カードを用いた安全な三入力多数決の計算について

西田 拓也 [*1]　林 優一 [*2]　水木 敬明 [*3]　曽根 秀昭 [*3]

[*1] 東北大学大学院情報科学研究科 曽根・水木研究室
980-8578 宮城県仙台市青葉区荒巻字青葉 6-3
nisida@s.tohoku.ac.jp
[*2] 東北大学大学院情報科学研究科
980-8579 宮城県仙台市青葉区荒巻字青葉 6-3-09
[*3] 東北大学サイバーサイエンスセンター
980-8578 宮城県仙台市青葉区荒巻字青葉 6-3

あらまし Alice，Bob，Carol の三人がある物事について賛否を問われており，賛成・反対どちらかを三人で一つだけ示さなければならないとする．そこで多数決をとることにしたが，Alice，Bob，Carol 各々は，自分の答えが賛成・反対どちらであるかを他の二人に知られたくない．また多数決の結果と異なる答えを持つ少数派がいたことが分かることも避けたいので，票数も明かしたくない．本論文では，自分の答えも，少数派の存在も明らかにせずに三人の多数決の結果だけを知ることが出来るような安全な多数決が，四枚の黒いカードと四枚の赤いカードを用いて合計八枚のカードで計算できる暗号プロトコルを与える．

Secure Three-Input Majority Computation Using a Deck of Cards

Takuya Nishida[*1]　Yu-ichi Hayashi[*2]　Takaaki Mizuki[*3]　Hideaki Sone[*3]

[*1] Sone Lab., Graduate School of Information Sciences, Tohoku University
6-3 Aramaki-Aza-Aoba, Aoba, Sendai, Miyagi 980-8578, Japan
nisida@s.tohoku.ac.jp
[*2] Graduate School of Information Sciences, Tohoku University
6-3-09 Aramaki-Aza-Aoba, Aoba, Sendai, Miyagi 980-8579, Japan
[*3] Cyberscience Center, Tohoku University
6-3 Aramaki-Aza-Aoba, Aoba, Sendai, Miyagi 980-8578, Japan

Abstract Alice, Bob and Carol are asked to determine whether they are all positive or negative to a certain project. Therefore, they have decided to take decision by majority. However, each of them does not want to let the other two players know his/her answer. In addition, they do not want to reveal the number of the votes; for example, they want to avoid getting to learn that there is a player whose answer is against the decision by majority. In this paper, we present a secure three-input majority computation protocol to resolve such a problem using eight physical cards.

1　はじめに

次のような場面を考える．Alice，Bob，Carol の三人は，ある物事について賛否を問われており，各自ある物事に対する反対あるいは賛成の答え，すなわち 1 ビットの入力を個人的に保有しているとする．これらの入力をそれぞれ $a \in \{0,1\}$，$b \in \{0,1\}$，$c \in \{0,1\}$ と表す．例え

ば，$a=0$ のとき Alice はその物事に反対であることを意味し，$a=1$ のとき賛成であることを意味する．三人まとめて賛成・反対どちらかの結論を示さなければならないとき，多数決によって決めることは一般によく行われている．しかし多数決をとる方法がしばしば問題になる場合がある．例えば一斉に自分の答えを話すとすると，各自の答えが全員に明かされてしまう．また，選挙における投票のような方法をとっても，票数が公になるので，多数決の結果と異なる答えを持つ少数派がいたかどうかは明らかになってしまう．そこで，Alice，Bob，Carol は，各自の入力 $a,b,c\in\{0,1\}$ を明らかにすることなく，多数決演算の出力 $\mathsf{maj}(a,b,c)$ だけを得たいと考えている．すなわち三人は

$$\mathsf{maj}(a,b,c)\stackrel{\text{def}}{=}\begin{cases}1 & \text{if } a+b+c\geq 2;\\ 0 & \text{if } a+b+c\leq 1\end{cases}$$

の値だけを知りたい．この関数は次の論理式

$$\mathsf{maj}(a,b,c)=(a\wedge b)\vee(b\wedge c)\vee(c\wedge a)$$

で表すこともできる．このように，入力を秘密のままにして，多数決の結果だけを得ることを**安全な多数決**と呼ぶ．

本論文では，安全な三入力多数決演算をカードを用いて実現する．すなわち，裏面が同じ $\left(\boxed{?}\right)$ である四枚の黒いカード $\boxed{\clubsuit}\boxed{\clubsuit}\boxed{\clubsuit}\boxed{\clubsuit}$ と四枚の赤いカード $\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\heartsuit}$ の合計八枚のカードを用いて三入力多数決を安全に計算する．安全な三入力多数決は暗号プロトコルの一種であり，後ろで示すように，物理的なカードの簡単な性質のみを利用して実現する．

