あみだくじのGarbled Circuitによる構成

増井 孝之[1,a)]　森田 光[1]

概要：本稿では Yao に基づく Garbled Circuit[1] を, あみだくじに適用することを試みた. あみだくじのシャッフル部分をマルチプレクサとその制御部に分け, それぞれ Garbled Circuit を適用して秘密計算する. 具体的には, Kolesnikov らに基づく freeXOR[2] などを適用してテーブルサイズを削減する.

キーワード：Garbled Circuit，あみだくじ，freeXOR

A construction of garbled circuits for Amidakuji

Takayuki Masui[1,a)]　Hikaru Morita[1]

Abstract: In this paper,The authors have tried to apply Yao's garbled circuit[1] to Amidakuji. To do secure computation, divide the shuffle part of Amidakuji into a multiplexer and its control unit, and apply a garbled circuit to each part. In detail, reduce table size by using Kolesnikov's freeXOR[2].

Keywords: Garbled Circuit, Amidakuji, freeXOR

1. まえがき

多人数での順番決めや抽選を実行する方法の 1 つとしてあみだくじが有効であり, 結果は公平であることが求められる. 一般に参加者全員が対面している場合は以下のようにして公平なあみだくじを構成できる.

(1) 参加者の中から代表者を選び, 代表者が紙にあみだくじに参加する人数分縦線を平行に引く.
(2) 下側の線端に代表者があみだくじの結果である順位を書き, 紙を折り曲げて結果を隠す.
(3) 代表者は隣り合う縦線の間に横線を自由に引く.
(4) 公平性を期すために, 代表者以外の参加者も横線を自由に引く.
(5) じゃんけんなどで縦線の上側の線端を選ぶ順番を決定する.

上記の手法による 4 人の順位を決めるあみだくじの構成例を図 1 に示す. 点線より下は実際には折り曲げられており, 参加者からは見えない状態である.

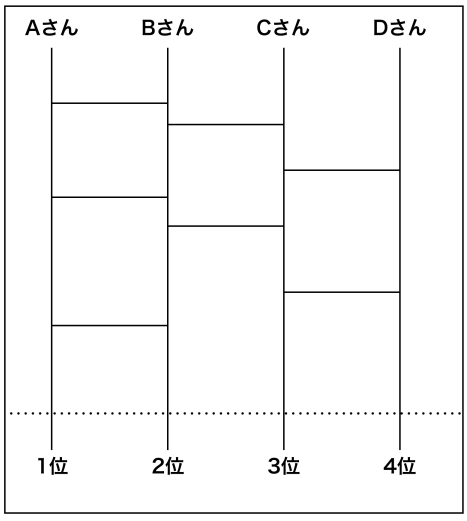


図 1　4 人の順位を決めるあみだくじの例

通信ネットワーク上であみだくじを行う場合, 一般的に参加者の人数に応じて信頼できる第三者機関があみだくじを構成する必要がある.

[1] 神奈川大学大学院
Graduate School of Kanagawa University
[a)] r201970122af@jindai.jp

本稿では秘密計算を実現する手法の 1 つである Yao に基づく Garbled Circuit[1] を用いて, 第三者機関の導入が必要がないあみだくじの構成法を提案する.

本稿の構成は, 以下の通りである. 2 節では本稿の思想を, 3 節ではあみだくじのシャッフル部分である制御部とマルチプレクサの構成法と, Garbled Circuit の適用法について述べ, 4 節で考察を行い, 5 節でまとめる.

2. 本稿の思想

本稿の思想は以下の 3 つである

- 多人数での順位決めや抽選は, 特定の機関が参加者に見えない形で結果を決めるのではなく, 参加者も結果に関与し, 納得できる方式が必要である
- 参加者全員が自由にシャッフリングの操作が可能なあみだくじに特化した Garbled Circuit の構成
- あみだくじの構成に FFT で使われるバタフライ演算の考えを用いることによる, ローカルな部分からマクロに変換

3. 提案手法:あみだくじの Garbled Circuit による構成

提案手法では, あみだくじの横線部分をシャッフリング部 (以降 $SHUF$ 部) に置き換えた後に, 各論理ゲートを Garbled Circuit に変換する. ここでは参加人数が $N = 2^n$ 人の場合のあみだくじの構成を示す. $SHUF$ 部とは参加者 i の ID(識別子) である $X_i, i \in \{1, 2, ..., N\}$ を入れ替える論理回路である. 参加人数が 2^n 人なので, X_i は n ビット長となる. FFT で使われるバタフライ演算を基にしたシャッフリングを行うため, $SHUF$ 部は $\frac{N}{2}\log_2 N$ 個となる. Z_i はあみだくじにより入れ替えた結果の ID である.

