Efficient Card-Based Cryptographic Protocols for Millionaires' Problem Utilizing Private Permutations

Takeshi Nakai[1](✉), Yuuki Tokushige[1], Yuto Misawa[2], Mitsugu Iwamoto[1], and Kazuo Ohta[1]

[1] Department of Informatics, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan
{t-nakai,yuuki.tokushige,mitsugu,kazuo.ohta}@uec.ac.jp

[2] Smart Card Systems Department, Toshiba Corporation, 1, Komukai, Toshiba-cho, Saiwai-ku, Kawasaki 212-8583, Japan
yuto1.misawa@toshiba.co.jp

Abstract. We propose several efficient card-based cryptographic protocols for the millionaires' problem by introducing a new operation called *Private Permutation* (PP) instead of the *shuffle* used in existing card-based cryptographic protocols. Shuffles are useful randomization techniques for designing card-based cryptographic protocols for logical gates, and this approach seems to be almost optimal. This fact, however, implies that there is room for improvements if we do not use logical gates as building blocks for secure computing, and we show that such an improvement is actually possible for the millionaires' problem. Our key technique, PP, is a natural randomization operation for permuting a set of cards behind the player's back, and hence, a shuffle can be decomposed into two PPs with one communication between them. Thus PP not only allows us to transform Yao's seminal protocol into a card-based cryptographic protocol, but also enables us to propose entirely novel and efficient protocols by securely updating bitwise comparisons between two numbers. Furthermore, it is interesting to remark that one of the proposed protocols has a remarkably deep connection to the well-known logical puzzle known as "*The fork in the road*".

1 Introduction

Background. Multiparty computation (MPC) can be realized by using several cards, and such a special implementation of MPC is known as a *card-based cryptographic protocol* [2,5]. Much of the research related to card-based cryptographic protocols has been devoted to secure computation of logical gates such as AND and XOR[1], since any computation can be implemented by their combinations.

Y. Misawa—This work was carried out when he was affiliated to the University of Electro-Communications.

[1] In card-based cryptographic protocols, NOT is easy to implement, and an OR operation is easily derived from an AND operation.

S. Foresti and G. Persiano (Eds.): CANS 2016, LNCS 10052, pp. 500–517, 2016.
DOI: 10.1007/978-3-319-48965-0_30

The central issue when designing efficient card-based cryptographic protocols for logical gates is to minimize the number of cards required in the protocol. For instance, Mizuki–Sone [9] realized AND and XOR operations on two binary inputs with six and four cards, respectively, and recently, Koch–Walzer–Härtel [7] reduced the number of cards to five in AND[2]. On the other hand, randomization is also important in order to realize secure card-based cryptographic protocols. In this regard, an operation known as *shuffle* is considered to be useful for implementing logical gates securely with a smaller number of cards, and the usage of this operation has been extensively studied thus far.

We note that shuffles in card-based cryptographic protocols are different from those in ordinary card games in terms of two points: The first difference is that a shuffle in a card-based cryptographic protocol specifies a certain permutation, whereas a shuffle in ordinary card games permutes the set of cards in a completely random manner. The second difference is that the result of a permutation *must not* be known to *any* players (including the player performing the shuffle). For instance, a *random bisection cut* [9] is a useful type of shuffles in the following manner: an even number of cards are divided into two sets consisting of the same number of cards, and these two sets are permuted (in this case, exchanged) many times until *none* of the players can recognize how many times the two sets of cards are permuted.

Motivation and Our Idea. We observe here that the following two problems exist in card-based cryptographic protocols based on logical gates and shuffles:

(1) Constructing a protocol by using logical gates is a general technique, but it can be less efficient than protocols specially developed to perform a certain function.
(2) From the viewpoint of MPC, a shuffle is not a single operation since it requires at least two players to communicate with each other[3], and hence, a card-based cryptographic protocol is not efficient if it uses a shuffle as a building block.