1.1 カードを用いた暗号プロトコル

Yao [13] による成果から始まった安全な計算の研究の主流（例 [4, 11]）は，暗号プロトコルのコンピュータおよび通信ネットワークの上での実装を目指すことが通常である．それに対して，本論文が扱うカードを用いた暗号プロトコルは，小さなカード組しか必要としない．カード組は安価で携帯しやすく，電力を必要としない．そのうえカードを用いた暗号プロトコルは，どのようにプロトコルの中の演算が実現され，かつ秘密が漏れていないかが非専門家でも容易に理解できる．デジタル時代といわれる今日においてもなお，選挙が投票用紙によって行われている事実からも，安全な計算をコンピュータに頼らずに実現することは重要な意義があるといえるだろう（例 [1, 5, 9, 10]）．

本論文で用いるカードの性質を説明する．すべてのカードについて，同色 $\left(\boxed{\clubsuit}\text{か}\boxed{\heartsuit}\right)$ のカードどうしは区別がつかないものとする．またカードを裏にして置いた状態を $\boxed{?}$ で表し，裏面 $\boxed{?}$ もすべて同じで区別がつかないものとする．ブール値を扱うために，次のような符号化規則を定義する．

$$\boxed{\clubsuit}\boxed{\heartsuit}=0,\quad \boxed{\heartsuit}\boxed{\clubsuit}=1. \tag{1}$$

ある与えられたビット $x\in\{0,1\}$ について，x と同じ値（符号化規則 (1) による）を示す二枚のカードが伏せられて $\boxed{?}\boxed{?}$ と置かれているとき，その二枚を x の**コミットメント**と呼び，

$$\underbrace{\boxed{?}\boxed{?}}_{x}$$

と表す．

x のコミットメントを構成する二枚のカードの順番を入れ替えると，x の否定である $\bar{x}$ のコミットメントになる．

$$\underbrace{\boxed{?}\boxed{?}}_{x}\ \rightarrow\ \overbrace{\boxed{?}\boxed{?}}^{\rightleftharpoons}\ \rightarrow\ \underbrace{\boxed{?}\boxed{?}}_{\bar{x}}.$$

したがって安全な NOT 演算は自明である．

既に考案されているカードを用いた暗号プロトコルとして表 1 に示すように AND 演算，XOR 演算，加算器などがある．また，表 1 のプロトコルに加えて，2.3 節で紹介するコピープロトコルがある．プロトコルには出力に関して二つのタイプがある．表 1 の上から二つのプロトコル（非コミット型）は出力（$a\wedge b$ の値）を公にするが，残りの九つのプロトコル（コミット型）は出力がコミットメントで得られる．出力がコミットメントで得られるとは，符号化規則 (1) に則って，例えば

$$\underbrace{\boxed{?}\boxed{?}}_{a\wedge b}\qquad\underbrace{\boxed{?}\boxed{?}}_{a\oplus b}$$

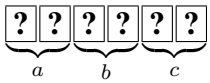
表 1: カードを用いた既存の暗号プロトコル

設計者	色数	枚数
○ 非コミット型 AND 演算		
den Boer [3]	2	5
Mizuki-Kumamoto-Sone [6]	2	4
○ コミット型 AND 演算		
Crépeau-Kilian [2]	4	10
Niemi-Renvall [10]	2	12
Stiglic [12]	2	8
Mizuki-Sone [7] (§2.2)	2	6
○ コミット型 XOR 演算		
Crépeau-Kilian [2]	4	14
Mizuki-Uchiike-Sone [8]	2	10
Mizuki-Sone [7]	2	4
○ コミット型半加算器		
Mizuki-Asiedu-Sone [5]	2	8
○ コミット型全加算器		
Mizuki-Asiedu-Sone [5]	2	10

