IMI研究集会, Fukuoka, Japan

デッキ分割法と アソシエーションスキーム

Internet Initiative Japan

須賀祐治

2023-05-31

Ongoing Innovation

※本発表では
非コミットメント型のみ扱います

カードプロトコル提案のポリシー（私見）

- シンプルかつ十分実装可能であること
- 他の研究領域との関連を見出すべき
- 「できそこない」にも光をあてる

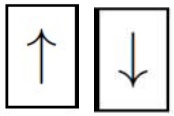
2022.10から2023.5までの貢献

- **カードプロトコルでリッカート尺度**

  CSS2022で最初に提案（一致，近い，意見が違う）

- **2者間・非コミットメント型**

  エクストラカードなし．各ユーザに２or3枚ずつ.

- **上下カードを利用** 

  名刺とか麻雀牌

リッカート尺度

- いわゆる一般的なアンケート尺度

非常に悪い　　非常に良い

1・・・2・・・3・・・4・・・5
1・・・2・・・3・・・4・・・5
1・・・2・・・3・・・4・・・5
1・・・2・・・3・・・4・・・5

非常に当てはまる	当てはまる	やや当てはまる	あまり当てはまらない	当てはまらない	全く当てはまらない
○	○	○	○	○	○

- 例えば4段階ならこういうの

そう思わない　どちらかといえばそう思わない　どちらかといえばそう思う　そう思う

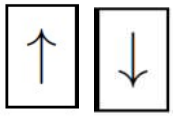
2022.10から2023.5までの貢献

- **カードプロトコルでリッカート尺度**

  CSS2022で最初に提案（一致，近い，意見が違う）

- **2者間・非コミットメント型**

  エクストラカードなし．各ユーザに２or3枚ずつ.

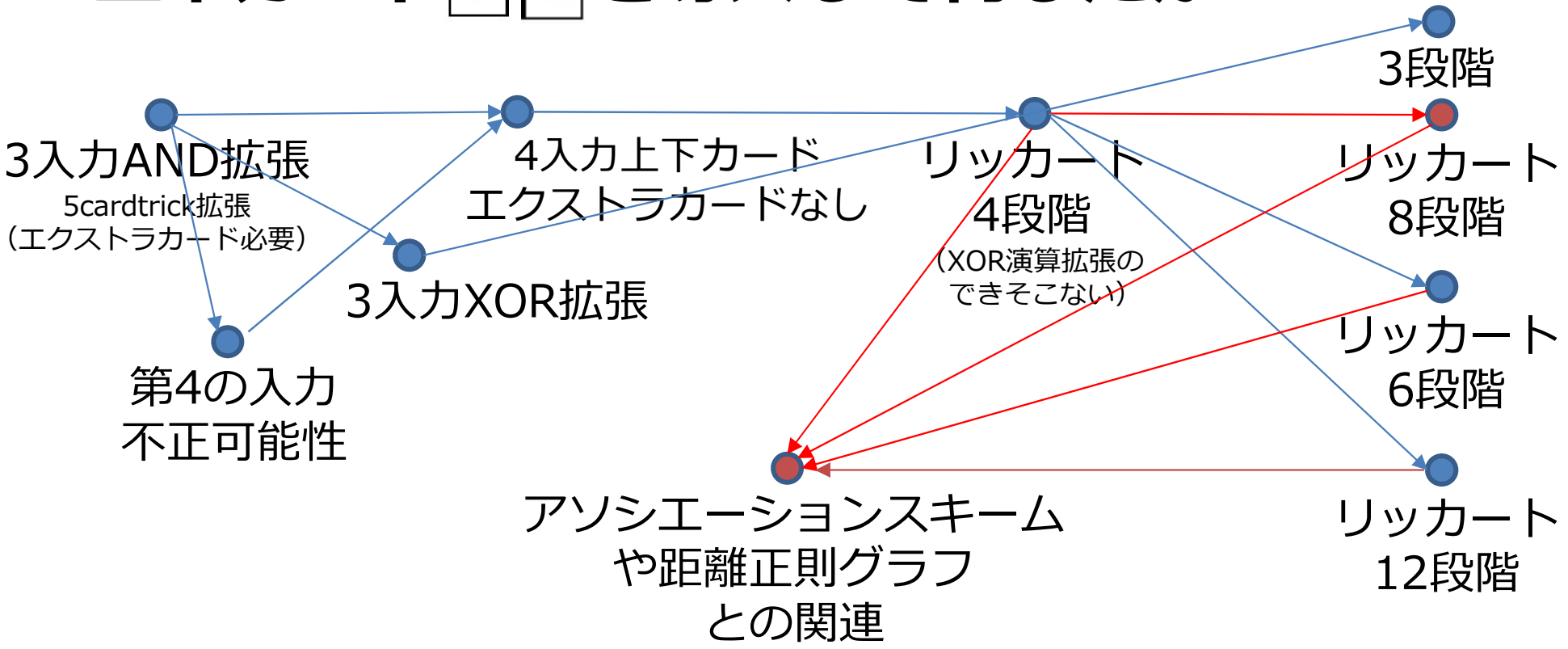
- **上下カードを利用** 

  名刺とか麻雀牌

- **アソシエーションスキーム・距離正則グラフとの関連**

ここ最近の一連の研究の流れ

- 上下カード↑ ↓を導入して何したか？



AND演算・XOR演算を実現する
カードプロトコルの用途

AND演算は「気まずくならない告白」

- 「入力1」同士ならハッピーエンド
- 出力0の場合，1入れてたとしても相手には入力がばれない

自分は好きだけど
相手にバレずに済んだ

$a \backslash b$	0	1
0	0	0
1	0	1

XOR演算は2-party2値入力「一致関数」

- 入力が同じなら出力0
- そうじゃなければ出力1(入力が違った)

この考え方は自然に
拡張できますね

マッチングに使える
例：犬好き/猫好き

$a\backslash b$	0	1
0	0	1
1	1	0

$a\backslash b$	i_1	i_2
i_1	0	1
i_2	1	0

2-party 多値入力「一致関数」

- 入力が同じなら出力0
- そうじゃなければ出力1(入力が違った)