We discuss (1) and (2) in detail before we propose our idea:

(1) In the secure computing of logical gates by card-based cryptographic protocols, it is known that one bisection cut is necessary and sufficient in state-of-the-art card-based cryptographic protocols [9]. This fact implies that, when we compute a certain function, random bisection cuts are necessary at least with the number of logical gates so as to represent the function. For instance, consider the case of the *millionaires' problem* initiated by Yao's seminal work [1], which is a secure two-party computation involving a comparison of two numbers without making each millionaire's wealth public. Comparing two numbers less than $m \in$

[2] We assume in this paper that the results are correctly computed with a probability 1. If a computation error is allowed with small probability, it is shown in [7] that four cards are sufficient.

[3] Although this fact is mentioned in [7], the efficiency of the protocol based on this fact is not discussed by these authors, and hence, they use shuffles as building blocks.

$\mathbb{N}$ by logical gates can be realized as in Fig. 1, in which logical AND and OR operations are necessary $2\lceil \log m \rceil - 1$ and $2\lceil \log m \rceil - 2$ times, respectively[4]. When executing these logical operations, the COPY operation [9] is also necessary for $\neg a_i$ and b_i in each comparison of bits, and hence, $6\lceil \log m \rceil - 5$ random bisection cuts are necessary in total in order to implement the millionaires' problem as shown in Fig. 1. Noticing that a random bisection cut is necessary for randomization, it seems difficult to reduce this number as long as we implement the millionaires' problem based on logical operations as shown in Fig. 1.

We can expect this inefficiency to be resolved if we design a card-based cryptographic protocol specialized for the function computed in the protocol, although an improvement such as this has not been studied intensively to date. Proceeding with this idea, it is natural to recall Yao's solution to the *millionaires' problem* ([1], see Sect. 3.1) since it does not depend on logical gates but specializes in comparing two numbers privately. As we will see in Sect. 3.1, for instance, Yao's protocol involves public key encryption, which is difficult to implement by logical gates, but is easy to realize by using face-down cards *without* public/private keys! As a result, it is easy to implement Yao's protocol by cards if we do *not* restrict ourselves to using card-based cryptographic protocols for *logical gates*.

When implementing Yao's protocol by using cards that do not depend on logical gates, it should become clear that his protocol uses *private computation* since it is an MPC protocol. On the other hand, every operation is assumed to be *public* in existing card-based cryptographic protocols. Hence, in this paper, we explicitly allow such a private operation if it is possible to implement by cards.

```
input: $a = (a_n ... a_2\, a_1)_2,\ b = (b_n ... b_2\, b_1)_2$ ;
$f_1 = \bar{a}_1 \wedge b_1$ ;
for( $i$ : 2 to $n$ ) {
    $f_i = \bar{a}_i \wedge b_i \vee ( \bar{a}_i \vee b_i ) \wedge f_{i-1}$ ;
}
output: $f_n$ ;
  if $f_n = 0$ then $a \geq b$
  if $f_n = 1$ then $a < b$
```

Fig. 1. Comparing protocol constructed by logical gates

(2) In previous work, a shuffle is considered as a building block for randomization, but actually, it is not a single operation from the viewpoint of MPC. For instance, a random bisection cut by Alice can be realized as follows: Alice first generates a random number r_A and permutes bisected cards r_A times behind her back, and sends the permuted cards to the other player, say Bob. Bob privately generates a random number r_B and permutes bisected cards r_B times behind

[4] Throughout this paper, logarithmic base is 2.

his back. If r_A and r_B are kept private by Alice and Bob, respectively, this protocol shuffles bisected cards $r_A + r_B$ times, and no one can know the number of permutations.

Note that such private randomness and private operations ((r_A, r_B) and permutation in this example, respectively) are often used in MPC. Hence, we call such a permutation behind someone's back *Private Permutation* (PP). The introduction of PPs makes it easy to see that shuffles, including a random bisection cut, generally consist of at least two PPs and one communication among these PPs. Note that the number of PPs and communications is considered as computational cost, and the number of cards is considered as memory cost.