のように出力を得られるということである．本論文で扱うプロトコルは全てコミット型である．

1.2 研究目的

本論文の目的は三つのコミットメント



を入力として与えられたとき，出力

を得られるコミット型暗号プロトコルを構築することである．前節で述べたように，我々は既にカードを用いて NOT，AND 及び XOR 演算をコミット型で安全に計算することができる．またコミットメントの複製も安全に行うことができる．したがって，あらゆる関数の計算をカードを用いて安全に計算できることは明らかである．事実，既存のプロトコル（2 章で紹介する）を用いて安全な多数決演算 $\mathsf{maj}(a,b,c)$ を簡単に実現できるが，八枚の追加カード，すなわち三つの入力コミットメント六枚と合わせて十四枚のカードが必要になる．このことは 3 章で説明する．

本論文ではこれを改良し，安全な三入力多数決演算を二枚のカードを加えるだけで，合計八枚のカードのみで実現した．その新しいプロトコルを 4 章で示す．5 章で本論文の結論を述べる．

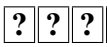
2 既存のプロトコル

この章では，“ランダム二等分割カット” と呼ばれるシャッフル手法を説明した後に，AND プロトコルとコピープロトコル [7] を紹介する．これらのプロトコルを次章で安全な多数決演算の実現のために利用する．

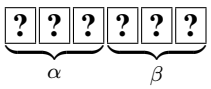
2.1 ランダム二等分割カット

ランダム二等分割カットは文献 [7] で提案された．例として六枚のカードに適用する．

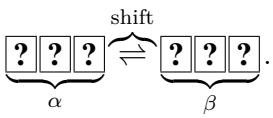
1. 六枚のカードが並んでいるとする．



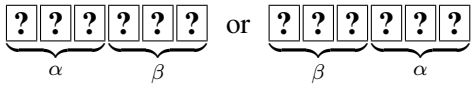
2. カード列の左半分を α，右半分を β とする．



3. α と β の位置をランダムに入れ替える．

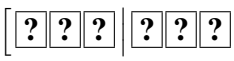


4. ランダムに入れ替えたことで，α と β の位置は最初と変わらないか，または入れ替わる．よってカード列は



のどちらかになり，どちらになる確率も等しく 1/2 である．

このようなシャッフル手法をランダム二等分割カットと呼び $[\cdot|\cdot]$ で表す．六枚のカードへのランダム二等分割カットの適用は



のように表される．

2.2 六枚のカードによる AND プロトコル

ランダム二等分割カットを用いて六枚のカードによる AND プロトコル [7] が構築できる．このプロトコルは三枚の ♣ と三枚の ♡ を用いて安全に論理積 $a \wedge b$ を計算する．

プロトコルの手順を紹介する前に，二つの操作 get と shift を定義する．二つのビット x と y の組 (x, y) について

$$\begin{aligned} \mathsf{get}^0(x,y) &= x; \\ \mathsf{get}^1(x,y) &= y; \\ \mathsf{shift}^0(x,y) &= (x,y); \\ \mathsf{shift}^1(x,y) &= (y,x) \end{aligned}$$

とする．すなわち，$\mathsf{get}^0(x,y)$ は最初のビットを返し，$\mathsf{get}^1(x,y)$ は二つ目のビットを返す．また，$\mathsf{shift}^0(x,y)$ は二つのビットをそのまま返し，$\mathsf{shift}^1(x,y)$ は順番を入れ替える．これらの記法を使うと AND 演算は

$$\begin{aligned} a \wedge b &= \mathsf{get}^a(0,b) \\ &= \mathsf{get}^a(\mathsf{shift}^0(0,b)) \end{aligned}$$

と書ける．なぜなら

$$a \wedge b = \begin{cases} 0 & \text{if } a = 0; \\ b & \text{if } a = 1 \end{cases}$$

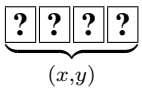
が成り立つからである．更に

$$\begin{aligned} \mathsf{get}^a(0,b) &= \mathsf{get}^{a\oplus 1}(b,0) \\ &= \mathsf{get}^{a\oplus 1}(\mathsf{shift}^1(0,b)) \end{aligned}$$