$a \backslash b$	i_1	i_2	i_3	i_4
i_1	0	1	1	1
i_2	1	0	1	1
i_3	1	1	0	1
i_4	1	1	1	0

例えば
4値入力

上下カードと上下シャッフル

上下カード $\boxed{\uparrow}\boxed{\uparrow}\cdots\boxed{\uparrow}$

- 1種類のカード束を利用（名刺，麻雀牌）
- 裏面は上下対称の柄（UNOはダメ）
- エンコーディングルール（1枚で1ビット表現）

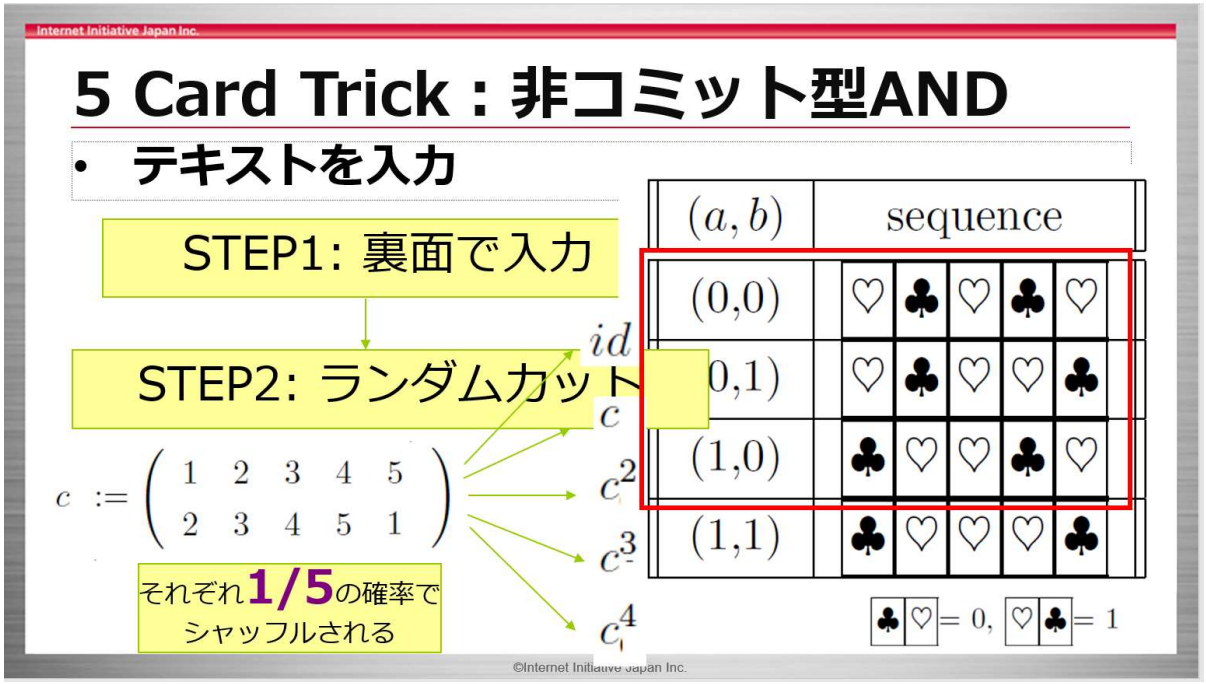
$$\boxed{\downarrow} = 0,\ \boxed{\uparrow} = 1$$

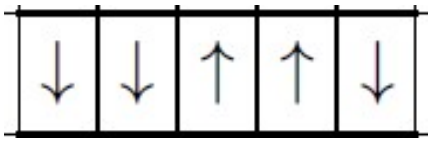
- 他にも（SCIS2022で提案した3通りの入力）

2枚で $\boxed{\downarrow}\boxed{\uparrow}(=0),\ \boxed{\uparrow}\boxed{\downarrow}(=1)\ \boxed{\uparrow}\boxed{\uparrow}(=\theta)$

