Card-based Protocols Using Triangle Cards

Kazumasa Shinagawa
Tokyo Institute of Technology, Tokyo, Japan
Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
shinagawakazumasa@gmail.com

Takaaki Mizuki
Tohoku University, Sendai, Japan
tm-paper+triacard@g-mail.tohoku-university.jp

Abstract

Suppose that three boys and three girls attend a party. Each boy and girl have a crush on exactly one of the three girls and three boys, respectively. The following dilemma arises: On one hand, each person thinks that if there is a mutual affection between a girl and boy, the couple should go on a date the next day. On the other hand, everyone wants to avoid the possible embarrassing situation in which their heart is broken "publicly." In this paper, we solve the dilemma using novel cards called *triangle cards*. The number of cards required is only six, which is minimal in the case where each player commits their input at the beginning of the protocol. We also construct multiplication and addition protocols based on triangle cards. Combining these protocols, we can securely compute any function $f : \{0, 1, 2\}^n \to \{0, 1, 2\}$.

2012 ACM Subject Classification Theory of computation → Cryptographic protocols

Keywords and phrases Cryptography without computer, Secure computation, Card-based protocols, Triangle cards, Three-valued computation, Secure matching problem

Digital Object Identifier 10.4230/LIPIcs.FUN.2018.31

Funding Supported by JSPS KAKENHI Grant Numbers 17J01169 and 17K00001.

1 Introduction

Three girls, Alice, Carol, and Ellen, and three boys, Bob, Dave, and Frank, are having a good time at a party. Assume that each boy has a crush on exactly one of the three girls, and each girl has a crush on exactly one of the three boys. On one hand, everyone thinks that if there is a mutual affection between a girl and a boy the couple should go on a date the next day. On the other hand, each of the six people is too shy to announce the person they have in mind: Everyone wants to avoid the possible embarrassing situation in which their heart is broken "publicly." Are there any solutions to this dilemma?

A cryptographic technique called secure multiparty computation provides a viable solution. This enables participants holding private inputs to securely compute the value of a desired function, without revealing their input information. In order to solve the social dilemma described above, it suffices to design a secure multiparty computation protocol for the function $f : \{0, 1, 2\}^6 \to \{0, 1\}^9$:

$$f(a_0, a_1, a_2, b_0, b_1, b_2) = (c_{0,0}, c_{0,1}, c_{0,2}, c_{1,0}, c_{1,1}, c_{1,2}, c_{2,0}, c_{2,1}, c_{2,2}),$$

where $c_{i,j} = 1$ if $(a_i = j) \wedge (b_j = i)$ and $c_{i,j} = 0$ otherwise. We call this function f the $(3, 3)$-*matching function*, and refer to the problem of securely computing f as the *secure* $(3, 3)$-*matching problem*. Because it is well-known in the field of cryptography that any function can be securely computed [2], the secure $(3, 3)$-matching problem can clearly be

resolved somehow. However, conventional secure multiparty computation protocols tend to be based on a deep mathematical perspective, and hence it seems to be unlikely that all participants executing a given protocol will concretely understand its correctness and security. Because the secure $(3, 3)$-matching problem would typically arise in everyday life, such as in the party scenario mentioned above, it is desirable to have a more simple and convenient solution. Thus, this paper solicits a solution to the secure matching problem using physical cards. The cards used in this paper are novel ones, called *triangle cards*, which can easily be constructed using sheets of paper and seals.

1.1 Triangle card

In this paper, we propose a novel card called a triangle card, whose shape is a regular triangle, where its front/back sides have the same symbol (e.g., ♠). We use the following encoding rules:

In an execution of a protocol, both faces of a card can be hidden by placing seals as follows:


.

We will formally define triangle cards in Section 2.

1.2 Our results

In this paper, we design a protocol for solving the secure $(3, 3)$-matching problem using six triangle cards. We also design a protocol for any function $f : (\mathbb{F}_3)^n \to \mathbb{F}_3$.

1.2.1 Secure $(3, 3)$-matching protocol

As shown in Table 1, we construct a protocol for the secure $(3, 3)$-matching problem using six cards and six shuffles. Our protocol is efficient in terms of both the number of cards and shuffles, because a straightforward solution based on the five-card trick [1] requires 42 cards and 15 shuffles, as explained in Section 3. Our protocol is optimal in terms of the number of cards when each party submits a (tuple of) card(s) as input at the beginning of a protocol, because the number of parties in $(3, 3)$-matching is six.

1.2.2 Protocol for any function

As shown in Table 1, we design a multiplication protocol over $\mathbb{F}_3$ using four cards. Regular 3-sided cards, proposed by Shinagawa et al. [5], also enable a secure multiplication protocol over $\mathbb{F}_3$, while requiring 15 cards. Although the previous work in [5] and ours are based on different types of cards, and thus incomparable, this paper implies that triangle cards are effective for computing a function over $\mathbb{F}_3$ compared with regular 3-sided cards. Based on the previous addition protocol in [5], we also design an addition protocol for triangle cards. Because our protocols output a card with *seals*, an output card for our protocol can be used as an input card for another protocol, i.e., our protocols are *composable*. By combining our protocols, we can securely compute any function over $\mathbb{F}_3$.