Table 1. Comparison of Proposed Card-based Cryptographic Protocols

Protocols	# of Comm.	# of PP	# of cards
Logical gates (Fig. 1)	$6\lceil \log m \rceil - 5$	$12\lceil \log m \rceil - 10$	$4\lceil \log m \rceil + 2$
Proposed protocol I (Yao)	1	2	$2m$
Proposed protocol II (storage)	$2\lceil \log m \rceil$	$2\lceil \log m \rceil$	$4\lceil \log m \rceil + 2$

Our Contributions. As shown above, the concept of a PP is motivated by (1) and (2). We propose two protocols corresponding to (1) and (2), denoted as proposed protocols I and II, respectively. The evaluations of our results presented in this paper are summarized in Table 1, where we use the number of communications, PPs, and cards as efficiency measures.

We resolve problem (1) by constructing a card-based cryptographic protocol for the millionaires' problem based on Yao's protocol for two numbers less than or equal to m (proposed protocol I). Even though this protocol is naïve, *only one communication and two PPs* are sufficient, which is a considerable improvement of card-based cryptographic protocols based on logical gates ($6\lceil \log m \rceil - 5$ and $12\lceil \log m \rceil - 10$, respectively). On the other hand, the number of cards required by the protocol is $2m$, which is much worse than the card-based millionaires' problem based on logical gates ($4\lceil \log m \rceil + 2$).

Regarding problem (2), we expect that a more efficient card-based protocol can be proposed in terms of the number of communications, PPs, and cards. Actually, we propose an entirely new and efficient card-based cryptographic protocol specially developed to solve the millionaires' problem. This protocol succeeds in reducing the number of communications and PPs to almost 1/3 and 1/6, respectively, compared to the protocol for logical gates, whereas the number of cards remains the same (see Proposed protocol II in Table 1). The new protocol compares two numbers bit by bit, starting from the less significant bit, and the compared results are recorded on cards, called *storage.* The results recorded in storage need to be kept secret from both Alice and Bob, to solve the millionaires' problem securely. Hence, we show how to manipulate the storage privately by using PPs. It is very interesting to note that the technique on which this manipulation is based proved to be the same as that of the well-known logical puzzle "*The Fork in the Road*[5]" [6, p. 25]. This observation will be introduced when explaining the idea of the proposed protocol II in Sect. 4.1.

Organization. The remaining part of this paper is organized as follows: We introduce several notations, basic operations of cards including PP, and the security notion for card-based cryptographic protocols in Sect. 2. In Sect. 3, the card-based cryptographic protocol for the millionaires' problem based on Yao's protocol is presented. Section 4 is devoted to the proposal of a new card-based cryptographic protocol *with storage*, which is efficient from the viewpoint of the number of communications and PPs. We summarize our results in Sect. 5 and discuss the improvements of the protocol proposed in Sect. 4.

2 Preliminaries

2.1 Notations and Basic Operations

In card-based cryptographic protocols, we normally use two types of cards such as $\boxed{\clubsuit}$ and $\boxed{\heartsuit}$, which are represented in the following sentences by $\clubsuit$ and $\heartsuit$, respectively. We assume that two cards with the same mark are indistinguishable. We also assume that all cards have the same design on their reverse sides, and that they are indistinguishable and represented as $\boxed{?}$. The Boolean values 0 and 1 are encoded as $\clubsuit\heartsuit$ and $\heartsuit\clubsuit$, respectively. Note that we regard the sequence of cards as a vector. In this paper, we use the following fundamental card operations [8]. Note that these operations are executed *publicly*.

- Face up: $\boxed{?} \mapsto \boxed{\clubsuit}$, $\boxed{?} \mapsto \boxed{\heartsuit}$
- Face down: $\boxed{\clubsuit} \mapsto \boxed{?}$, $\boxed{\heartsuit} \mapsto \boxed{?}$
- Swap: $\boxed{?}\boxed{?}$ (represents $x \in \{0,1\}$) $\mapsto$ $\boxed{?}\boxed{?}$ (represents $\neg x \in \{0,1\}$)

If a pair of face-down cards for the Boolean value $x \in \{0,1\}$, it is called *commitment*. The term Swap indicates reversal of the left and the right of the commitment.