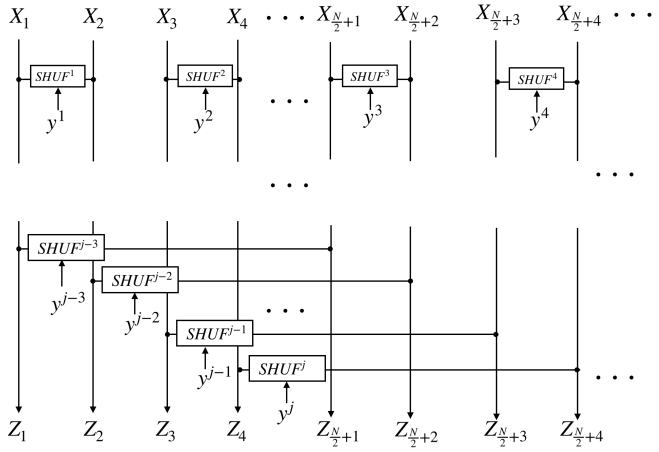


図 2 参加者 N 人のあみだくじの構成例

$SHUF$ 部はそれぞれ独立しており, 識別するために $SHUF^j, j \in \{1, 2, ..., \frac{N}{2}\log_2 N\}$ のように書く. 図 3 は隣り合う ID をシャッフリングする $SHUF$ 部の構成図であり, 制御部 (以降, $CONT$ 部) とマルチプレクサ部 (以降, MUX 部) に分けて構成される. $CONT$ 部の入力は $y^j = (y^j_N, y^j_{N-1}, ..., y^j_1), y^j_i \in \{0, 1\}$ であり, 出力は MUX 部の制御信号となる $c^j \in \{0, 1\}$ である. y^j は MUX 部に入力された ID を入れ替えるかどうかの参加者全員が行う投票のようなものである. 例えば, X_4 の所持者は y^j_4 を $SHUF^j$ の制御部である $CONT^j$ に入力をする. ここでは MUX^j 部の入力は X_i,X_{i+1}, c^j とし, c^j が 1 の場合は ID の入れ替えを行う.

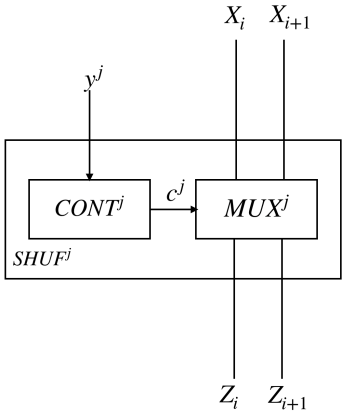


図 3 $SHUF^j$ 部の構成

$CONT$ 部は XOR ゲートがツリー状に接続された論理回路によって構成される. 図 4 は参加人数が N 人の場合の $CONT^j$ 部である. 入力は参加者全員による X_i と X_{i+1} を入れ替えるかどうかの投票である y^j であり, 出力は MUX^j 部の制御信号 c^j である.

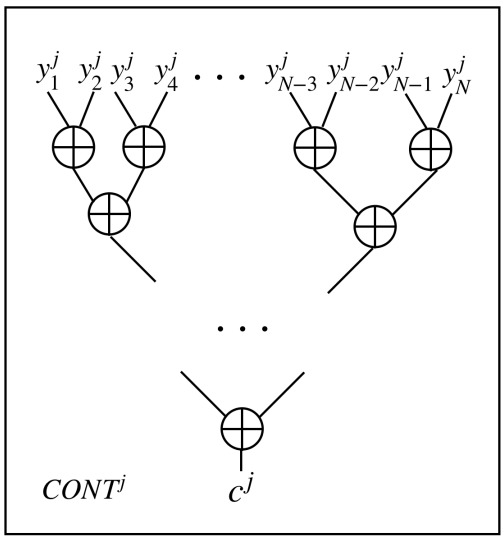


図 4 $CONT^j$ 部の構成

マルチプレクサ部は Kolesnikov らが提案した手法を用いる [2,3]. 図 5 は X_i と X_{i+1} を入れ替える MUX^j の構成図である. ただし, ここでの $\wedge$ と $\oplus$ はビットごとの演算を意味する.