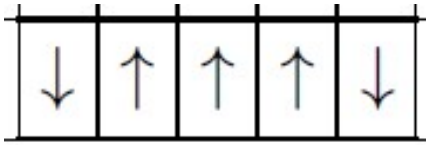
上下カードの特性を見ておきたい

- 5 Card Trickと同じく「一緒くた」にできるか？



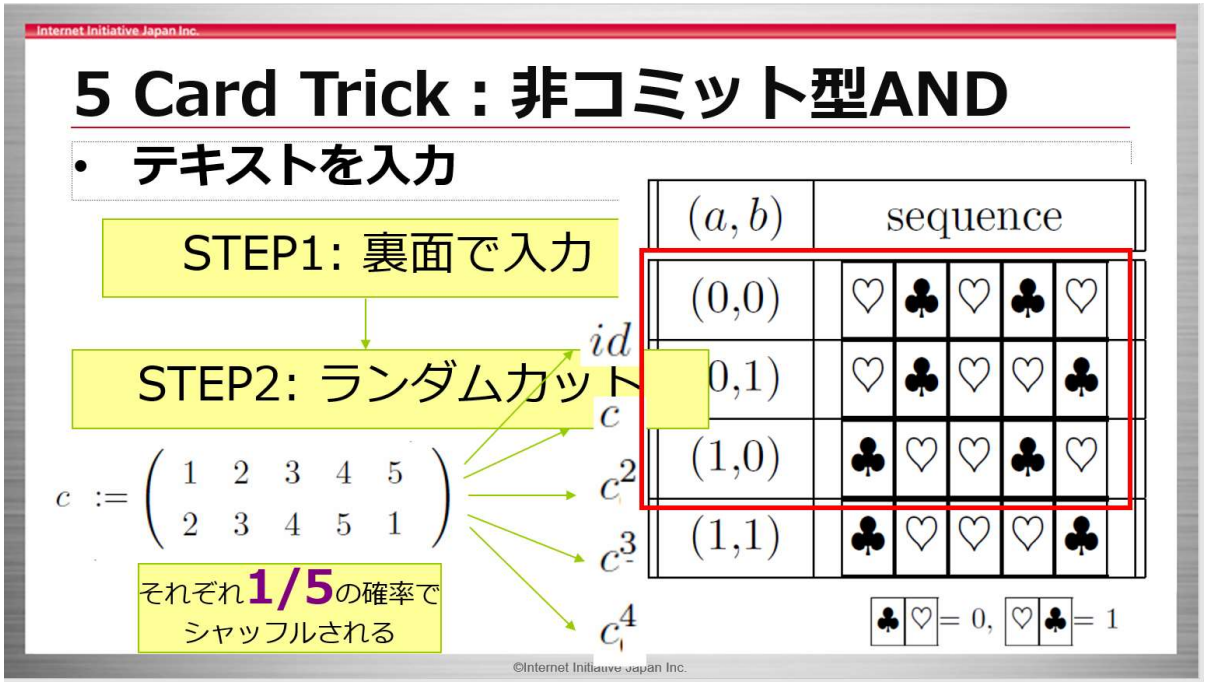


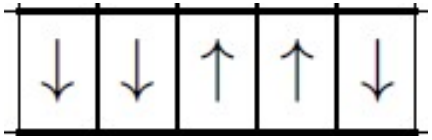
これら2つを
（シャッフル後に）
同一視できるようにしたい



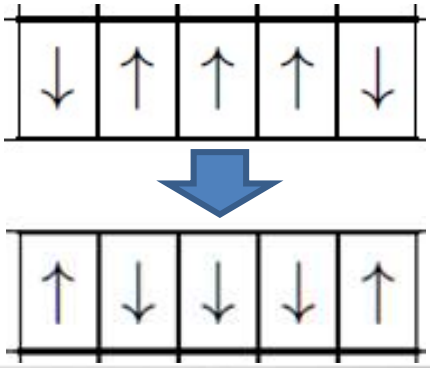
上下カードの特性を見ておきたい

- 5 Card Trickと同じく「一緒くた」にできるか？



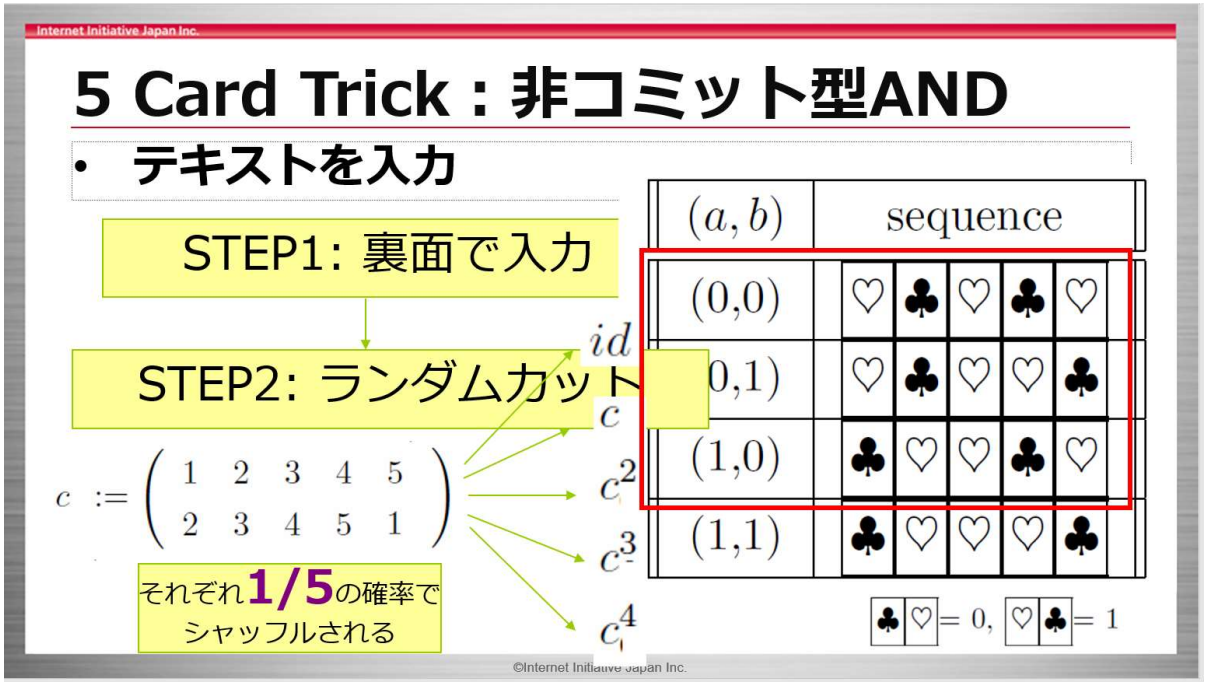


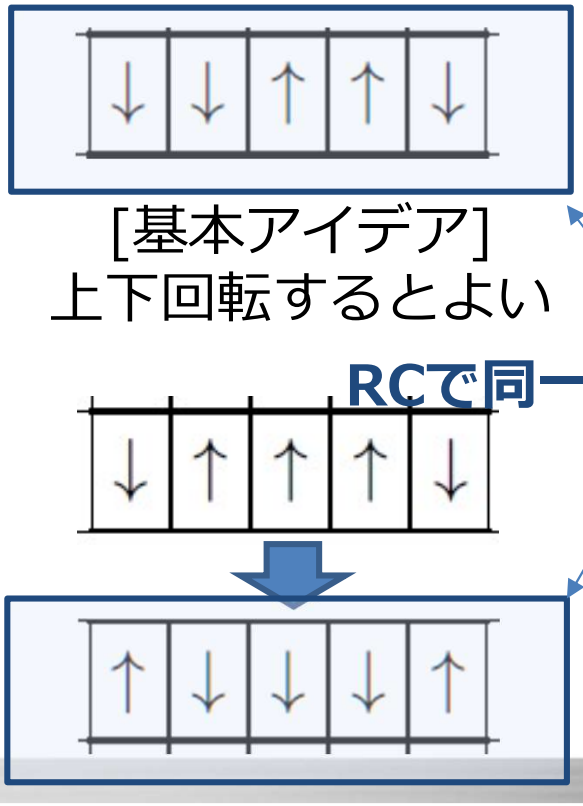
[基本アイデア]
上下回転するとよい



上下カードの特性を見ておきたい

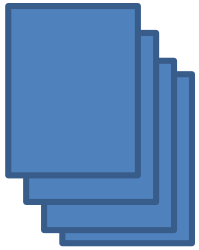
- 5 Card Trickと同じく「一緒くた」にできるか？



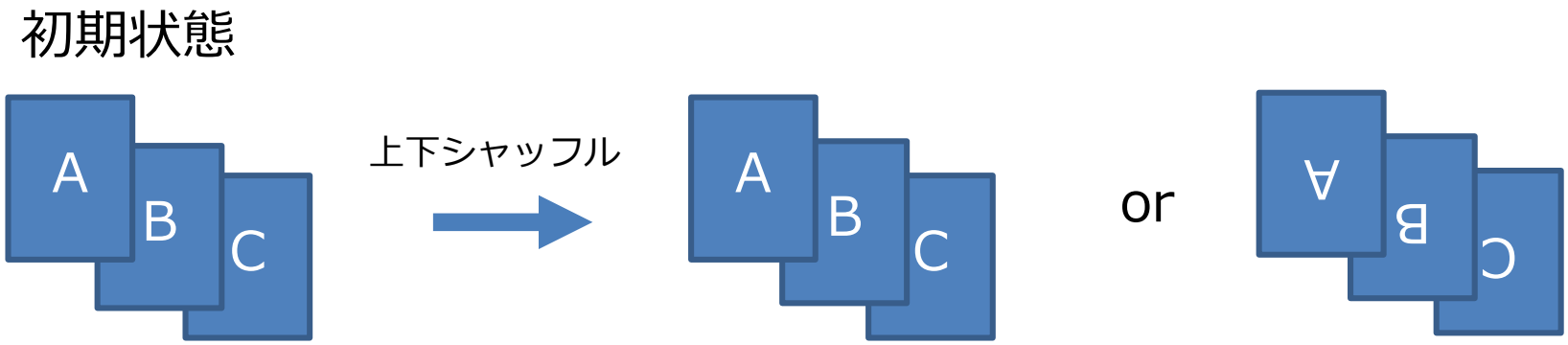


上下シャッフル

- 「上下カードの束」を回転させる処理



- 2通りの結果が得られる

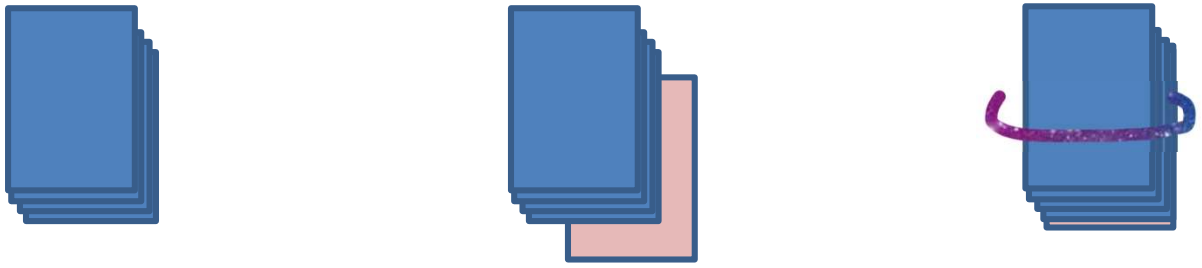


上下シャッフルの実装例（これだけじゃない）

- カード束の最下位のカードの表面（おもてめん）を秘匿して輪ゴムで留めて**放り投げる**

上下シャッフルの実装例（これだけじゃない）

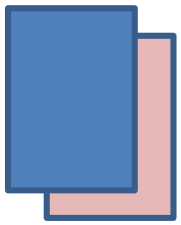
- カード束の最下位のカードの表面（おもてめん）を秘匿して輪ゴムで留めて**放り投げる**

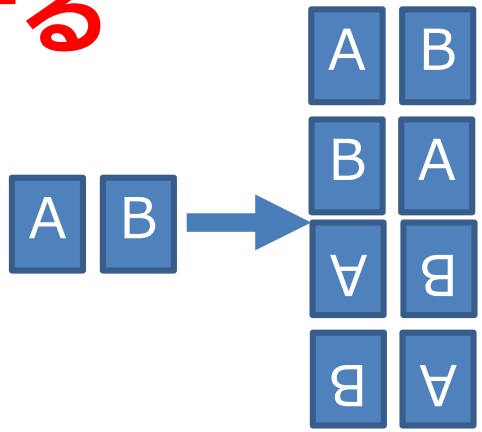


- 上下関係がランダムで入れ替わる（2通り）

上下シャッフルの実装例（これだけじゃない）

- カード束の最下位のカードの表面（おもてめん）を秘匿して輪ゴムで留めて**放り投げる**



- 2枚だと表面と表面を合わせて輪ゴムで留めて放り投げればよい（ただし左右の位置関係も変わる＝ランダムカットもやってくれる；4通りの変化）

“上下カード版” Three Card Trick

- Five Card Trickと
  同じく真ん中に
  エクストラカードを置く
- AND演算を実現

$$\boxed{\downarrow} = 0, \quad \boxed{\uparrow} = 1$$

(a,b)	sequence
(0,0)	↓ ↑ ↓
(0,1)	↓ ↑ ↑
(1,0)	↑ ↑ ↓
(1,1)	↑ ↑ ↑

Antonio Marcedone, Zikai Wen, and Elaine Shi, Secure Dating with Four or Fewer Cards, Cryptology ePrint Archive, Paper 2015/1031.