[5] This problem is summarized as follows: An logician finds himself on an island inhabited by two tribes: liars and truth-tellers. Members of the one tribe always tell the truth, whereas members of the other tribe always tell lies. The logician reaches a fork in a road and has to ask a native bystander which branch he should take to reach the village. He has no way of telling whether the native is a truth-teller or a liar. The logician only asks *one* question. From the reply he knows which road to take. What question does he ask?.

2.2 Random Bisection Cut and Private Permutation

Random Bisection Cut. This is a key technique to realize efficient card-based cryptographic protocols for logical gates, e.g., 6-card AND protocol [9], which is described as follows:

For a positive integer v, suppose that there is a sequence of $2v$ face-down cards. Denote the left and right halves by $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$, respectively.

$$\overbrace{\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{=:\boldsymbol{u}_1}}^{v\text{ cards}}\overbrace{\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{=:\boldsymbol{u}_2}}^{v\text{ cards}} \tag{1}$$

Then, $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are interchanged or left unchanged with probability 1/2. Depicting this by using figures, one of either

$$\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\boldsymbol{u}_1}\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\boldsymbol{u}_2} \quad \text{or} \quad \underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\boldsymbol{u}_2}\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\boldsymbol{u}_1} \tag{2}$$

is selected with a probability 1/2. If *no player knows* whether one of the above is selected, such a shuffle is known as a *random bisection cut.*

A random bisection cut is known to be a convenient randomization technique for implementing card-based protocols for logical gates securely. However, it has two drawbacks.

The first drawback is that it is not known how to use this technique other than in the card-based cryptographic protocols for logical gates. In other words, this technique is not useful for implementing Yao's protocol, for instance, as a card-based cryptographic protocol because it does not use logical gates.

The second drawback is that it is not possible for one player to realize this technique. That is, a random bisection cut by Alice can be realized as follows: Alice first generates a random number r_A and permutes the bisected cards r_A times behind her back, and sends the permuted cards to the other player, say Bob. Bob privately generates a random number r_B and permutes the bisected cards r_B times behind his back. If r_A and r_B are kept private by Alice and Bob, respectively, this protocol permutes the bisected cards $r_A + r_B$ times, and no one can know the number of permutations. As long as we implement card-based cryptographic protocols based on logical gates, at least one shuffle such as a random bisection cut is necessary for *every* logical gate, which would have a highly adverse impact on the efficiency of the protocols.

Private Permutation. We resolve the above-mentioned drawbacks by decomposing the shuffle operation into the private permutations behind the player's back and the communication between them. Hence, we introduce a new randomization operation called *Private Permutation* (PP), which can be formalized as follows:

For a positive integer t, let $\boldsymbol{c} \in \{\clubsuit, \heartsuit\}^t$ be a vector consisting of t face-down cards. For a set $\mathcal{P}_t$ of all permutations over[6] $[t] := \{1, 2, \ldots, t\}$, let $\mathcal{R}_t \subset \mathcal{P}_t$

[6] In this paper, we define $[n] := \{1, 2, \ldots, n\}$ for an integer $n \in \mathbb{N}$.

be a set of possible permutations. We also define $\mathcal{R}_t = \{\pi_0, \pi_1, \ldots, \pi_{|\mathcal{R}_t|-1}\}$. Then, for a positive integer t and a set of possible permutations $\mathcal{R}_t$, the private permutation is formalized as follows:

$$\mathsf{PP}^{[t]}_{\mathcal{R}_t}(\boldsymbol{c}, s) := \pi_s(\boldsymbol{c}), \quad s = 0, 1, \ldots, |\mathcal{R}_t| - 1.$$

Note that the same function was introduced by others [7,8] although we impose an additional assumption on this function. Namely, we assume that *the player executing* $\mathsf{PP}^{[t]}_{\mathcal{R}_t}$ *keeps s secret, whereas he/she makes the other parameters public*, which is easy to realize by permuting the cards behind the player's back. This requirement is firstly introduced in this paper explicitly by considering that shuffle implicitly assumes the necessity of PPs. Note that, in the existing card-based cryptographic protocols, every operation other than shuffle is assumed to be executed in public. Note that, not only the random bisection cut, but also several different types of *shuffles*, e.g., [10] can be realized by PPs in a similar manner by specifying $\mathcal{R}_t$ appropriately.