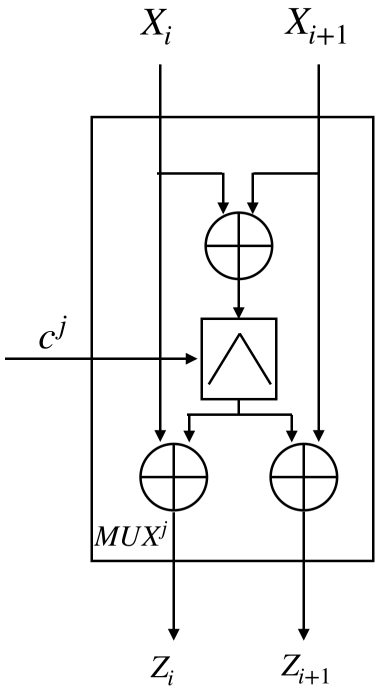
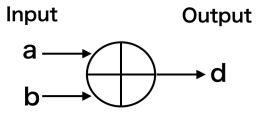


図 5 MUX^j 部の構成

各論理ゲートを Yao に基づく Garbled Circuit に変換する手法を図 6 のような 2 つの入力ワイヤ a,b と 1 つの出力ワイヤ d を持つ XOR ゲートを例に説明する.



真理値表

a	b	d
0	0	0
0	1	1
1	0	1
1	1	0

図 6 XOR ゲートとその真理値表

(1) 入力ワイヤー a と b, 出力ワイヤー d のそれぞれのワイヤーの 0 と 1 に対応する乱数 K を生成する. 以降, ワイヤー a の 0 に対応する乱数を K_0^a のように書く.

(2) それぞれの乱数を用いて Garbled XOR Gate を生成する. 具体的には表 1 のように 4 つの暗号文から構成される Garbled Computation Table(以降 GCT) を構成する. 暗号文 $E_K(m)$ は鍵 k で復号でき, 平文 m を入手できることを意味する.

表 1 Garbled XOR Gate

input wire a	input wire b	output wire d	GCT
K_0^a	K_0^b	K_0^d	$E_{K_0^a}(E_{K_0^b}(K_0^d))$
K_0^a	K_1^b	K_1^d	$E_{K_0^a}(E_{K_1^b}(K_1^d))$
K_1^a	K_0^b	K_1^d	$E_{K_1^a}(E_{K_0^b}(K_1^d))$
K_1^a	K_1^b	K_0^d	$E_{K_1^a}(E_{K_1^b}(K_0^d))$

AND ゲートも同様の手法で Garbled Circuit に変換可能である.

Garbled Circuit の計算は, 例えば, 入力となる乱数 K_0^a,K_1^b が手に入ると, GCT の $E_{K_0^a}(E_{K_1^b}(K_1^d))$ だけが正しく復号され, Garbled XOR Gate の出力である K_1^d を手にいれることができる.

本稿の $CONT^j$ 部のように, Garbled Circuit 生成者に自分の入力である y_i^j を隠しつつ対応する乱数 $K_\sigma^{y_i^j}$,$\sigma \in \{0,1\}$ を入手する必要がある場合は, 1-out-of-2 Obvilious Transfer[4] を実行する.

4. 考察

4.1 GCT のサイズの考察

提案手法の参加人数が $N = 2^n$ 人の場合の全体の GCT のサイズについて考察する.

まず始めに, あみだくじを構成するのに必要となる論理ゲートの数について考察する. 1 つの $SHUF$ 部の $CONT$ 部の XOR ゲートは $2^n - 1$ 個, MUX 部では XOR ゲートが $3n$ 個, AND ゲートが n 個となる. 表 2 に 1 つの $SHUF$ 部を構成するのに必要なゲート数をまとめる.