ランダムカット後も [↑] 3枚並ぶのは(1,1)の場合のみ

“上下カード版” Three Card Trick

- **Five Card Trickと同じく真ん中にエクストラカードを置く**
- **AND演算を実現**

$$\boxed{\downarrow} = 0, \quad \boxed{\uparrow} = 1$$

本質は，これらを一緒くたにできるかどうか

(0,0)	↓	↑	↓
(0,1)	↓	↑	↑
(1,0)	↑	↑	↓
(1,1)	↑	↑	↑

Antonio Marcedone, Zikai Wen, and Elaine Shi, Secure Dating with Four or Fewer Cards, Cryptology ePrint Archive, Paper 2015/1031.

ランダムカット後も [↑] 3枚並ぶのは(1,1)の場合のみ

真ん中のエクストラカードを外すと

(a,b)	sequence
(0,0)	↓ ↑ ↓
(0,1)	↓ ↑ ↑
(1,0)	↑ ↑ ↓
(1,1)	↑ ↑ ↑

3 Card Trick (AND)

真ん中のエクストラカードを外すとXOR

(a,b)	sequence		
(0,0)	↓	↑	↓
(0,1)	↓	↑	↑
(1,0)	↑	↑	↓
(1,1)	↑	↑	↑

3 Card Trick (AND)

(a,b)	sequence	
(0,0)	↓	↓
(0,1)	↓	↑
(1,0)	↑	↓
(1,1)	↑	↑

(XOR)

$\boxed{\downarrow} = 0, \quad \boxed{\uparrow} = 1$

問題：2-party多値入力の一致関数の実現

- 入力が同じなら出力0
- そうじゃなければ出力1(入力が違った)

単純にXORを拡張して
これが実現できると
うれしいなと
思うのが自然

$a \backslash b$	i_1	i_2	i_3	i_4
i_1	0	1	1	1
i_2	1	0	1	1
i_3	1	1	0	1
i_4	1	1	1	0

[追加スライド]XOR2回とAND1回で実現可能

- **Aさん a_1a_2（ビット表現）**
- **Bさん b_1b_2（ビット表現）**

- **$(a_1+b_1+1) \wedge (a_2+b_2+1)$**

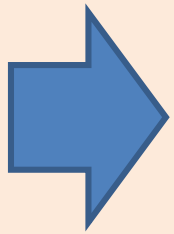
- **結局4枚で2-party 4-valuedを実現**
  AND演算時のエクストラカードは使い回す

いろいろトライしてるときの副産物

- ポリシー3:「できそこない」にも光をあてる

一致関数（対角0でそれ以外1）ではないけれど

$a\backslash b$	i_1	i_2	i_3	i_4
i_1	0	1	1	1
i_2	1	0	1	1
i_3	1	1	0	1
i_4	1	1	1	0



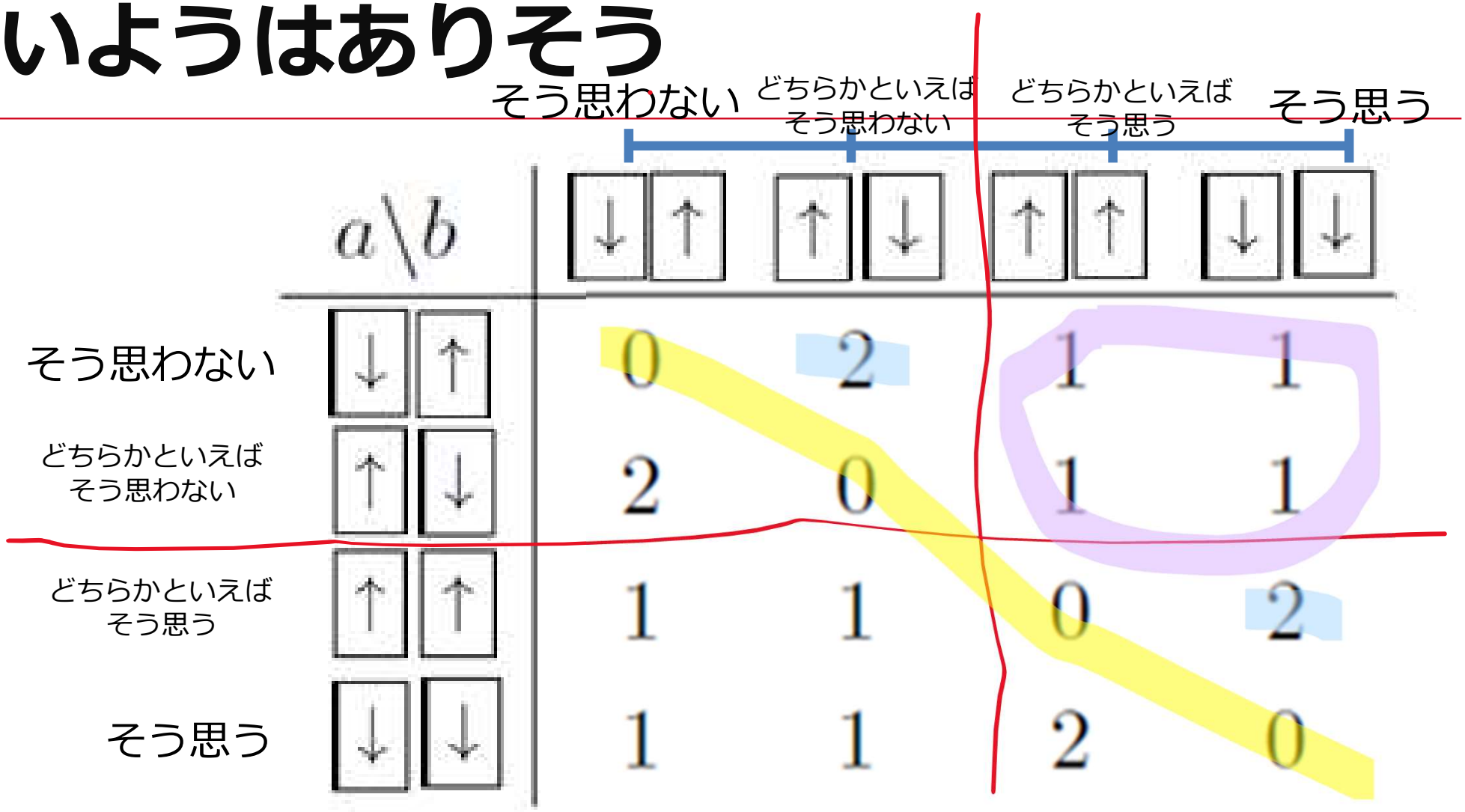
$a\backslash b$	↓↑	↑↓	↑↑	↓↓
↓↑	0	2	1	1
↑↓	2	0	1	1
↑↑	1	1	0	2
↓↓	1	1	2	0

使いようはありそう

そう思わない　どちらかといえば そう思わない　どちらかといえば そう思う　そう思う

$a \backslash b$	↓↑	↑↓	↑↑	↓↓
そう思わない ↓↑	0	2	1	1
どちらかといえば そう思わない ↑↓	2	0	1	1
どちらかといえば そう思う ↑↑	1	1	0	2
そう思う ↓↓	1	1	2	0

使いようはありそう



アソシエーションスキーム

アソシエーションスキーム(1/2)

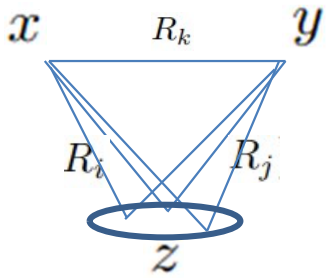
有限集合 X（位数 $n := |X|$）　　分割 $R_0, R_1, \ldots, R_d$