For instance, consider the set of permutations capable of randomly *interchanging* the first and the latter halves of a vector as follows: For a positive integer v, $\mathcal{R}^{\mathsf{ic}}_{2v} := \{\pi_0, \pi_1\} \subset \mathcal{P}_{2v}$ where

$$\pi_0 := (1, \ldots, v, v+1, \ldots, 2v), \text{ and } \pi_1 := (v+1, \ldots, 2v, 1, \ldots, v), \tag{3}$$

which means that $\pi_0(\boldsymbol{c}) = (\boldsymbol{u}_1, \boldsymbol{u}_2)$ and $\pi_1(\boldsymbol{c}) = (\boldsymbol{u}_2, \boldsymbol{u}_1)$ for $\boldsymbol{c} := (\boldsymbol{u}_1, \boldsymbol{u}_2)$ given by (1). Then, the random bisection cut for $2v$ cards is represented as $\mathsf{PP}^{[2v]}_{\mathcal{R}^{\mathsf{ic}}_{2v}}(\boldsymbol{c}, s) = \pi_s(\boldsymbol{c})$ where s is chosen from $\{0, 1\}$ uniformly at random and it is known only by the player executing this operation. In executing the random bisection cut, for the sequence of cards $\boldsymbol{c}$, Alice executes $\mathsf{PP}^{[2v]}_{\mathcal{R}^{\mathsf{ic}}_{2v}}(\boldsymbol{c}, r_A) =: \boldsymbol{c}'$ by using her private randomness $r_A \in \{0, 1\}$, and $\boldsymbol{c}'$ is sent to Bob. Bob also executes $\mathsf{PP}^{[2v]}_{\mathcal{R}^{\mathsf{ic}}_{2v}}(\boldsymbol{c}', r_B)$ by using his private randomness $r_B \in \{0, 1\}$.

Efficiency Measures. Most of the previous work, e.g., [8,11], considers the number of shuffles as the *computational complexity* since shuffle is the most time-consuming operation. On the other hand, in this paper we consider that the computational complexity is evaluated by the number of PPs and communications since such measures are suitable for MPC. In this paper, successive PPs executed by one player without communication and/or face up is counted as *one* PP since the composition of permutations is also regarded as a permutation and the subsequent private permutation can be executed at once behind the player's back.

2.3 Security Notion

Throughout this paper, we assume that both Alice and Bob are semi-honest players. Following [4], we introduce the security notion (perfect secrecy) of card-based cryptographic protocols for the millionaires' problem.

In defining the security of card-based cryptographic protocols, *view* plays a key role. View is roughly defined to be *a vector of random variables*[7] *corresponding to the data that each player can obtain in the protocol.* More precisely, view is a vector which consists of random variables corresponding to the input of the player, the output of the protocol, public information all players can gain, and random values which are used when the player makes a random choice.

For a fixed integer $m \in \mathbb{N}$, let $a \in [m]$ and $b \in [m]$ be positive integers representing the wealth of Alice and Bob respectively. In this case, the inputs by Alice and Bob for the protocol are a and b, respectively. The common output of the millionaires' problem for Alice and Bob is represented as $\chi^{\mathsf{ge}}(a, b)$ where

$$\chi^{\mathsf{ge}}(u, v) := \begin{cases} 1 & \text{if } u \geq v \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

for positive integers $u, v \in [m]$.