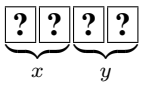
であるから，ランダムビット $r \in \{0,1\}$ に対して

$$a \wedge b = \mathsf{get}^{a\oplus r}(\mathsf{shift}^r(0,b)) \qquad (2)$$

が成り立つ．これから紹介する AND プロトコルの基本アイデアは式 (2) を基にしている．以降，二つのビット x と y について

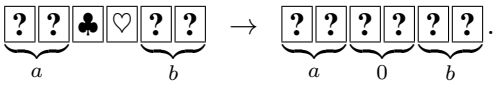


のように書かれているときは，

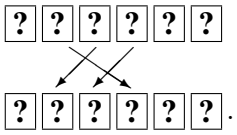


を意味しているものとする．

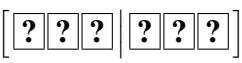
1. 六枚のカードを次のように置く．



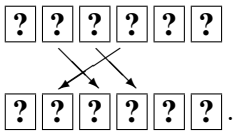
2. 次のように並び替える．



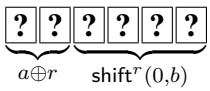
3. ランダム二等分割カットを適用する．



4. 次のように並び替える．



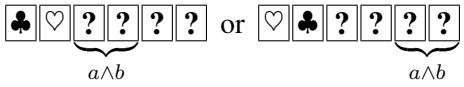
このとき，r をランダムビットとして，六枚のカードは



のように並んでいる．この r はランダム二等分割カットによって生じたものである．

5. 左端の二枚のカードをめくる．この $a \oplus r$ の値を知ることで，残る二つのコミットメントの内どちらが求める値なのかが分かる．(式 (2) を思い出してほしい，もし $a \oplus r = 0$ ならば $a \wedge b = \mathsf{get}^0(\mathsf{shift}^r(0,b))$ であり，そ

うでなければ $a \wedge b = \mathsf{get}^1(\mathsf{shift}^r(0, b))$ である.）よって $a \wedge b$ のコミットメントは



のように得られる.

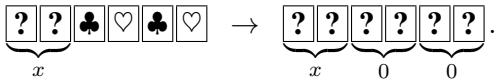
ステップ 5 で左端の $a \oplus r$ のコミットメントをめくっても，ランダムビット r により入力 a に関する情報は一切漏れていないことに注意してほしい．また，表になった二枚のカードは別の計算に再利用できる.

六枚のカードによる OR プロトコルも同じ手順で簡単に実現できる [5].

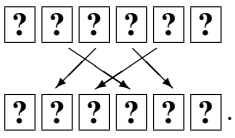
2.3 コピープロトコル

ビット x のコミットメントが与えられたとき，四枚のカードを加えることで，x のコミットメントを二つに複製することができる [7].

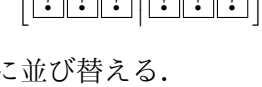
1. x のコミットメントの右側に四枚のカードを次のように並べる.



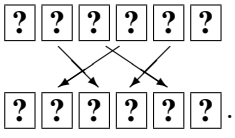
2. 次のように並び替える.



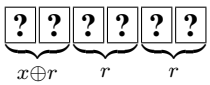
3. ランダム二等分割カットを適用する.



4. 次のように並び替える.

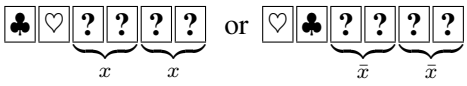


このとき，r をランダムビットとして，六枚のカードは



のように並んでいる．この r はランダム二等分割カットによって生じたものである.

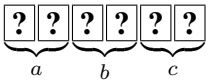
5. 左端の二枚のカードをめくる．二つに複製された x のコミットメントは



のように得られる．既に述べたとおり，NOT 演算によって $\bar{x}$ から安全に x のコミットメントを得ることができる.

3 単純な多数決プロトコル

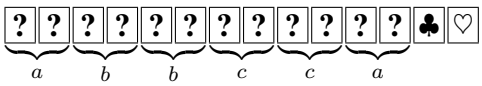
前章で紹介した既存のプロトコルを利用することで，簡単に安全な多数決演算を実現できる．具体的には，三つのコミットメント



から，多数決演算の結果のコミットメント

を得るのに，十四枚のカードが必要になる.

まず，2.3 節のコピープロトコルでは四枚のカードを加えることで，二つに複製されたコミットメントと再利用可能な二枚のカードを得ることができた．よって三つの入力コミットメントに八枚のカードを加え，コピープロトコルを三回繰り返すことで



のように各コミットメントをそれぞれ二つに複製できる．ここで

$$\mathsf{maj}(a, b, c) = (a \wedge b) \vee (b \wedge c) \vee (c \wedge a)$$

であることを思い出そう．コピープロトコルを繰り返した結果，二枚のカードが再利用可能になっているので，2.2 節で述べた AND プロトコルと OR プロトコルが上の論理式に基づいて実行できる．したがって安全な三入力多数決演算は八枚のカードを追加することで簡単に実現できる.