- (i) $R_0 = \{(x, x) \in X \times X \mid x \in X\}$
- (ii) $R_0, R_1, \ldots, R_d$ は集合 $X \times X$ の分割となる．
  $R_0 \cup R_1 \cup \cdots \cup R_d = X \times X$ かつ $i \neq j$ ならば
  $R_i \cap R_j = \emptyset$ である．

アソシエーションスキーム(2/2)

有限集合 X（位数 $n := |X|$）　　分割 $R_0, R_1, \ldots, R_d$

- (i) $R_0 = \{(x, x) \in X \times X \mid x \in X\}$
- (ii) $R_0, R_1, \ldots, R_d$ は集合 $X \times X$ の分割となる．$R_0 \cup R_1 \cup \cdots \cup R_d = X \times X$ かつ $i \neq j$ ならば $R_i \cap R_j = \emptyset$ である．



- (iii) 各 $i \in \{0, 1, \ldots, d\}$ に対し $R_i^T = Ri'$ を満たす $i' \in \{0, 1, \ldots, d\}$ が存在する．ここで $R_i^T = \{(y, x) \in X \times X \mid (x, y) \in R_i\}$ とする．
- (iv) 任意の $i, j, k \in \{0, 1, \ldots, d\}$ に対し集合 $\{z \in X \mid (x, z) \in R_i, (z, y) \in R_j\}$ の位数 p_{ij}^k は組 $(x, y) \in R_k$ の取り方に依らず i, j, k のみによって定まる．

アソシエーションスキームの行列的定義

- d+1個のn次正方行列 $\{A_i\}$ $0 \leqq i \leqq d$
- A_i は成分が0か1となる位数nの正方行列

(i) $A_0 = I$ （n 次の単位行列）

(ii) $\sum_{i=0}^{d} A_i = J$ （成分が全て1 となるn 次正方行列）

(iii) 任意の i に対して $A_i^T = A_{i'}$ となる i' が存在する

(iv) $A_i A_j = \sum_{k=0}^{d} p_{ij}^k A_k$

- **Relation matrix** $\sum_{k=0}^{d} kA_k$

0	2	1	1
2	0	1	1
1	1	0	2
1	1	2	0

- **A_0= [1000] A_1= [0011] A_2= [0100]**
  0100 0011 1000
  0010 1100 0001
  0001 1100 0010

(i) $A_0 = I$ （n 次の単位行列）

(ii) $\sum_{i=0}^{d} A_i = J$ （成分が全て1 となるn 次正方行列）

(iii) 任意の i に対して $A_i^T = A_{i'}$ となる i' が存在する

(iv) $A_i A_j = \sum_{k=0}^{d} p_{ij}^k A_k$

$A_1 * A_2 = 0 * A_0 + 1 * A_1 + 0 * A_2$

- **A_0= [1000] A_1= [0011] A_2= [0100]**
  0100 0011 1000
  0010 1100 0001
  0001 1100 0010

距離正則グラフ

- 有限集合Xをグラフの頂点集合として
- （インデックスと入れ替えて）
  アソシエーションでの関係が
  そのままグラフの距離になるケース

距離正則グラフの例

- **Relation matrix** $\sum_{k=0}^{d} kA_k$

	v_1	v_2	v_3	v_4
v_1	0	2	1	1
v_2	2	0	1	1
v_3	1	1	0	2
v_4	1	1	2	0

←これが距離を表現

v_1 v_2

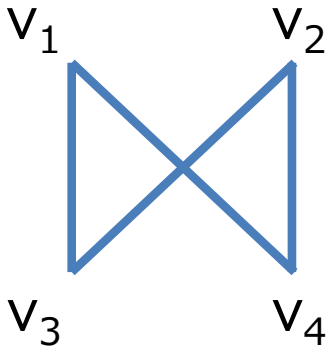
v_3 v_4

距離正則グラフの例

- **Relation matrix** $\sum_{k=0}^{d} kA_k$

	v_1	v_2	v_3	v_4
v_1	0	2	1	1
v_2	2	0	1	1
v_3	1	1	0	2
v_4	1	1	2	0

←これが距離を表現



$K_{2,2}$

さきほどのできこそない事例の再掲

$a \backslash b$	↓↑	↑↓	↑↑	↓↓
↓↑	0	2	1	1
↑↓	2	0	1	1
↑↑	1	1	0	2
↓↓	1	1	2	0

バイナリ表現に戻すと…

↓ = 0, ↑ = 1

- ちょうどハミング距離になってる

$a \backslash b$	↓↑	↑↓	↑↑	↓↓
↓↑	0	2	1	1
↑↓	2	0	1	1
↑↑	1	1	0	2
↓↓	1	1	2	0

$a \backslash b$	0 1	1 0	1 1	0 0
0 1	0	2	1	1
1 0	2	0	1	1
1 1	1	1	0	2
0 0	1	1	2	0

バイナリ表現変換

$\boxed{\downarrow}$を 0, $\boxed{\uparrow}$を 1 で表現

$a \backslash b$	0 1	1 0	1 1	0 0
0 1	0	2	1	1
1 0	2	0	1	1
1 1	1	1	0	2
0 0	1	1	2	0

ハミングスキーム H(q,n)

- アソシエーションスキームの1例

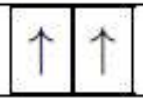
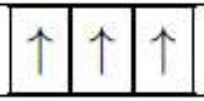
$X := {F_q}^n$ とし $d(x,y)(x,y \in X)$ をハミング距離とし $R_i := \{(x,y) \mid d(x,y) = i\}$ と分割した場合，アソシエーションスキームの条件を満たす．このアソシエーションスキームを $H(q,n)$ と記載しハミングスキームと呼ぶ．

- H(2,2): {0,1}上でn個ベクトル

$a \backslash b$	0 1	1 0	1 1	0 0
0 1	0	2	1	1
1 0	2	0	1	1
1 1	1	1	0	2
0 0	1	1	2	0

3入力8段階の場合（出力はハミング距離）

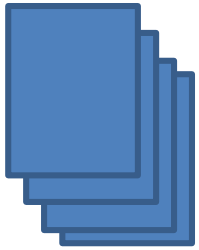
$a\backslash b$	↓↓↓	↑↑↓	↑↓↑	↓↑↑	↑↑↑	↓↓↑	↓↑↓	↑↓↓
↓↓↓	0	2	2	2	3	1	1	1
↑↑↓	2	0	2	2	1	3	1	1
↑↓↑	2	2	0	2	1	1	3	1
↓↑↑	2	2	2	0	1	1	1	3
↑↑↑	3	1	1	1	0	2	2	2
↓↓↑	1	3	1	1	2	0	2	2
↓↑↓	1	1	3	1	2	2	0	2
↑↓↓	1	1	1	3	2	2	2	0

方式	XOR (本稿 2.4 節)	4 尺度 [12]	8 尺度 [14]
カード種類			
入力パターン数	2	4	8
アソシエーション スキーム	H(2,1)	H(2,2)	H(2,3)
距離正則グラフ	K_2	$K_{2,2}$	H(2,3)
関係行列	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 2 & 1 & 1 \\ 2 & 0 & 1 & 1 \\ 1 & 1 & 0 & 2 \\ 1 & 1 & 2 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 2 & 2 & 2 & 3 & 1 & 1 & 1 \\ 2 & 0 & 2 & 2 & 1 & 3 & 1 & 1 \\ 2 & 2 & 0 & 2 & 1 & 1 & 3 & 1 \\ 2 & 2 & 2 & 0 & 1 & 1 & 1 & 3 \\ 3 & 1 & 1 & 1 & 0 & 2 & 2 & 2 \\ 1 & 3 & 1 & 1 & 3 & 2 & 2 & 2 \\ 1 & 1 & 3 & 1 & 2 & 2 & 0 & 2 \\ 1 & 1 & 1 & 3 & 2 & 2 & 2 & 0 \end{bmatrix}$

