にある．平均サイクル長のプレイヤーの人数に対する割合は, プレイヤーの人数が素数である場合, 生成される置換が最長サイクルとなるため 100% となる．

プレイヤーの人数が合成数である場合, 複数の長さが同一の置換の積となるため平均サイクル長はプレーヤーの人数に対しておよそ 3 割から 4 割程度短くなっている．

* プレーヤーの人数=平均サイクル長

Vol.2019-CSEC-85 No.16
Vol.2019-IOT-45 No.16
2019/5/24

表 1 平均サイクル長と各プレーヤー人数に対する割合

プレイヤーの人数	平均サイクル長	各プレーヤー人数に対する割合
3	*3.00	100%
4	3.33	83%
5	*5.00	100%
6	4.00	67%
7	*7.00	100%
8	6.00	75%
9	7.50	83%
10	6.89	69%
11	*11.0	100%
12	6.91	58%
13	*13.0	100%
14	9.85	70%
15	10.4	70%
16	11.3	71%
17	*17.0	100%
18	10.7	59%
19	*19.0	100%
20	12.1	61%
21	14.7	70%
22	15.7	72%
23	*23.0	100%
24	13.2	55%
25	21.8	87%
26	18.6	71%
27	20.7	77%
28	18.0	64%
29	*29.0	100%
30	15.8	53%

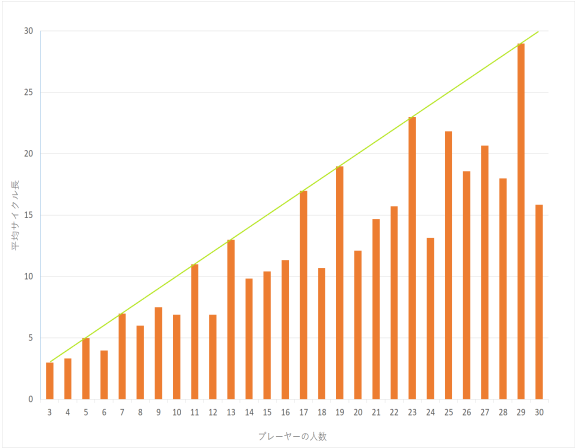


図 2 平均サイクル長の解析

6. 置換の型を秘匿した置換生成プロトコル

4 節で示した置換生成プロトコルは各サイクルエリアに対して巡回シフトを実行している．巡回シフトは全てのプレイヤーの目の前で行われるため，プレイヤーは巡回シフトの適用範囲から各サイクルエリア長を知ることができる．置換の型はサイクルエリアの組み合わせを表すものであり，各プレイヤーは置換の型を容易に特定することができる．その一方，置換の型を秘匿にしたまま置換生成を行うプロトコルも存在する．

本節ではそのようなプロトコルとして，Permutation Randomizing Protocol [4] を紹介する．さらに置換の型を秘匿にしたまま置換生成を行う別なプロトコルを，4 節の手法を応用することで提案する．

6.1 Permutation Randomizing Protocol

Permutation Randomizing Protocol は，あるコミットメント置換 σ が与えられたときに，σ と同じ型の一様ランダムな置換を得るプロトコルである．具体的には，$\pi \in S_n$ を一様ランダムに選んだ後，$\pi^{-1}\sigma\pi$ を出力している．はじめにこのプロトコルにおいて必要となる Permutation Division Protocol [4] を紹介する．

6.1.1 サブプロトコル

Permutation Division Protocol は，2 つのコミットメント置換 v と w に対して，コミットメント置換 $v^{-1}w$ を得るプロトコルである．

(1) コミットメント置換 v と w を以下のように並べる．

$$\boxed{?}\boxed{?}\cdots\boxed{?}\quad(v)$$
$$\boxed{?}\boxed{?}\cdots\boxed{?}\quad(w)$$

(2) 上下 2 枚のカードを 1 つのペアにして Pile-Scramble シャッフルを行う．ランダムな置換 $r \in S_n$ で並び替えが起こったとする．

$$\left[\begin{array}{ccc}\boxed{?} & \boxed{?} & \cdots & \boxed{?}\\ \boxed{?} & \boxed{?} & \cdots & \boxed{?}\end{array}\right]$$
$$\downarrow$$
$$\boxed{?}\boxed{?}\cdots\boxed{?}\quad(rv)$$
$$\boxed{?}\boxed{?}\cdots\boxed{?}\quad(rw)$$

(3) 1 行目を表向きにして恒等置換となるように並び替える．ただし，対応する 2 行目のカードも同時に並び替えるようにする．

$$\begin{array}{cccc}\boxed{r^{-1}(v^{-1}(1))} & \boxed{r^{-1}(v^{-1}(2))} & \cdots & \boxed{r^{-1}(v^{-1}(n))}\\ \boxed{?} & \boxed{?} & \cdots & \boxed{?}\quad(rw)\end{array}$$
$$\downarrow$$
$$\boxed{1}\boxed{2}\cdots\boxed{n}$$
$$\boxed{?}\boxed{?}\cdots\boxed{?}\quad(v^{-1}w)^{\dagger}$$