Table 1 Comparison between our protocols and previous ones.

	Type of cards	Number of cards	Number of shuffles
∘ Protocol for Secure $(3,3)$-Matching Problem			
Based on [1]	♣, ♡	42	15
Ours	triangle	6	6
∘ Multiplication over $\mathbb{F}_3$			
[5]	regular 3-sided	15	2
Ours	triangle	4	2
∘ Addition over $\mathbb{F}_3$			
[5]	regular 3-sided	2	1
Ours	triangle	2	1

1.3 Related work

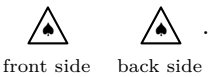
While most existing card-based protocols (cf. [1,4]) utilize a binary pair of cards $\boxed{\clubsuit}\boxed{\heartsuit}$, there are some studies based on different types of cards, such as cards with rotationally symmetric backs [3,7], polarizing cards [6], and regular n-sided polygon cards [5]. Part of our technique is motivated by this previous works. In particular, we employ a rotation shuffle [3,5] and a turning shuffle [6].

2 Triangle card

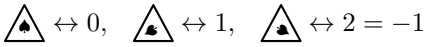
In this section, we define triangle cards and the operations used in our protocols.

2.1 Definition of triangle cards

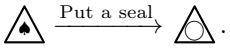
A *triangle card* is a card of regular triangular shape, whose front and back sides show the same symbol, as follows:



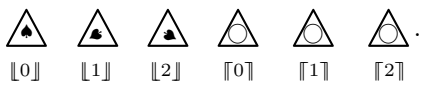
We use the following encoding rule:



The value of a card can be hidden from participating parties by placing seals on both sides, as follows:

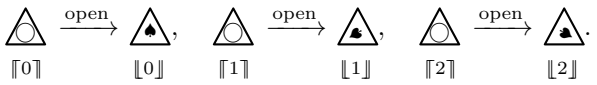


For a value $a \in \mathbb{F}_3$, we call a card of a without seals an *opened card*, denoted by $\llbracket a \rrbracket$, and a card of a with seals a *closed card*, denoted by $\lceil a \rceil$. That is, there are six states for a triangle card, as follows:

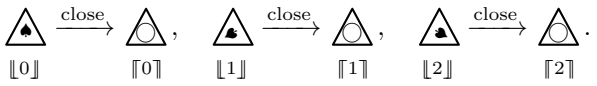


We define operations for a single triangle card as follows:

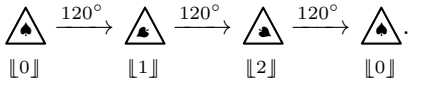
- Open/Close: An *open* operation transforms a closed card $\llbracket a \rrbracket$ into the opened card $\lfloor\!\lfloor a \rfloor\!\rfloor$, as follows:

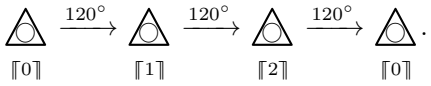


  Conversely, a *close* operation transforms an opened card $\lfloor\!\lfloor a \rfloor\!\rfloor$ into the closed card $\llbracket a \rrbracket$, as follows:



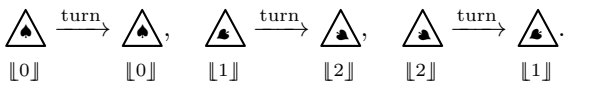
- Rotate: A *rotation* by 120° transforms a card of value a into a card of value $a+1$, as follows:

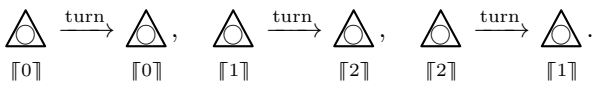




  Consequently, a rotation by $(n \times 120)°$ transforms a card of value a into a card of value $a+n$. We call this operation a *rotation n-times*.

- Turn: A *turn* operation turns over a card. That is, it transforms a card of value a into a card of value $-a$, as follows:



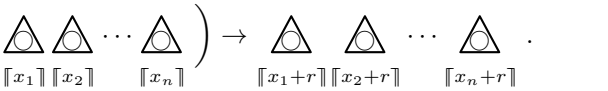


2.2 Shuffles for triangle cards

A *shuffle* is a probabilistic operation on a sequence of cards. In this paper, we use three types of shuffle: rotation shuffle, turning shuffle, and flower shuffle. We present example implementations of these shuffles in the appendix.