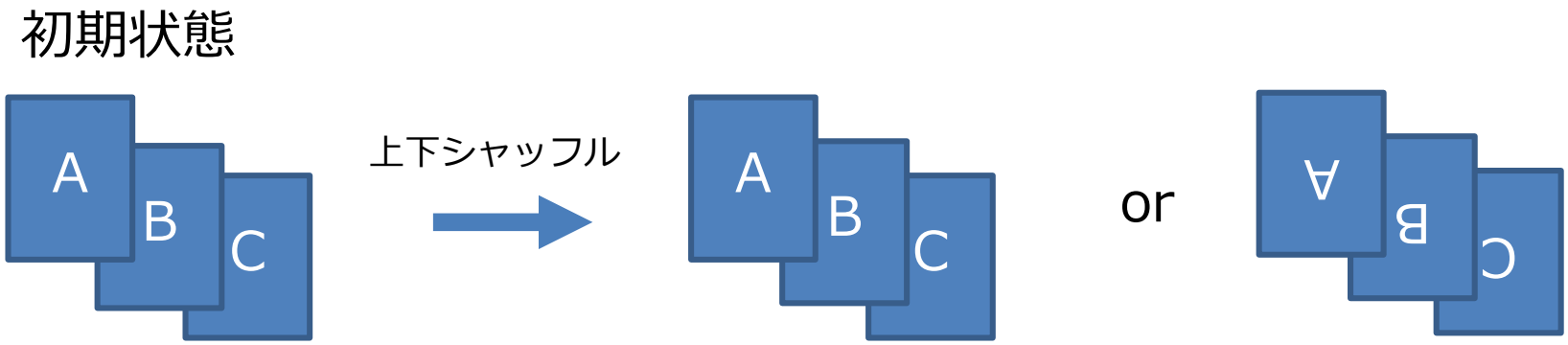
ではどうやって
カードプロトコルで
ハミング距離を出力するのか
↓
デッキ分割法

上下シャッフル(再掲)