表 2 1 つの $SHUF$ 部を構成するのに必要なゲートの数

	制御部	MUX	合計
XOR	$2^n - 1$	$3n$	$3n + 2^n - 1$
AND	0	n	n
合計	$2^n - 1$	$4n$	$4n + 2^n - 1$

Yao らに基づく Garbled Circuit では, λ をセキュリティパラメータとした時の, 論理ゲート 1 つあたりの GCT のサイズは 4λ ビットとなる. 表 2 より, $SHUF$ 部が $\frac{N}{2}\log_2 N$ 個のあみだくじ全体の GCT のサイズは $n2^{n-1}(4\lambda(4n+2^n-1))$ となる. これは例えば,$\lambda = 128$,$n = 4$ とすると GCT だけで $63.5KB$ となり, あみだくじの実行においての通信量の増大が懸念される. そこで, Kolesnikov らに基づく freeXOR を適用して GCT のサイズの削減を試みる.

freeXOR による Garbled XOR Gate の構成を図 6 の XOR ゲートを例に説明する.

(1) 入力ワイヤー a と b のそれぞれのワイヤーの 0 に対応する乱数 K_0^a,K_0^b を生成する.

(2) オフセットとなる乱数 R を生成する. この時,K_0^a,K_0^b と R のビット長は同じにする

(3) $K_0^d = K_0^a \oplus K_0^b$ とする

(4) $K_1^a = K_0^a \oplus R$,$K_1^b = K_0^b \oplus R$,$K_1^d = K_0^d \oplus R$ とする.

これにより XOR ゲートに関しては GCT を構成せずに $SHUF$ 部の秘密計算が可能となる．そのため, freeXOR 適用後の SHUF 部が $\frac{N}{2}\log_2 N$ 個のあみだくじ全体の GCT のサイズは $2^{n+1}n^2\lambda$ ビットとなる．$\lambda = 128, n = 4$ とすると 8.192KB となる．以上の GCT のサイズに関する考察を表 3 にまとめる．

表 3 1 つの SHUF 部を構成するのに必要な GCT のサイズ

	制御部	MUX	合計
Yao	$4\lambda(2^n-1)$	$16\lambda n$	$4\lambda(4n+2^n-1)$
Yao+freeXOR	0	$4\lambda n$	$4\lambda n$

ID のビット長が長いほどあみだくじ全体のテーブルサイズが大きくなるが, 参加人数が数十人ではせいぜい数十 KB のため, GCT のサイズに関する通信量は大きな問題にならないと考えられる．

4.2 $N \neq 2^n$ の場合

参加人数が 2 のべき乗ではない場合についてのあみだくじの構成法について考察する．図 7 は参加人数が 3 人の場合のあみだくじの構成である．

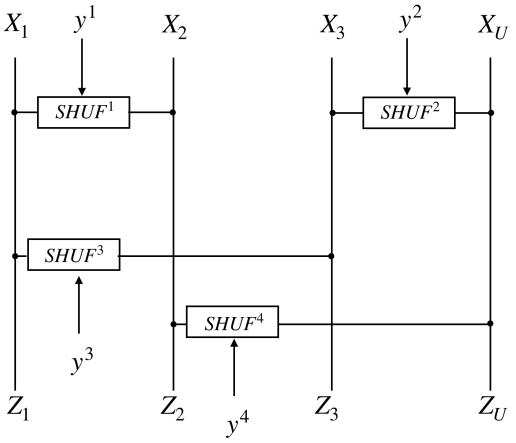
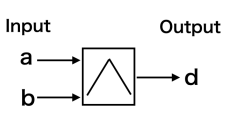


図 7 参加人数が 3 人の場合のあみだくじの構成

図 7 のように, ダミー ID である X_U を構成することで, ダミーを合わせた参加人数を 2 のべき乗にすればよい．また順番決めのあみだくじを実行する場合は, ダミーに当たる順位は欠番として扱う．

4.3 ハッシュ関数による Garbled Circuit の構成と計算量削減 [5]