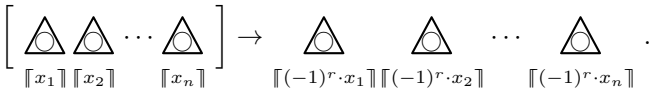
2.2.1 Rotation shuffle

This takes a sequence of n closed cards $(\llbracket x_1 \rrbracket, \llbracket x_2 \rrbracket, \cdots, \llbracket x_n \rrbracket)$ and outputs a sequence of n closed cards $(\llbracket x_1 + r \rrbracket, \llbracket x_2 + r \rrbracket, \cdots, \llbracket x_n + r \rrbracket)$, where r is a uniformly random number over $\mathbb{F}_3$ that is independent from the inputs and other randomness, and information-theoretically hidden from all parties. (See Figure 2 in the appendix.) We denote this shuffle as follows:



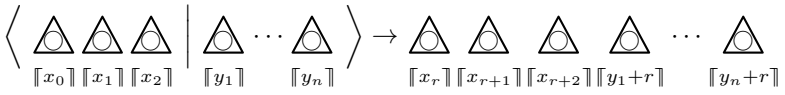
2.2.2 Turning shuffle

This takes a sequence of n closed cards $(\llbracket x_1 \rrbracket, \llbracket x_2 \rrbracket, \cdots, \llbracket x_n \rrbracket)$ and outputs a sequence of n closed cards $(\llbracket (-1)^r \cdot x_1 \rrbracket, \llbracket (-1)^r \cdot x_2 \rrbracket, \cdots, \llbracket (-1)^r \cdot x_n \rrbracket)$, where r is a uniformly random number over $\{0, 1\}$ that is independent from the inputs and other randomness, and information-theoretically hidden from all parties. (See Figure 3 in the appendix.) We denote this shuffle as follows:



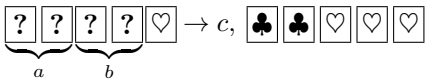
2.2.3 Flower shuffle

This takes a sequence of $3 + n$ closed cards $(\llbracket x_0 \rrbracket, \llbracket x_1 \rrbracket, \llbracket x_2 \rrbracket, \llbracket y_1 \rrbracket, \cdots, \llbracket y_n \rrbracket)$ and outputs a sequence of $3 + n$ closed cards $(\llbracket x_r \rrbracket, \llbracket x_{r+1} \rrbracket, \llbracket x_{r+2} \rrbracket, \llbracket y_1 + r \rrbracket, \cdots, \llbracket y_n + r \rrbracket)$, where r is a uniformly random number over $\mathbb{F}_3$ that is independent of the inputs and other randomness, and information-theoretically hidden from all parties. (See Figure 4 in the appendix.) We denote this shuffle as follows:



3 A solution based on existing methods

In this section, we briefly describe a straightforward solution for the secure $(3, 3)$-matching problem using a deck of cards employed in previous studies, where their front sides show $\boxed{\clubsuit}$, $\boxed{\heartsuit}$ and their back sides show the same $\boxed{?}$. A pair of face-down cards is called a *commitment* to 0 (resp. 1) if its face-up symbols are $(\clubsuit, \heartsuit)$ (resp. $(\heartsuit, \clubsuit)$). The main tool used is the *five-card trick*, proposed by den Boer [1], which takes two commitments to $a, b \in \{0, 1\}$ and a single additional card, and outputs the value $c = a \wedge b$ with five free cards as follows:

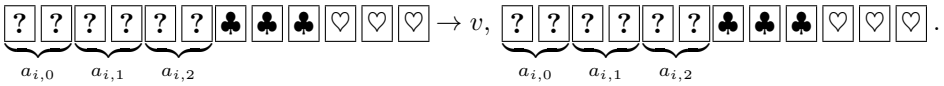


$\rightarrow c,$ $\boxed{\clubsuit}\boxed{\clubsuit}\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\heartsuit}$.

This requires a single shuffle called a *random cut*. Let Alice (resp. Bob) input 1 if she likes Bob (resp. Alice), and 0 otherwise. This solves a secure matching problem between two parties, where either both parties like each other or one party does not like the other. A straightforward solution for the secure $(3, 3)$-matching problem is to apply the five-card trick for every boy-girl pair. That is, each girl (indexed by i) submits a tuple of three commitments to $(a_{i,0}, a_{i,1}, a_{i,2}) \in \{0, 1\}^3$, where $a_{i,j} = 1$ if and only if she likes the j-th boy. Similarly, each boy submits a tuple of three commitments in the same manner. Then, for every $i, j \in \mathbb{F}_3$ we apply the five-card trick to the commitments to $a_{i,j}$ and $b_{j,i}$. We set $c_{i,j} = 1$ if the output is 1, and 0 otherwise. (Recall that $c_{i,j} = 1$ means that the i-th girl and j-th boy have a mutual affection.) This idea solves the secure $(3, 3)$-matching problem using $36 + 1$ cards and 9 shuffles.

One weakness of the above idea is that each party may deviate from the input rule, i.e., each party may submit two or more commitments to 1. In order to prevent a deviation from the input rule, we should check the input format in a zero-knowledge manner. Based on the idea of Shinagawa et al. [5] (Section 5, a voting protocol), we can check the validity

$v \in \{\mathsf{true}, \mathsf{false}\}$ of three commitments using additional six cards, without destroying the input, as follows:



This also requires a single shuffle. Thus, the number of cards required is 42 $(= 36 + 6)$, and the number of shuffles required is 15 $(= 9 + 6)$.