- 「上下カードの束」を回転させる処理

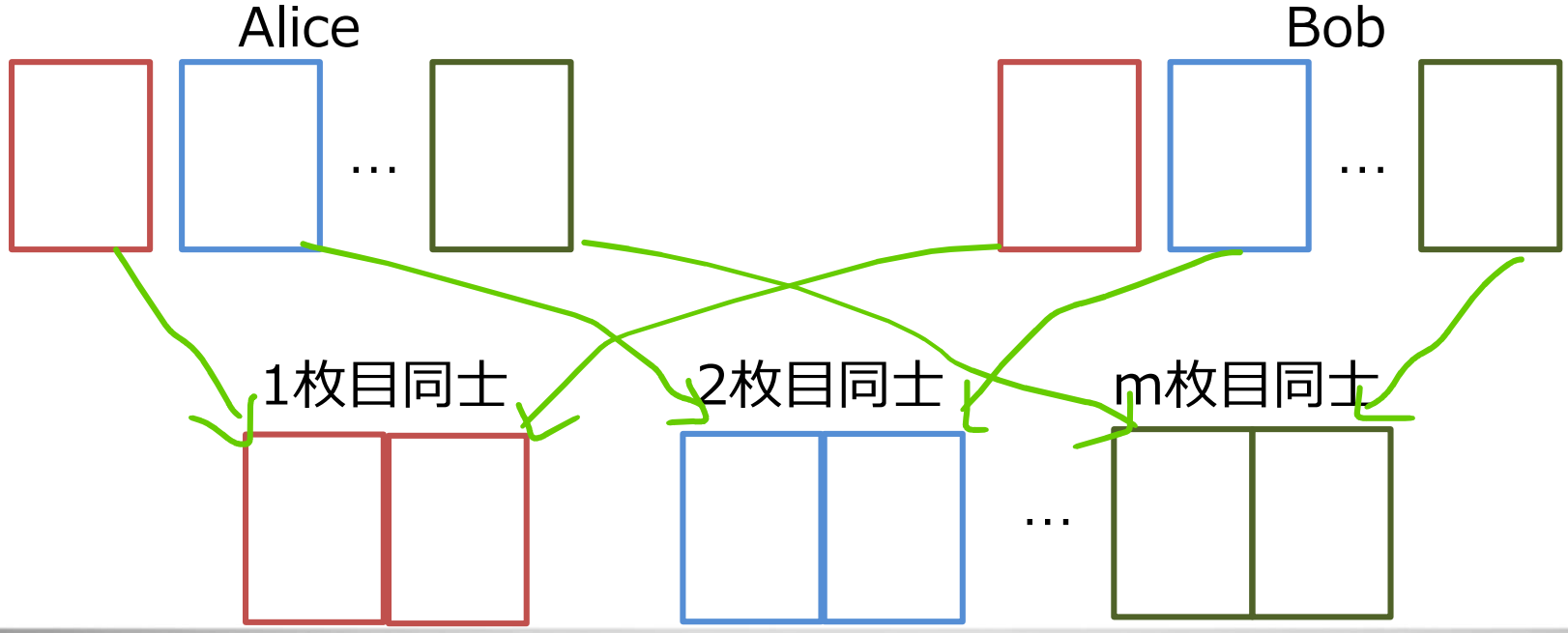


- 2通りの結果が得られる



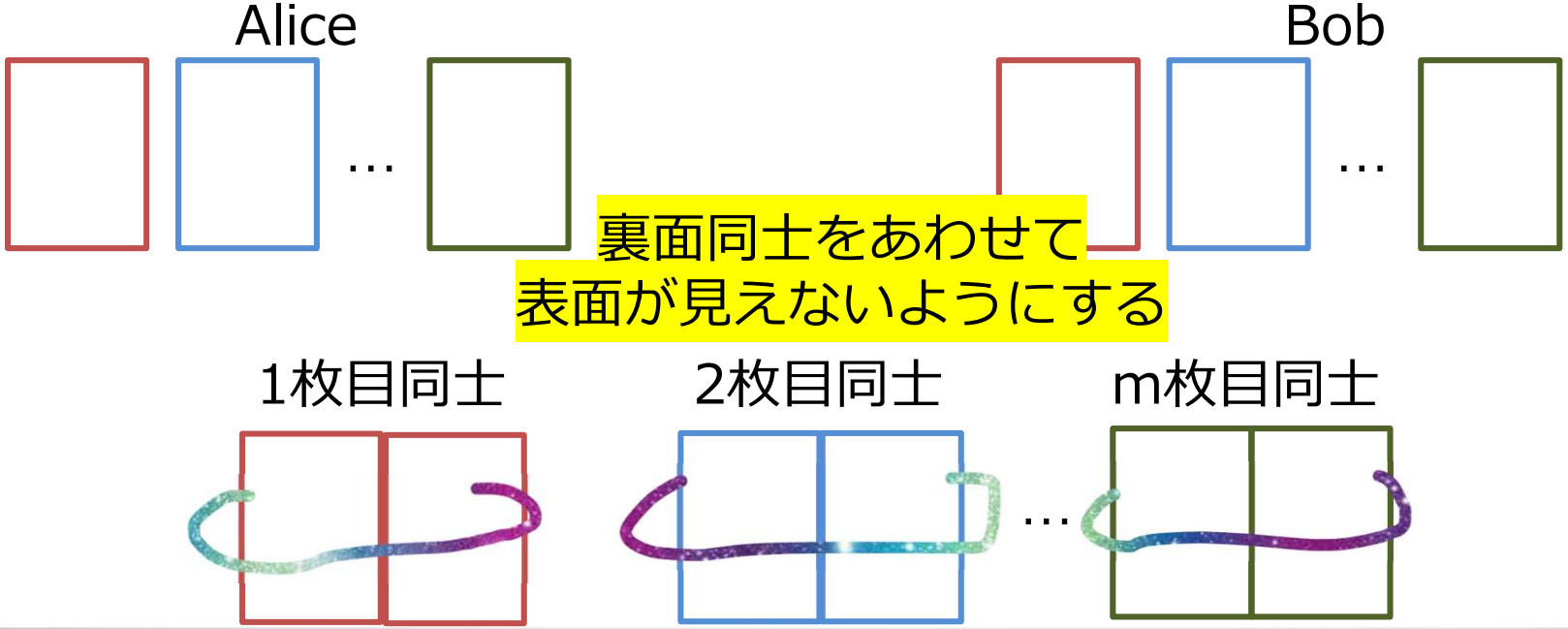
m-デッキ分割法

- 1ユーザのカード入力：m枚
- 1枚目同士，2枚目同士,…，m枚目同士の2枚を1つのデッキとして輪ゴムで止めてぶん投げる



m-デッキ分割法

- 1ユーザのカード入力：m枚
- 1枚目同士，2枚目同士,…，m枚目同士の2枚を1つのデッキとして輪ゴムで止めてぶん投げる

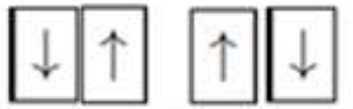


mデッキ分割法でやっていること

- **以下の3つのシャッフルを同時にやっている**
  - (1) パイルスクランブル
  - (2) 上下シャッフル
  - (3) ランダムカット

- **ポイント：各デッキは2枚しかない**

デッキをそれぞれ開示したとき

- k枚目の入力が同じ  →HW＝0
- k枚目の入力が違う  →HW＝1

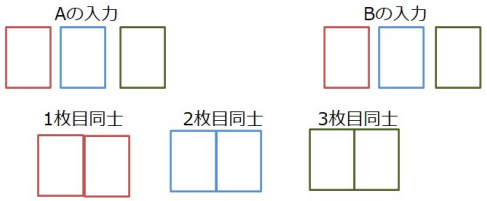
- どのビット(k枚目のk)なのかはわからない

- m個のデッキを開示すれば
  トータルでのハミング距離は算出可能

カードプロトコル提案のポリシー

- **シンプルかつ十分実装可能であること**

  シャッフルは1回放り投げるだけ



- **他の研究領域との関連を見出すべき**

  アソシエーションスキームと距離正則グラフ

- **「できそこない」にも光をあてる**

  XOR拡張の「一致関数」は実現できないが
  意味をもたせた（リッカート尺度）

ハミングスキーム適用の他の実例

- **アンケートマッチング**
- **AかBかを入力する質問をm個用意**

  例）犬好き/猫好き, indoor派/outdoor派
- **2-partyでmデッキ分割法を適用**

- **どの質問で一致したかは秘匿しつつ**
  相手と自分の嗜好がどのくらい近いか分かる
- **ある意味マッチングの並列処理・同時に処理**

問題

- Johnson scheme J(v,k) に関連する
  カードプロトコルは構成できるか？
- もしくは既存のカードプロトコルがJohnson schemeと関連している事例はあるか？

$\Omega = \{1, 2, \cdots, v\}$ とし k を $k \leq v/2$ なる自然数

X を Ω の k-部分集合全体の集合

$x, y \in X$ に対して $d(x, y) := k - |x \cap y|$

$g_i := \{(x, y) \mid d(x, y) = i\}$

$(X, \{g_i \mid i = 0, 1, \cdots, k\})$

例：J(4,2)

- 集合{1,2,3,4}からサイズ2の部分集合を考える

$\Omega = \{1, 2, \cdots, v\}$ とし k を $k \leq v/2$ なる自然数

X を Ω の k-部分集合全体の集合

$x, y \in X$ に対して $d(x, y) := k - |x \cap y|$

$g_i := \{(x, y) \mid d(x, y) = i\}$

$(X, \{g_i \mid i = 0, 1, \cdots, k\})$

	12	13	14	23	24	34
12	0	1	1	1	1	2
13	1	0	1	1	2	1
14	1	1	0	2	1	1
23	1	1	2	0	1	1
24	1	2	1	1	0	1
34	2	1	1	1	1	0

上下カード利用ではなくて

- 2色カード ❤k枚，♣(v-k)枚，2-party
- 入力：v個の集合のうち入れたいk個部分集合
- v-デッキ分割法（2枚束をv個）を適用
- J(v,k)そのもののカードプロトコルを実装可能
- Intersectionの個数を計算可能（PSIっぽい）

- Application: 複数人投票可能な選挙で投票.
  被り具合のみを共有可能.

位数6に関して

(6通りの入力を可能とするカードプロトコル)

須賀，6段階リッカート尺度入力カードベースプロトコルの構成とアソシエーションスキーム・距離正則グラフとの関連性について，4F2-3, SCIS2023, 2023.

SCIS2023での2方式を見ておきます

方式	提案方式 1	提案方式 2	提案方式 2'
カード種類			
入力パターン数	6	6	6
アソシエーション スキーム	$Z_6(0, 1+3+5, 2+4)$	$Z_6(0, 1+5, 2+4, 3)$	$Z_6(0, 1+2+4+5, 3)$
距離正則グラフ	$K_{3,3}$	6-gon	$K_{2,2,2}$
関係行列	$\begin{bmatrix} 0 & 2 & 2 & 1 & 1 & 1 \\ 2 & 0 & 2 & 1 & 1 & 1 \\ 2 & 2 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 2 & 2 \\ 1 & 1 & 1 & 2 & 0 & 2 \\ 1 & 1 & 1 & 2 & 2 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 2 & 2 & 3 & 1 & 1 \\ 2 & 0 & 2 & 1 & 3 & 1 \\ 2 & 2 & 0 & 1 & 1 & 3 \\ 3 & 1 & 1 & 0 & 2 & 2 \\ 1 & 3 & 1 & 2 & 0 & 2 \\ 1 & 1 & 3 & 2 & 2 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 2 & 1 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 2 & 1 & 1 \\ 1 & 1 & 2 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 2 \\ 1 & 1 & 1 & 1 & 2 & 0 \end{bmatrix}$

提案方式２（パイルスクランブルのみ適用）

- ３枚カード入力は2 × 3 = 6 通りのパターンに制限できる

$a \backslash b$	↑	↑	↑	↓	↓	↓
↑	0	2	2	3	1	1
↑	2	0	2	1	3	1
↑	2	2	0	1	1	3
↓	3	1	1	0	2	2
↓	1	3	1	2	0	2
↓	1	1	3	2	2	0

提案方式２’（3-デッキ分割法の適用）

- 素直なハミングウエイトで解釈し直すと…

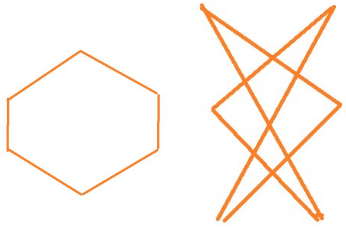
$a \backslash b$	↑	↓	↑	↓	↑	↓
↑	0	2	1	1	1	1
↓	2	0	1	1	1	1
↑	1	1	0	2	1	1
↓	1	1	2	0	1	1
↑	1	1	1	1	0	2
↓	1	1	1	1	2	0