今示した単純な実装の代わりに，全加算器プロトコル [5]（表 1）を利用することもできる．このプロトコルは三つの入力コミットメントに四枚のカードを加えることで，和と桁上げのコミットメントを出力する．桁上げ $(a \wedge b) \vee ((a \oplus b) \wedge c)$ と $\mathsf{maj}(a,b,c)$ は等しいので，加えるカードが四枚でも安全な三入力多数決演算は実現できる．

次章では，これらをさらに改良する．すなわち，提案するプロトコルは安全な三入力多数決演算のために設計されており，非常に簡潔で，そのうえ加えるカードの枚数は二枚だけでよい．

4 多数決プロトコルの改良

この章では，効率的で簡潔な安全な三入力多数決プロトコルを構築する．前章で説明した単純な実装よりも六枚少ない（全加算器プロトコル [5] よりも二枚少ない）カード枚数で実現できる．

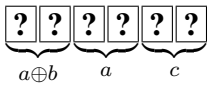
4.1 節で提案プロトコルの骨子を，詳細な手順を 4.2 節で述べる．

4.1 プロトコルの概要

三つのビット $a,b,c \in \{0,1\}$ について，もし $a = b$ ならば $\mathsf{maj}(a,b,c)$ は a に等しく，そうでなければ $\mathsf{maj}(a,b,c)$ は c に等しい．これは

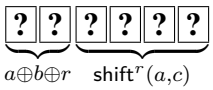
$$\begin{aligned} \mathsf{maj}(a,b,c) &= \mathsf{get}^{a\oplus b}(a,c) \\ &= \mathsf{get}^{a\oplus b}(\mathsf{shift}^0(a,c)) \end{aligned}$$

と書ける．よって提案プロトコルでは，まず



というカード列を作る．

左端の二枚のカードをめくれば，$\mathsf{get}^{a\oplus b}(a,c)$ のコミットメントの位置を知ることができるが，$a \oplus b$ の値が漏れてしまう．そこで提案プロトコルでは $a \oplus b$ の値を隠すためにランダム化を施す．ちょうど 2.2 節の AND プロトコルと同じ操作で



のようにする．r はランダムビットである．

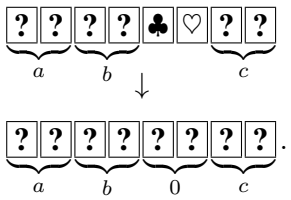
$$\mathsf{maj}(a,b,c) = \mathsf{get}^{a\oplus b\oplus r}(\mathsf{shift}^r(a,c))$$

の式に基づき，左端の二枚のカードをめくることで我々は $\mathsf{maj}(a,b,c)$ のコミットメントを安全に得ることが出来る．次節に完全なプロトコルの手順を示す．

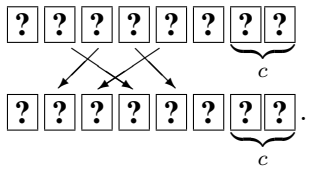
4.2 八枚のカードを用いた安全な三入力多数決プロトコル

ビット a，b，c のコミットメントと，二枚の追加カードを用いて，安全な三入力多数決プロトコルを構築する．

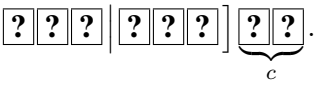
1. 八枚のカードを次のように置く．



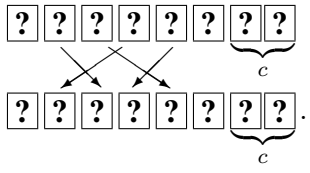
2. 次のように並び替える．



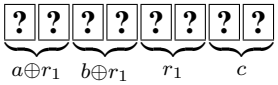
3. 左の六枚のカードにランダム二等分割カットを適用する．



4. 次のように並び替える．

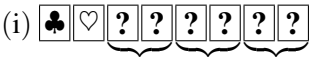


このとき，八枚のカードは

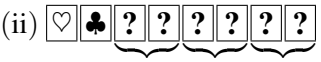


のように並んでいる．ランダムビット r_1 はステップ 3 のランダム二等分割カットによって生じたものである．

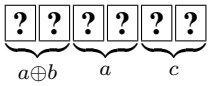
5. 左端の二枚のカードをめくることで

(i) 

または

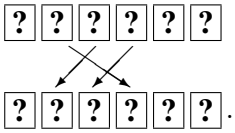
(ii) 