4 Our main solution

In this section, we solve the $(3, 3)$-matching problem using six triangle cards. Recall that the $(3, 3)$-matching function $f : (\mathbb{F}_3)^6 \to \{0, 1\}^9$ is defined follows:

$$f(a_0, a_1, a_2, b_0, b_1, b_2) = (c_{0,0}, c_{0,1}, c_{0,2}, c_{1,0}, c_{1,1}, c_{1,2}, c_{2,0}, c_{2,1}, c_{2,2}),$$

where $c_{i,j} = 1$ if $(a_i = j) \wedge (b_j = i)$ and $c_{i,j} = 0$ otherwise.

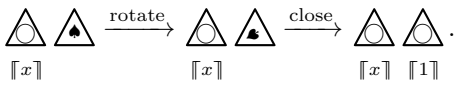
We first design a protocol for checking $x \stackrel{?}{=} 0$ in Section 4.1. Then, we construct a $(3, 3)$-matching protocol in Section 4.2.

4.1 IsZero protocol

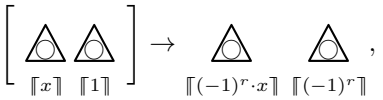
The IsZero protocol takes two cards $(\llbracket x \rrbracket, \llbracket 0 \rrbracket)$, and outputs a predicate $c \in \{0, 1\}$ for $x \stackrel{?}{=} 0$ without changing the sequence or revealing information of x beyond the predicate c.

The protocol proceeds as follows.

1. Rotate and close the right card as follows:

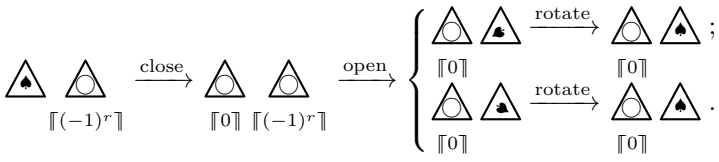


2. Apply a turning shuffle to these.

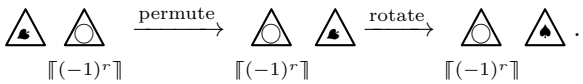


   where r is a random value generated in the shuffle.

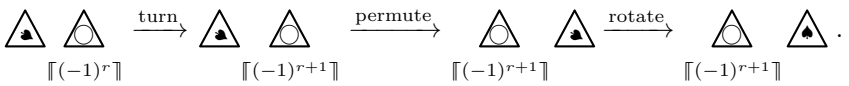
3. Open the left card. Let $v = (-1)^r \cdot x$ be the opened value.
   - **a.** The case $v = 0$: Close the left card, open the right card, and rotate the right card 2 (resp. 1) times if the opened value is 1 (resp. 2), as follows:



   - **b.** Case $v = 1$: Exchange the two cards, and rotate the right card 2-times, as follows:



c. Case $v = 2$: Turn the right card, exchange the two cards, and rotate the right card 1-time, as follows:



4. Output $c = 1$ if $v = 0$ and $c = 0$ otherwise.

4.1.1 Correctness

First, let us check that the resulting sequence is equal to the original sequence $(\llbracket x \rrbracket, \llbracket 0 \rrbracket)$. When $v = 0$, it holds that $x = 0$. Thus, the resulting sequence $(\llbracket 0 \rrbracket, \llbracket 0 \rrbracket)$ is equal to the original sequence. When $v = 1$, it is either the case that $(x, r) = (1, 0)$ or $(x, r) = (2, 1)$. Thus, the resulting sequence $(\llbracket (-1)^r \rrbracket, \llbracket 0 \rrbracket)$ is equal to the original sequence. When $v = 2$, either $(x, r) = (2, 0)$ or $(x, r) = (1, 1)$. Thus, the resulting sequence $(\llbracket (-1)^{r+1} \rrbracket, \llbracket 0 \rrbracket)$ is equal to the original sequence. Moreover, we can observe that c is equal to the predicate for $x \stackrel{?}{=} 0$, because $v = 0$ if and only if $x = 0$.

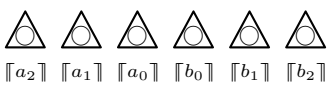
4.1.2 Security

When $c = 1$, i.e., $x = 0$, the opened value v is always 0. When $c = 0$, the opened value $v = 1$ (or 2) with a probability of exactly $1/2$. Thus, the distribution of v is statistically independent from x given c.

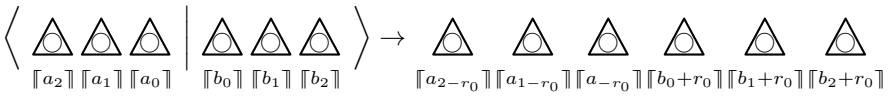
4.2 Secure $(3, 3)$-matching protocol

The protocol proceeds as follows.