本稿では Yao に基づいた Garbled Circuit を用いたが, この方式では Garbled Circuit を計算する参加者は持っている乱数が GCT のどの行に対応する鍵なのかが分からないため, Garbled 化されたゲート 1 つにつき 4 通りの計算をする必要があり, 計算量の増大が懸念される．そこで, ハッシュ関数による Garbled Circuit の構成と計算量削減について示す．ここでは, 図 8 のような 2 つの入力ワイヤ a,b と 1 つの出力ワイヤ d を持つ AND ゲートを例に説明する．



真理値表

a	b	d
0	0	0
0	1	0
1	0	0
1	1	1

図 8 AND ゲートとその真理値表

(1) 入力ワイヤー a と b, 出力ワイヤー d のそれぞれのワイヤーの 0 と 1 に対応する乱数 K を生成する．ただし, それぞれワイヤーの 2 つの乱数の最下位ビットが異なるようにする．例えば, K_0^a の最下位ビットが 1 の場合は K_1^a の最下位ビットは 0 となる．以降, 乱数の最下位ビットを取り出す関数を $LSB(.)$ と表す．

(2) それぞれの乱数を用いて Garbled AND Gate を生成する．具体的には表のように 4 つの暗号文で構成される GCT を構成する．$K_{\sigma_a}^a || K_{\sigma_b}^b$ は $K_{\sigma_a}^a$ と $K_{\sigma_b}^a$ のデータの結合であり, H はハッシュ関数, $H(x)$ は x のハッシュ値を表している．この時, $LSB(K_0^a) = 1$ の場合は GCT の上 2 行と下 2 行を, $LSB(K_0^b) = 1$ の最下位ビットが 1 の場合は 1 行目と 2 行目, 3 行目と 4 行目を入れ替える．表 4 は $LSB(K_0^a) = 1, LSB(K_0^b) = 0$ の場合である．

表 4 Garbled AND Gate

input wire a	input wire b	output wire d	GCT
K_1^a	K_0^b	K_0^d	$H(K_1^a \| K_0^b) \oplus K_0^d$
K_1^a	K_1^b	K_1^d	$H(K_1^a \| K_1^b) \oplus K_1^d$
K_0^a	K_0^b	K_0^d	$H(K_0^a \| K_0^b) \oplus K_0^d$
K_0^a	K_1^b	K_0^d	$H(K_0^a \| K_1^b) \oplus K_0^d$

これにより, 計算者は持っている 2 つの乱数 $K_{\sigma_a}^a, K_{\sigma_b}^b$ を用いて $H(K_{\sigma_a}^a || K_{\sigma_b}^b)$ を計算し, GCT の $2LSB(K_{\sigma_a}^a) + LSB(K_{\sigma_b}^b) + 1$ 行目とビットごとに XOR を計算することで, $K_{\sigma_a}^a, K_{\sigma_b}^b$ に対応する出力である $K_{\sigma_d}^d$ が得られる．

5. まとめ

本稿では信頼できる第三者機関が必要ない, あみだくじの Garbled Circuit による構成法および, テーブルサイズと参加人数が 2 のべき乗ではない場合, およびハッシュ関数を用いた計算量削減の考察を示した．あみだくじのシャッフル部をマルチプレクサとその制御部に分け, それぞれに Garbled Circuit を適用して秘密計算することで, 通信ネットワーク環境での順位決めや抽選を, 参加者のプライバシーが保護された上で全員が結果に納得できるように実行が可能である．

参考文献

[1] A.C.-C. Yao, “How to generate and exchange secrets, ” FOCS, 162-167, 1986.

[2] V.Kolesnikov and T.Schneider, “Improved Garbled Circuit: Free XOR Gates and Applications”, Proc. ICALP 2008, LNCS 5126, pp.486 - 498, Springer-Verlag ,2008.

[3] V.Kolesnikov,A.-R.Sadeghi,and T.Schneider,“Improved garbled circuit building blocks and applications to auctions and computing minima”, Cryptology and Network Security (CANS’09). LNCS. Springer, Heidelberg, 2009

[4] S. Even, O. Goldreich, and A. Lempel“A randomized protocol for signing contract” CRYPTO, pp. 205 - 210, 1982.

[5] 菊池 亮,“安全なデータ活用を実現する秘密計算技術 4)Garbled circuit を用いた秘密計算と混合的構成”, 情報処理, 2018 年 9 月,Vol 59, No 10 ,pp.893 - 897, 情報処理学会