(4) 2 行目の $v^{-1}w$ を出力として得る．

次にこのサブプロトコルを用いて本プロトコルを紹介する．

6.1.2 本プロトコル

コミットメント置換 σ は与えられているものとする．

(1) 恒等置換を 2 つ用意し，以下のように並べる．さらに

† $(rv)^{-1}(rw) = v^{-1}r^{-1}rw = v^{-1}w$

Pile-Scramble シャッフルを上下 2 枚のカードを 1 つのペアにして行う．ランダムな置換 $\pi \in S_n$ で並び替えが起こったとする．

$$\begin{bmatrix} ? & ? & \cdots & ? \\ ? & ? & \cdots & ? \end{bmatrix} \downarrow \quad \begin{matrix} ?\,?\,\cdots\,? & (\pi) \\ ?\,?\,\cdots\,? & (\pi) \end{matrix}$$

(2) コミットメント置換 σ と π に対して，Permutation Division Protocol を適用し，$\sigma^{-1}\pi$ を得る．

(3) $\sigma^{-1}\pi$ と π に対してもう一度 Permutation Division Protocol を適用し，$\pi^{-1}\sigma\pi$ を得る．

以上により，シャッフル 3 回によって σ と同じ型の置換を一様ランダムに生成できることが分かる．

6.2 提案プロトコル

ここでは，4 節の手法を応用することでも，あるコミットメント置換 σ の型は秘匿にしつつ同じの型の一様ランダムな置換を得ることができることを示す．

(1) 各プレイヤー p_i に対して，表面に i の数字が書かれたカードを 2 枚用意し，次のように並べる

$$\text{プレイヤー ID}: \boxed{1}\ \boxed{2}\cdots\ \boxed{n}$$
$$\text{ギフト　ID}: \boxed{1}\ \boxed{2}\cdots\ \boxed{n}$$

(2) ギフト ID とプレイヤー ID を裏返し，ペアにして Pile-Scramble シャッフルを実行する．シャッフルにより，次のようにランダムな置換 $r_1 \in S_n$ で並べ替えが起こったとする．

$$\text{プレイヤー ID}: \underbrace{?}_{1}\ \underbrace{?}_{2}\ \cdots\ \underbrace{?}_{n}$$
$$\text{ギフト　ID}: \underbrace{?}_{1}\ \underbrace{?}_{2}\ \cdots\ \underbrace{?}_{n}$$
$$\downarrow$$
$$\begin{bmatrix} ? & ? & \cdots & ? \\ ? & ? & \cdots & ? \end{bmatrix} \downarrow \quad \begin{matrix} \underbrace{?}_{r_1^{-1}(1)}\ \underbrace{?}_{r_1^{-1}(2)}\ \cdots\ \underbrace{?}_{r_1^{-1}(n)} & (r_1) \\ \underbrace{?}_{r_1^{-1}(1)}\ \underbrace{?}_{r_1^{-1}(2)}\ \cdots\ \underbrace{?}_{r_1^{-1}(n)} & (r_1) \end{matrix}$$

(3) コミットメント置換 σ をギフト ID の下に並べる．ギフト ID とコミットメント置換をペアにし Pile-Scramble シャッフルを実行する．シャッフルにより，ランダムな置換 $\pi \in S_n$ で並べ替えが起こったとする．

$$\text{ギフト　ID}: \underbrace{?}_{r_1^{-1}(1)}\ \underbrace{?}_{r_1^{-1}(2)}\ \cdots\ \underbrace{?}_{r_1^{-1}(n)} \quad (r_1)$$
$$\text{コミットメント置換}: \underbrace{?}_{\sigma^{-1}(1)}\ \underbrace{?}_{\sigma^{-1}(2)}\ \cdots\ \underbrace{?}_{\sigma^{-1}(n)} \quad (\sigma)$$
$$\downarrow$$
$$\begin{bmatrix} ? & ? & \cdots & ? \\ ? & ? & \cdots & ? \end{bmatrix} \downarrow \quad \begin{matrix} \underbrace{?}_{\pi^{-1}(r_1^{-1}(1))}\ \underbrace{?}_{\pi^{-1}(r_1^{-1}(2))}\ \cdots\ \underbrace{?}_{\pi^{-1}(r_1^{-1}(n))} & (\pi r_1) \\ \underbrace{?}_{\pi^{-1}(\sigma^{-1}(1))}\ \underbrace{?}_{\pi^{-1}(\sigma^{-1}(2))}\ \cdots\ \underbrace{?}_{\pi^{-1}(\sigma^{-1}(n))} & (\pi\sigma) \end{matrix}$$