The information obtained by Alice and Bob in the protocol can be classified into *private information* denoted by r_A and r_B, and *public information* denoted by λ. Hence, Alice's (resp. Bob's) view can be described as the sequence of random variables corresponding to her (resp. his) input a (resp. b), output of the protocol, private information r_A (resp., r_B) and public information λ. The private information r_A (resp., r_B) is the random number generated by Alice (resp., Bob) and used in PPs. The public information is the cards that Alice and Bob made public by turning them face up. Note that, in ordinary MPC, view includes information that each player receives via private channel, but in card-based cryptographic protocols, there is no private channel. Only face-up cards can reveal information, and hence, we can define the face-up cards are included in the view as public information. Let R_A, R_B, and Λ be random variables corresponding to the values r_A, r_B, and λ, respectively. Then, the views of Alice and Bob are represented as $(A, \chi^{\mathsf{ge}}(A, B), R_A, \Lambda)$ and $(B, \chi^{\mathsf{ge}}(A, B), R_B, \Lambda)$, respectively.

Intuitively, if all Alice's (resp., Bob's) private and public information can be simulated from Alice's (resp., Bob's) input and output, we can say that no information is contained in the private and public information other than Alice's (resp., Bob's) input and output. Hence, we can formulate perfect secrecy of card-based cryptographic protocols for the millionaires' problem as follows:

Definition 1 (Perfect secrecy). *Consider the millionaires' problem for Alice and Bob. We say that the card-based cryptographic protocol for the millionaires' problem is perfectly secure if there exist polynomial-time simulators* S_A *and* S_B *such that for all possible inputs* a *and* b*, it holds that*

$$\mathsf{S}_A(a, c_{a,b}) \overset{\text{perf}}{\equiv} (a, \chi^{\mathsf{ge}}(a, b), R_A, \Lambda) \quad \textit{and} \quad \mathsf{S}_B(b, c_{a,b}) \overset{\text{perf}}{\equiv} (b, \chi^{\mathsf{ge}}(a, b), R_B, \Lambda) \tag{5}$$

[7] Throughout the paper, random variables are represented by capital letters. The probability that a random variable X takes a value x is represented by $\mathsf{Pr}\{X = x\}$ which is also written as $P_X(x)$ for short. Mathematically, random variable is defined to be a map from probability space to the set of real numbers. However, for simplicity, we allow the cards ♣, ♡ to be treated as the values of random variables in each view.

where $U \overset{\text{perf}}{\equiv} V$ means that the (joint) probability distributions P_U and P_V corresponding to the random variables U and V, respectively, are perfectly the same.

3 Proposed Protocol I: Card-Based Cryptographic Protocol for Millionaires' Problem Based on Yao's Solution

3.1 Yao's Solution and Our Idea Behind the Proposed Protocol I

We propose a card-based cryptographic protocol that resolves the millionaires' problem by cards based on Yao's original solution. Before providing our protocol, we explain Yao's public key based solution [1] with the following:

Yao's Solution to the Millionaires' Problem. For a fixed integer $m \in \mathbb{N}$, assume that Alice and Bob have wealth represented by positive integers a and b, respectively, where $a, b \in [m]$. Let $\mathcal{X} := [2^N - 1]$ be a set of N-bit integers. $(\mathsf{Enc}_A, \mathsf{Dec}_A)$ is a public-key encryption of Alice. Hence, $\mathsf{Enc}_A : \mathcal{X} \to \mathcal{X}$ is an encryption under Alice's public key, and Dec_A is a decryption under Alice's private key.