1. Place the six cards as follows:

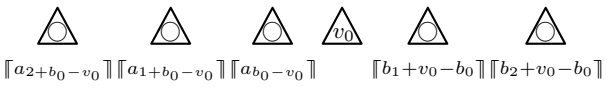


2. Apply a flower shuffle to the sequence:

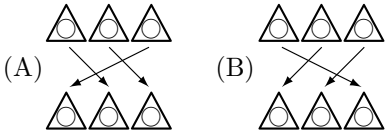


where r_0 is a random value generated in the shuffle. Note that the first three cards of the resulting sequence are $(a_{2-r_0}, a_{1-r_0}, a_{-r_0})$, because those of the original sequence are arranged in reverse order (a_2, a_1, a_0).

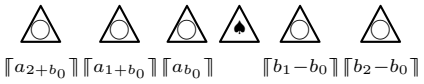
3. Open the fourth card. Let $v_0 = b_0 + r_0$ be the opened value. Now, the current sequence can be expressed as follows:



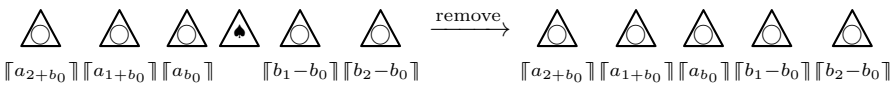
4. Permute the first three cards according to (A) if $v_0 = 1$ and (B) if $v_0 = 2$:



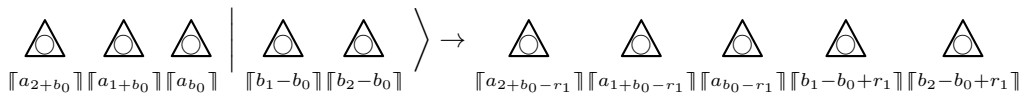
Rotate the fourth/fifth/sixth cards $(3-v_0)$-times. Now, the current sequence is as follows:



5. Apply the IsZero protocol to the third/fourth cards, and let c_0 be the output of this.
6. Remove the fourth card.



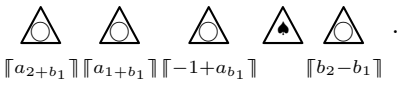
7. Apply a flower shuffle to the sequence:



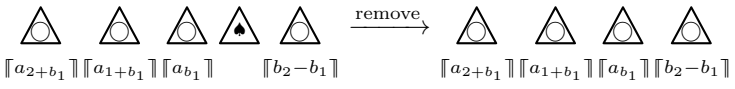
where r_1 is a random value generated in the shuffle.

8. Open the fourth card. Let $v_1 = b_1 - b_0 + r_1$ be the opened value. Now, the current sequence can be expressed as follows:

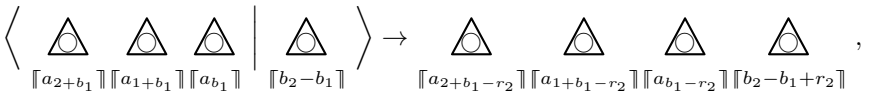
9. Permute the first three cards according to (A) in Step 4 if $v_1 = 1$ and (B) in Step 4 if $v_1 = 2$. Rotate the fourth/fifth cards $(3 - v_1)$-times, and rotate the third card 2-times. Now, the current sequence is as follows:



10. Apply the IsZero protocol to the third/fourth cards, and let c_1 be the output of this.
11. Remove the fourth card.

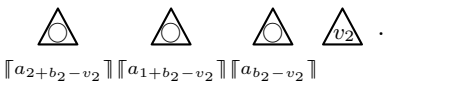


12. Apply a flower shuffle to the sequence:

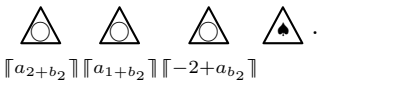


where r_2 is a random value generated in the shuffle.

13. Open the fourth card, and let $v_2 = b_2 - b_1 + r_2$ be the opened value. Now, the current sequence can be expressed as follows:



14. Permute the first three cards according to (A) in Step 4 if $v_2 = 1$ and (B) in Step 4 if $v_2 = 2$. Rotate the fourth/fifth cards $(3 - v_2)$-times, and rotate the third card 1-time. Now, the current sequence is as follows:



15. Apply the IsZero protocol to the third/fourth cards, and let c_2 be the output of this.
16. For $j = 0, 1, 2$, do the following: If $c_j = 1$, the $(j+3)$-th input b_j is publicly announced[1] by the $(j+3)$-th party, and we set $c_{j,b_j} = 1$ and $c_{i,j} = 0$ for $i \neq b_j$. Otherwise, set $(c_{0,j}, c_{1,j}, c_{2,j}) = (0, 0, 0)$.
17. Output $(c_{0,0}, c_{0,1}, c_{0,2}, c_{1,0}, c_{1,1}, c_{1,2}, c_{2,0}, c_{2,1}, c_{2,2})$.

4.2.1 Correctness

Let $j \in \{0, 1, 2\}$ be any index. From the correctness of the IsZero protocol, the value $c_{i,j}$ is 1 if it holds that both $i = b_j$ and $-j + a_{b_j} = 0$, and 0 otherwise. That is, $c_{i,j} = 1$ if $(a_i = j) \wedge (b_j = i)$, and $c_{i,j} = 0$ otherwise. Therefore, the above protocol correctly computes the $(3, 3)$-matching function.