(4) コミットメント置換を表向きにして恒等置換となるように並び替える．ただし，対応するギフト ID も裏側にしたまま同時に並び替えるようにする．

$$\text{ギフト ID}: \underbrace{?}_{\pi^{-1}(r_1^{-1}(1))}\ \underbrace{?}_{\pi^{-1}(r_1^{-1}(2))}\ \cdots\ \underbrace{?}_{\pi^{-1}(r_1^{-1}(n))} \quad (\pi r_1)$$
$$\text{コミットメント置換}: \boxed{\pi^{-1}(\sigma^{-1}(1))}\ \boxed{\pi^{-1}(\sigma^{-1}(2))}\ \cdots\ \boxed{\pi^{-1}(\sigma^{-1}(n))} \quad \pi\sigma$$
$$\downarrow$$
$$\underbrace{?}_{\sigma(r_1^{-1}(1))}\ \underbrace{?}_{\sigma(r_1^{-1}(2))}\ \cdots\ \underbrace{?}_{\sigma(r_1^{-1}(n))} \quad (\sigma^{-1}r_1)$$
$$\boxed{1}\ \boxed{2}\ \cdots\ \boxed{n}$$

(5) プレイヤー ID とギフト ID をもう一度ペアにし Pile-Scramble シャッフルを行う．

(6) プレイヤー ID のみを表側にする．プレイヤー ID に対応するプレイヤーがペアになっているギフト ID を確認する．各プレイヤーは確認したギフト ID が示すプレイヤーへとギフトを贈る．

このプロトコルは手順 (4) においてコミットメント置換 $\pi\sigma$ を表向きにしている．この置換は Pile-Scramble シャッフル π により，元の置換 σ とは確率的に独立な置換となっているため置換の型の秘匿性は保たれている．

もしコミットメント置換 σ が不動点のない各型の代表元から割合に応じて既に選択されているならば，このプロトコルと組み合わせることにより，不動点を持たない一様ランダムな置換を生成できることになる．

7. むすび

本稿では 筆者らが CSS 2018 にて提案したプロトコルについて, 一様ランダムに最長サイクルを出力することを証明し, 巡回シフトを行う範囲を変更することによって, 最長サイクル以外の様々なサイクル長からなる置換を生成できるプロトコルを新たに提案した．また, 巡回シャッフルを用いる手法として 2016 年に Ibaraki と Manabe が提案したプロトコルについて, 生成される巡回置換の平均サイクル長の解析を行った．さらに, 2018 年に Hashimoto らが提案した Permutation Randomizing Protocol を紹介し，同様な置換生成が可能であるプロトコルを提案した．

参考文献

[1] Miklós Bóna, “A Walk Through Combinatorics: An Introduction to Enumeration and Graph Theory (3rd Edition),” World Scientific Publishing Co, Inc, May 2011.
[2] C. Crépeau and J. Kilian, “Discreet solitary games,” Proc. CRYPTO ’93, Lecture Notes in Computer Science, vol. 773, pp. 319–330, Springer-Verlag, 1994.
[3] Hannah Fry: The Problems with Secret Santa - Numberphile(オンライン), 入手先 ⟨https://www.youtube.com/watch?v=5kC5k5QBqcc,⟩ (参照 2019-04-10)
[4] Yuji Hashimoto, Koji Nuida, Kazumasa Shinagawa, Masaki Inamura, Goichiro Hanaoka, “Toward Finite-runtime Card-based Protocol for Generating a Hidden Random Permutation without Fixed Points, ” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol.E101-A, no.9, pp.1503-1511, 2018.
[5] Takuya Ibaraki and Yoshifumi Manabe, “A More Efficient Card-Based Protocol for Generating a Random Permutation Without Fixed Points,” Third International Conference on Mathematics and Computers in Sciences and in Industry, pp.252-256, 2016.
[6] Rie Ishikawa, Eikoh Chida, and Takaaki Mizuki, “Efficient Card-Based Protocols for Generating a Hidden Random Permutation without Fixed Points,” Unconventional Compulation and Natural Computation (UCNC 2015), Lecture Notes in Computar Science, Springer-Verlag, vol.9252, pp.215-226, 2015.
[7] Takaaki Mizuki and Hideaki Sone, “Six-card secure AND and four-card secure XOR,” Proc. Frontiers in Alogorithmics(FAW 2009), Lecture Notes in Computer Science, vol.5598, pp.358-369, Springer-Verlag, 2009.
[8] Takaaki Mizuki, Isaac Kobina Asiedu, and Hideaki sone, “Voting with a logarithmic number of cards,”Proc. Unconventional Computation and Natural Computation (UCNC 2013), Lecture Notes in Computer Science, Springer-Verlag, vol. 7956, pp.162-173, 2013.
[9] 齋藤 敬宏, 千田 栄幸, 水木 敬明, “プレゼント交換に適したシンプルなカードベース置換生成,” コンピュータセキュリティシンポジウム (CSS2018) 論文集, 4A1-2, pp.1-6, 2018.