$\langle 1 \rangle$ Bob selects a random N-bit integer $x \in \mathcal{X}$, and computes $c := \mathsf{Enc}_A(x)$ privately.
$\langle 2 \rangle$ Bob sends Alice the number $c - b + 1$
$\langle 3 \rangle$ For $i = 1, 2, \ldots, m$, Alice computes privately the values of $y_i = \mathsf{Dec}_A(c - b + i)$.
$\langle 4 \rangle$ Alice generates a random prime p of $N/2$ bits, and computes the values $z_i := y_i \bmod p$ for $i = 1, 2, \ldots, m$. If $|z_u - z_v| \geq 2$ for all distinct $u, v \in [m]$, then go to next step; otherwise generates another random prime and repeat the process until all z_u differ by at least 2;
$\langle 5 \rangle$ Alice makes $z' = (z_1, z_2, \ldots, z_a, z_{a+1} + 1, z_{a+2} + 1, \ldots, z_m + 1)$; each value is in the mod p sense.
$\langle 6 \rangle$ Alice sends Bob p and the vector z'.
$\langle 7 \rangle$ Bob looks at the b-th number in z'. If it is equal to $x \bmod p$, then $a \geq b$, otherwise $a < b$.
$\langle 8 \rangle$ Bob sends Alice the result.

Our Idea Behind Proposed Protocol I. We first point out that the key steps of Yao's protocol are $\langle 5 \rangle$–$\langle 7 \rangle$, where Alice *privately* adds 1 to z_{a+1} to z_m in the m-dimensional vector, and sends the vector to Bob. He *privately* checks the b-th value in the vector, and outputs the result. These private operation can be implemented by PP, which corresponds to the step $\langle 3 \rangle$ in the following proposed protocol I.

Note that, in Yao's solution, $\langle 1 \rangle$–$\langle 4 \rangle$ are necessary for realizing the key steps $\langle 5 \rangle$–$\langle 7 \rangle$ securely, since they prevent the vector z' in $\langle 5 \rangle$ from leaking Alice's wealth a to Bob. However, in a card-based cryptographic protocol, these steps can be replaced with *single step* since face down play the role of encryption. Furthermore, the communication in $\langle 8 \rangle$ can be removed in the card-based protocol since face-up cards on the tabletop can immediately be recognized by both Alice and Bob.

3.2 Proposed Protocol I

Based on the ideas discussed in the previous section, we propose a card-based cryptographic protocol for the millionaires' problem based on Yao's solution. We refer to this protocol *proposed protocol I.* The definitions of a, b, m are same in the previous section.

Proposed Protocol I (Card-based Yao's Solution)

(1) Alice prepares m pairs of ♣♡ and turn them all face down. This preparation is represented in a vector form as $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m)$ where $\boldsymbol{x}_1 = \boldsymbol{x}_2 = \cdots = \boldsymbol{x}_m =$ ♣♡.

(2) For $i = 1, 2, \ldots, m$, repeat the operation in which Alice swaps $\boldsymbol{x}_i$ if $i > a$; otherwise do not. Each swap operation must be executed *privately*, and it is described as the following PP with respect to $\mathcal{R}^{\mathsf{ic}}_2 := \{\pi_0, \pi_1\}$ which is given by (3) with $v = 1$:

$$\mathsf{PP}^{[2]}_{\mathcal{R}^{\mathsf{ic}}_2}(\boldsymbol{x}_i, \chi^{\mathsf{ge}}(i-1, a)),\ i = 1, 2, \ldots, m, \tag{6}$$

where $\chi^{\mathsf{ge}}(\cdot, \cdot)$ is defined in (4), i.e., $\chi^{\mathsf{ge}}(i-1, a) = 1$ iff $i > a$. As a result, Alice *privately* generates the sequence of cards $\boldsymbol{x}' := (\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_m)$ where $\boldsymbol{x}'_i := \mathsf{PP}^{[2]}_{\mathcal{R}^{\mathsf{ic}}_2}(\boldsymbol{x}_i, \chi^{\mathsf{ge}}(i-1, a))$.

(3) Alice sends Bob $\boldsymbol{x}'$.