4.2.2 Security

From the security of the IsZero protocol, the opened values of the IsZero protocol are statistically independent from the inputs $a_{b_0}, a_{b_1}, a_{b_2}$ given the outputs c_0, c_1, c_2. The opened values $v_0 = b_0 + r_0, v_1 = b_1 + r_1, v_2 = b_2 + r_2$ in Steps 3, 8, and 13, respectively, are masked by a uniformly random number $r_0, r_1, r_2 \in \mathbb{F}_3$. Thus, the distribution of the opened values and the distribution of the inputs are statistically independent given an output value.

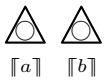
5 Secure computation for any function

In this section, we construct addition and multiplication protocols. Given two closed cards $\llbracket a \rrbracket, \llbracket b \rrbracket$ as inputs, the former protocol outputs $\llbracket a + b \rrbracket$, and the latter outputs $\llbracket ab \rrbracket$. Because any function over $\mathbb{F}_3$ can be expressed using additions and multiplications, we can securely compute any function over $\mathbb{F}_3$ by combining these protocols.

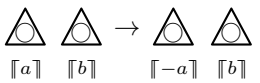
5.1 Addition protocol

The protocol proceeds as follows:

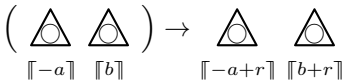
1. Place the two cards as follows:



2. Apply the turning operation to the left card:



3. Apply a rotation shuffle to the cards:

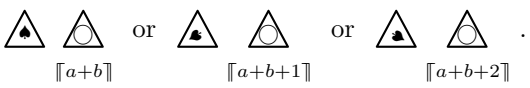


[1] We note that the j-th boy such that $c_j = 1$ can maliciously announce the i-th girl such that $i \neq b_j$, and this is not noticed when the i-th girl is attracted to the j-th boy. Thus, the levels of trust between boys and girls are different. To avoid this asymmetry of boys and girls, we can use one additional card. That is, by using an additional card, we can copy the input of each boy and verify the girl they have in mind.

Table 2 All possibilities of the final sequence in our addition protocol.

(a,b)	$a+b$	s_0	s_1	s_2
$(0,0)$	0	$(0,0)$	$(1,1)$	$(2,2)$
$(0,1)$	1	$(0,1)$	$(1,2)$	$(2,0)$
$(0,2)$	2	$(0,2)$	$(1,0)$	$(2,1)$
$(1,0)$	1	$(2,0)$	$(0,1)$	$(1,2)$
$(1,1)$	2	$(2,1)$	$(0,2)$	$(1,0)$
$(1,2)$	0	$(2,2)$	$(0,0)$	$(1,1)$
$(2,0)$	2	$(1,0)$	$(2,1)$	$(0,2)$
$(2,1)$	0	$(1,1)$	$(2,2)$	$(0,0)$
$(2,2)$	1	$(1,2)$	$(2,0)$	$(0,1)$

4. Open the first card. Then, a commitment to $a+b$ is obtained as follows:



5.1.1 Correctness

After applying a rotation shuffle to the sequence $(\llbracket -a \rrbracket, \llbracket b \rrbracket)$ in Step 3, the resulting sequence is one of s_0, s_1, s_2 as follows:

$$s_0 = (\llbracket -a \rrbracket, \llbracket b \rrbracket),$$
$$s_1 = (\llbracket -a+1 \rrbracket, \llbracket b+1 \rrbracket),$$
$$s_2 = (\llbracket -a+2 \rrbracket, \llbracket b+2 \rrbracket).$$

Table 2 shows all possibilities of the final sequence in our addition protocol. We can observe that in each sequence, the right value is $a+b+\ell$, where ℓ is the left value. Therefore, our addition protocol is correct.

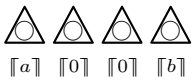
5.1.2 Security

Let v be the opened value in Step 4. Owing to the rotation shuffle in Step 3, v is equal to $-a+r$, where $r \in \mathbb{F}_3$ is a uniformly random value that is hidden from all parties and independent from the inputs (a,b). Thus, the distribution of v and the distribution of the inputs are statistically independent.

5.2 Multiplication protocol

The protocol proceeds as follows:

1. Place the four cards $(\llbracket a \rrbracket, \llbracket 0 \rrbracket, \llbracket 0 \rrbracket, \llbracket b \rrbracket)$ as follows:



2. Apply our addition protocol to the leftmost three cards $(\llbracket a \rrbracket, \llbracket 0 \rrbracket, \llbracket 0 \rrbracket)$:

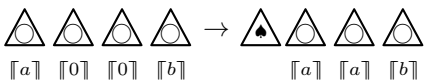
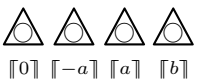


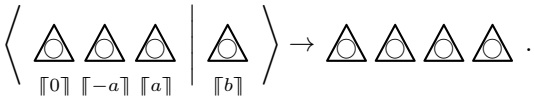
Table 3 All possibilities of the final sequence in our multiplication protocol. (An underlined value corresponds to the output.)