を得る．(ii) の場合には，NOT 演算を $\overline{a \oplus r}$ と $\bar{a}$ に施すことで，結局どちらの場合でも我々は

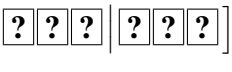


のように並んだカード列を得ることができる．ここからはこの伏せられた六枚のカードのみを用いる．

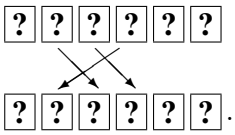
6. 次のように並び替える．



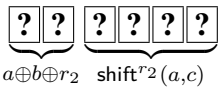
7. ランダム二等分割カットを適用する．



8. 次のように並び替える．

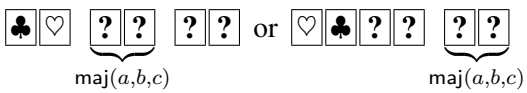


このとき，六枚のカードは



のように並んでいる．ランダムビット r_2 はステップ 7 のランダム二等分割カットによって生じたものである．

9. 左端の二枚のカードをめくる．$a \oplus b \oplus r_2$ の値を知ることによって $\mathsf{maj}(a,b,c)$ のコミットメントは



のように得られる．

5 おわりに

本論文では，八枚のカードを用いた安全な三入力多数決演算を実現するコミット型プロトコルを構築した．単純な実装では十四枚のカードを必要とするのに対して，そこから六枚のカードを減らすことができた．また，提案プロトコルは簡潔で容易に理解できる．

今後の課題として，五入力以上の安全な多数決演算も二枚のカードを追加するだけで構築できるのかどうかということと，また，六枚のカードだけで安全な三入力多数決演算を実現する非コミット型（あるいはコミット型）のプロトコルがあるのかどうかという問題等が挙げられる．

参考文献

[1] Balogh, J., Csirik, J. A., Ishai, Y. and Kushilevitz, E.: Private computation using a PEZ dispenser, *Theoretical Computer Science*, Vol. 306, pp. 69–84 (2003).

[2] Crépeau, C. and Kilian, J.: Discreet solitary games, *Proc. CRYPTO '93*, Lecture Notes in Computer Science, Vol. 773, Springer-Verlag, pp. 319–330 (1994).

[3] den Boer, B.: More efficient match-making and satisfiability: the five card trick, *Proc. EUROCRYPT '89*, Lecture Notes in Computer Science, Vol. 434, Springer-Verlag, pp. 208–217 (1990).

[4] Goldreich, O.: *Foundations of Cryptography II: Basic Applications*, Cambridge University Press, Cambridge (2004).

[5] Mizuki, T., Asiedu, I. K. and Sone, H.: Voting with a logarithmic number of cards, *Proc. UCNC 2013*, Lecture Notes in Computer Science, Vol. 7956, Springer-Verlag, pp. 162–173 (2013).

[6] Mizuki, T., Kumamoto, M. and Sone, H.: The five-card trick can be done with four cards, *Proc. ASIACRYPT 2012*, Lecture

Notes in Computer Science, Vol. 7658, Springer-Verlag, pp. 598–606 (2012).

[7] Mizuki, T. and Sone, H.: Six-card secure AND and four-card secure XOR, *Proc. Frontiers in Algorithmics (FAW 2009)*, Lecture Notes in Computer Science, Vol. 5598, Springer-Verlag, pp. 358–369 (2009).

[8] Mizuki, T., Uchiike, F. and Sone, H.: Securely computing XOR with 10 cards, *Australasian Journal of Combinatorics*, Vol. 36, pp. 279–293 (2006).

[9] Moran, T. and Naor, M.: Polling with physical envelopes: a rigorous analysis of a human-centric protocol, *Proc. EUROCRYPT 2006*, Lecture Notes in Computer Science, Vol. 4004, Springer-Verlag, pp. 88–108 (2006).

[10] Niemi, V. and Renvall, A.: Secure multiparty computations without computers, *Theoretical Computer Science*, Vol. 191, pp. 173–183 (1998).

[11] Schneider, T.: *Engineering Secure Two-Party Computation Protocols*, Springer-Verlag, Berlin Heidelberg (2012).

[12] Stiglic, A.: Computations with a deck of cards, *Theoretical Computer Science*, Vol. 259, pp. 671–678 (2001).

[13] Yao, A.: Protocols for secure computations, *Proceedings of the 23th IEEE Symposium on Foundations of Computer Science (FOCS '82)*, pp. 160–164 (1982).