位数6でかつSymmetric AS一覧

- 対称：2人の入力の順番で結果が変わってはいかん

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 2 & 2 & 2 & 2 \\ 1 & 0 & 2 & 2 & 2 & 2 \\ 2 & 2 & 0 & 1 & 2 & 2 \\ 2 & 2 & 1 & 0 & 2 & 2 \\ 2 & 2 & 2 & 2 & 0 & 1 \\ 2 & 2 & 2 & 2 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 & 3 & 2 & 2 \\ 1 & 0 & 1 & 2 & 3 & 2 \\ 1 & 1 & 0 & 2 & 2 & 3 \\ 3 & 2 & 2 & 0 & 1 & 1 \\ 2 & 3 & 2 & 1 & 0 & 1 \\ 2 & 2 & 3 & 1 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 2 & 2 & 2 \\ 1 & 1 & 0 & 2 & 2 & 2 \\ 2 & 2 & 2 & 0 & 1 & 1 \\ 2 & 2 & 2 & 1 & 0 & 1 \\ 2 & 2 & 2 & 1 & 1 & 0 \end{bmatrix}$$

K_6 $\overline{K_{2,2,2}}$ $\overline{K_{3,3}}$



問題

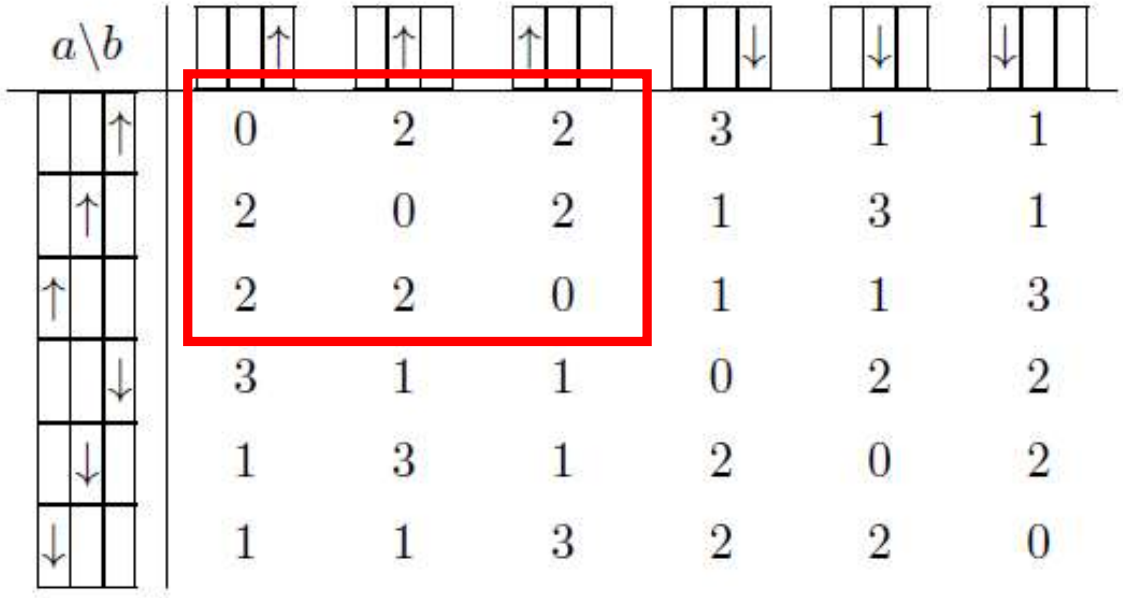
- 位数4,6のアソシエーションスキームと関連するカードプロトコルは構成できるか？

- できていないのは今のところ
  2-party 4or6-値入力一致関数だけ残っている
  （つまり距離正則グラフが完全グラフとなる
  アソシエーションスキームのみ）

ちなみに位数3のASとの関連はある

- 局所的に見ると構成可能であることはわかる
- 上下カードでなくて2種類3枚のカードでOK

位数３の対称ASは
K_3しかない



問題

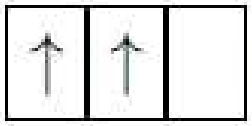
- では2枚入力 3-valued n-party 一致関数は作れますか？（上下カード・非コミットメント型）

- 参考までにこれは実装可能
  XOR-(θ, θ, θ)

$a \backslash b$	0	θ	1
0	0	θ	1
θ	θ	θ	θ
1	1	θ	0

Cross scheme

12段階へ


と２種類３枚のカード

$a\backslash b$	↑↑	↓↓	↓↑	↑↓	↑ ↑	↓ ↓	↓ ↑	↑ ↓	↑↑	↓↓	↓↑	↑↓
↑↑	0	1	2	2	3	4	3	4	3	4	3	4
↓↓	1	0	2	2	4	3	4	3	4	3	4	3
↓↑	2	2									3	4
↑↓	2	2									4	3
↑ ↑	3	4									3	4
↓ ↓	4	3									4	3
↓ ↑	3	4									3	4
↑ ↓	4	3									4	3
↑↑	3	4									2	2
↓↓	4	3									2	2
↓↑	3	4									0	1
↑↓	4	3	4	3	4	3	4	3	2	2	1	0

３枚の並べ方＝$_3C_2$=3通り
上下カードはそれぞれ２通り

3*2*2=12通りの入力が可能

そう思わない　どちらかといえば そう思わない　どちらかといえば そう思う　そう思う

$a \backslash b$	↑↑	↓↓	↓↑	↑↓	↑ ↑	↓ ↓	↓ ↑	↑ ↓	↑↑	↓↓	↓↑	↑↓
↑↑	0	1	2	2	3	4	3	4	3	4	3	4
↓↓	1	0	2	2	4	3	4	3	4	3	4	3
↓↑	2	2	0	1	3	4	3	4	3	4	3	4
↑↓	2	2	1	0	4	3	4	3	4	3	4	3
↑ ↑	3	4	3	4	0	1	2	2	3	4	3	4
↓ ↓	4	3	4	3	1	0	2	2	4	3	4	3
↓ ↑	3	4	3	4	2	2	0	1	3	4	3	4
↑ ↓	4	3	4	3	2	2	1	0	4	3	4	3
↑↑	3	4	3	4	3	4	3	4	0	1	2	2
↓↓	4	3	4	3	4	3	4	3	1	0	2	2
↓↑	3	4	3	4	3	4	3	4	2	2	0	1
↑↓	4	3	4	3	4	3	4	3	2	2	1	0

局所的に見ると（入力を制限できれば実装可能）

$a\backslash b$	↑↑	↓↓	↓↑	↑↓	↑ ↑	↓ ↓	↓ ↑	↑ ↓	↑↑	↓↓	↓↑	↑↓
↑↑	0	1	2	2	3	4	3	4	3	4	3	4
↓↓	1	0	2	2	4							
↓↑	2	2	0	1	3							
↑↓	2	2	1	0	4							
↑ ↑	3	4	3	4	0							
↓ ↓	4	3	4	3	1							
↓ ↑	3	4	3	4	2							
↑ ↓	4	3	4	3	2							
↑↑	3	4	3	4	3							
↓↓	4	3	4	3	4							
↓↑	3	4	3	4	3							
↑↓	4	3	4	3	4							

問題

- 局所的に有意義な関係を見出す一般的な方法を考えよ

- カードプロトコルにおいて
  カード入力を制限することで
  新しい意味を見いだせますか？

問題

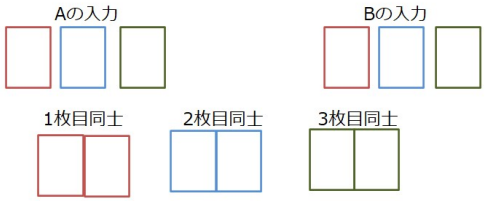
- 1枚カードを加えることによる拡張方法を一般化できますか？

- H(2,2) + ”one card“ = 12-valued new scheme

まとめに代えて：提案ポリシー（私見）

- **シンプルかつ十分実装可能であること**

  シャッフルは1回放り投げるだけ



- **他の研究領域との関連を見出すべき**

  アソシエーションスキームと距離正則グラフ

- **「できそこない」にも光をあてる**

  XOR拡張の「一致関数」は実現できないが
  意味をもたせた（リッカート尺度）

Ongoing Innovation

IIJ Internet Initiative Japan

Stay safe and healthy