(a,b)	ab	s_0	s_1	s_2
$(0,0)$	0	$(\underline{0},0,0,0)$	$(0,0,\underline{0},1)$	$(0,\underline{0},0,2)$
$(0,1)$	0	$(0,0,\underline{0},1)$	$(0,\underline{0},0,2)$	$(\underline{0},0,0,0)$
$(0,2)$	0	$(0,\underline{0},0,2)$	$(\underline{0},0,0,0)$	$(0,0,\underline{0},1)$
$(1,0)$	0	$(\underline{0},2,1,0)$	$(1,2,\underline{0},1)$	$(2,\underline{0},1,2)$
$(1,1)$	1	$(0,2,\underline{1},1)$	$(2,\underline{1},0,2)$	$(\underline{1},0,2,0)$
$(1,2)$	2	$(0,\underline{2},1,2)$	$(\underline{2},1,0,0)$	$(1,0,\underline{2},1)$
$(2,0)$	0	$(\underline{0},1,2,0)$	$(1,2,\underline{0},1)$	$(2,\underline{0},1,2)$
$(2,1)$	2	$(0,1,\underline{2},1)$	$(1,\underline{2},0,2)$	$(\underline{2},0,1,0)$
$(2,2)$	1	$(0,\underline{1},2,2)$	$(\underline{1},2,0,0)$	$(2,0,\underline{1},1)$

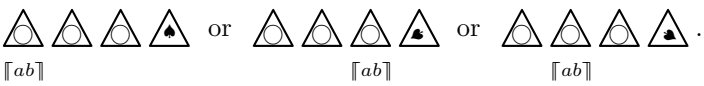
3. Close the first card, and apply a turning operation to the second card:



4. Apply a flower shuffle to the sequence:



5. Open the fourth card. Then, the closed card $\llbracket ab \rrbracket$ is obtained as follows:



5.2.1 Correctness

After applying a flower shuffle to the sequence $(\llbracket 0 \rrbracket, \llbracket -a \rrbracket, \llbracket a \rrbracket, \llbracket b \rrbracket)$ in Step 4, the resulting sequence is one of s_0, s_1, s_2 as follows:

$$s_0 = (\llbracket 0 \rrbracket, \llbracket -a \rrbracket, \llbracket a \rrbracket, \llbracket b \rrbracket),$$
$$s_1 = (\llbracket -a \rrbracket, \llbracket a \rrbracket, \llbracket 0 \rrbracket, \llbracket b+1 \rrbracket),$$
$$s_2 = (\llbracket a \rrbracket, \llbracket 0 \rrbracket, \llbracket -a \rrbracket, \llbracket b+2 \rrbracket).$$

Table 3 shows all the possibilities of the final sequence in our multiplication protocol. We can observe that in each sequence, the underlying value is equal to ab, and the first/second/third value is underlined if the fourth value is 0/2/1, respectively. Therefore, our multiplication protocol is correct.

5.2.2 Security

Let v_1, v_2 be the opened values in Steps 2 and 5, respectively. From Section 5.1.2, v_1 is statistically independent from the inputs. Owing to the flower shuffle in Step 4, v_2 is equal to $b + r$, where $r \in \mathbb{F}_3$ is a uniformly random value that is hidden from all parties and independent from v_1 and the inputs (a, b). Thus, the distribution of the opened values and the distribution of the inputs are statistically independent.

6 Conclusion

In this paper, we proposed novel cards called triangle cards, and solved the secure $(3,3)$-matching problem using six triangle cards. We also designed a protocol for any function over $\mathbb{F}_3$. One could ask if our technique applies to $\mathbb{F}_n$ for any n. A straightforward generalization to this case does not work. This is because for a regular n-sided polygon with $n > 3$, a rotation over $n-1$ points with a single fixed point cannot be implemented by a physical operation. In contrast, in the triangle case a turning operation corresponds to an operation of this type: a rotation between 1 and 2 with a fixed point 0. Our protocols exploits this property. This is why we concentrate on a triangle rather than a general polygon. An interesting open question is that of finding new physical objects that enable the construction of an efficient protocol for the secure (n,m)-matching problem for any integers n, m.

References

1 Bert den Boer. More efficient match-making and satisfiability: *The Five Card Trick*. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings*, volume 434 of *Lecture Notes in Computer Science*, pages 208–217. Springer, 1989. doi:10.1007/3-540-46885-4_23.

2 Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 218–229. ACM, 1987. doi:10.1145/28395.28420.

3 Takaaki Mizuki and Hiroki Shizuya. Practical card-based cryptography. In Alfredo Ferro, Fabrizio Luccio, and Peter Widmayer, editors, *Fun with Algorithms - 7th International Conference, FUN 2014, Lipari Island, Sicily, Italy, July 1-3, 2014. Proceedings*, volume 8496 of *Lecture Notes in Computer Science*, pages 313–324. Springer, 2014. doi:10.1007/978-3-319-07890-8_27.

4 Takaaki Mizuki and Hideaki Sone. Six-card secure AND and four-card secure XOR. In Xiaotie Deng, John E. Hopcroft, and Jinyun Xue, editors, *Frontiers in Algorithmics, Third International Workshop, FAW 2009, Hefei, China, June 20-23, 2009. Proceedings*, volume 5598 of *Lecture Notes in Computer Science*, pages 358–369. Springer, 2009. doi:10.1007/978-3-642-02270-8_36.

5 Kazumasa Shinagawa, Takaaki Mizuki, Jacob C. N. Schuldt, Koji Nuida, Naoki Kanayama, Takashi Nishide, Goichiro Hanaoka, and Eiji Okamoto. Multi-party computation with small shuffle complexity using regular polygon cards. In Man Ho Au and Atsuko Miyaji, editors, *Provable Security - 9th International Conference, ProvSec 2015, Kanazawa, Japan, November 24-26, 2015, Proceedings*, volume 9451 of *Lecture Notes in Computer Science*, pages 127–146. Springer, 2015. doi:10.1007/978-3-319-26059-4_7.

6 Kazumasa Shinagawa, Takaaki Mizuki, Jacob C. N. Schuldt, Koji Nuida, Naoki Kanayama, Takashi Nishide, Goichiro Hanaoka, and Eiji Okamoto. Secure multi-party computation using polarizing cards. In Keisuke Tanaka and Yuji Suga, editors, *Advances in Information and Computer Security - 10th International Workshop on Security, IWSEC 2015, Nara, Japan, August 26-28, 2015, Proceedings*, volume 9241 of *Lecture Notes in Computer Science*, pages 281–297. Springer, 2015. doi:10.1007/978-3-319-22425-1_17.

7 Kazumasa Shinagawa, Koji Nuida, Takashi Nishide, Goichiro Hanaoka, and Eiji Okamoto. Committed AND protocol using three cards with more handy shuffle. In *2016 International Symposium on Information Theory and Its Applications, ISITA 2016, Monterey, CA, USA, October 30 - November 2, 2016*, pages 700–702. IEEE, 2016. URL: http://ieeexplore.ieee.org/document/7840515/.

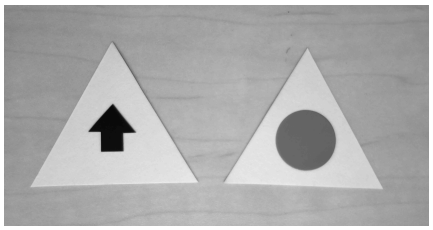


Figure 1 Triangle cards: an opened card (left) and a closed card (right).

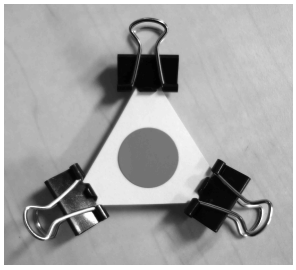


Figure 2 A rotation shuffle.

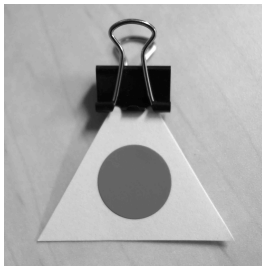


Figure 3 A turning shuffle.

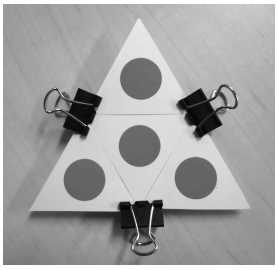


Figure 4 A flower shuffle.

8 Itaru Ueda, Akihiro Nishimura, Yu-ichi Hayashi, Takaaki Mizuki, and Hideaki Sone. How to implement a random bisection cut. In Carlos Martín-Vide, Takaaki Mizuki, and Miguel A. Vega-Rodríguez, editors, *Theory and Practice of Natural Computing - 5th International Conference, TPNC 2016, Sendai, Japan, December 12-13, 2016, Proceedings*, volume 10071 of *Lecture Notes in Computer Science*, pages 58–69, 2016. doi:10.1007/978-3-319-49001-4_5.

A Implementation of cards and shuffles

Figure 1 shows an example implementation of a triangle card. The left card is an opened card, and the right card is a closed card. A rotation shuffle is implemented by using three clips: all cards are stacked using three clips as in Figure 2, and then the stack is spun like a "roulette wheel." A turning shuffle is implemented by using a single clip: all cards are stacked using a clip as in Figure 3, and then the stack is thrown in a spinning manner, like a coin toss (this technique is called a *spinning throw* [8]). A flower shuffle for $(\llbracket x_0 \rrbracket, \llbracket x_1 \rrbracket, \llbracket x_2 \rrbracket, \llbracket y_1 \rrbracket, \cdots, \llbracket y_n \rrbracket)$ is implemented by using three clips: As in Figure 4, the last n cards $\llbracket y_1 \rrbracket, \cdots, \llbracket y_n \rrbracket$ are stacked and placed on the center of the flower, $\llbracket x_0 \rrbracket$ is placed on the top petal, $\llbracket x_1 \rrbracket$ is placed on the right petal, and $\llbracket x_2 \rrbracket$ is placed on the left petal. Then, the flower is spun like a "roulette wheel."