(4) Bob *privately* moves $\boldsymbol{x}'_b$ to the first element of $\boldsymbol{x}'$, which is described as the following PP:

$$\mathsf{PP}^{[2m]}_{\mathcal{R}^{\mathsf{mf}}_{2m}}(\boldsymbol{x}', b-1) = \pi_{b-1}(\boldsymbol{x}') \tag{7}$$

where $\mathcal{R}^{\mathsf{mf}}_{2m} := \{\pi_i\}_{i=0}^{m-1}$ such that $\pi_i : \boldsymbol{x}' \mapsto (\boldsymbol{x}_{i+1}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \boldsymbol{x}_{i+2}, \ldots, \boldsymbol{x}_m)$.

(5) Bob reveals the left most commitment of $\mathsf{PP}^{[2m]}_{\mathcal{R}^{\mathsf{mf}}_{2m}}(\boldsymbol{x}', b-1)$, i.e., $\boldsymbol{x}'_b$. If the value represented by $\boldsymbol{x}'_b$ is 0, then $a \geq b$, otherwise $a < b$.

The remaining cards are completely randomized by Alice or Bob in public in order to discard the information of $\boldsymbol{x}'$ except for $\boldsymbol{x}'_b$. We call this operation "the remaining cards are *discarded*," hereafter.

Note that steps (1) and (2) in the proposed protocol correspond to step $\langle 1 \rangle$–$\langle 5 \rangle$, and the steps (3) and (4) correspond to steps $\langle 6 \rangle$ and $\langle 7 \rangle$, respectively, which shows that the step (2) considerably simplifies Yao's protocol. We omit the proof of correctness of the proposed protocol since it is almost obvious from Yao's protocol.

Note that $(\mathsf{Enc}_A, \mathsf{Dec}_A)$ in Yao's millionaires' protocol must be public-key encryption since a is obtained by Bob in step $\langle 5 \rangle$ if $(\mathsf{Enc}_A, \mathsf{Dec}_A)$ is a private key encryption. On the other hand, in the proposed protocol I, such leakage of a to Bob is prevented by requiring that all cards except $\boldsymbol{x}'_b$ are completely randomized by Alice or Bob publicly at the end of the protocol.

Efficiency of the proposed protocol I. In the proposed protocol, note that the constant numbers for PP and communication are sufficient. We use two PPs in steps (2) and (4), and one communication in step (3), and this outperforms the protocol based on logical gates (see Fig. 1). We note that the steps (4) and (5) are necessary so that Bob turns $\boldsymbol{x}'_b$ face up publicly without making b public[8].

We can show the perfect secrecy of the proposed protocol in the following theorem, but we omit the proof since it is almost obvious.

Theorem 1. *The proposed protocol I is perfectly secure; it satisfies* (5) *in Definition 1.*

Remark. Thanks to the special operations of card, e.g., face up, face down, and swap, etc., the proposed protocol I is not only a direct transformation of Yao's, but also is superior to the original protocol. For instance, the proposed protocol I does not use any randomness, whereas randomness is necessary for generating public/private keys in the original solution by Yao. Furthermore, it is worth observing that both Alice and Bob can know the output result simultaneously in the proposed protocol I, whereas Yao's original protocol, Bob is required to announce his result to Alice (see step $\langle 8 \rangle$).

4 Proposed Protocol II: Card-Based Cryptographic Protocol for Millionaires' Problem with Storage

4.1 Ideas Behind Proposed Protocol II

In order to reduce the number of cards to below $2m$, it is natural to represent the wealth of Alice and Bob as binary numbers with $\lceil \log m \rceil$ bits (i.e., $2\lceil \log m \rceil$ cards). This approach enables us to consider the strategy by comparing the Alice's and Bob's wealth bit-by-bit starting from their least significant bit although our strategy is not based on the use of logical gates.

Let $(a_n, \ldots, a_1)$ and $(b_n, \ldots, b_1)$ be the binary representation of the positive integers a and b, respectively, where $n := \lceil \log m \rceil$ and $a_i, b_i \in \{0, 1\}$, $i = 1, 2, \ldots, n$. For each $i \in [n]$, assume that a_i and b_i are represented by pairs of cards $\alpha_{i,l}\alpha_{i,r}$ and $\beta_{i,l}\beta_{i,r}$, respectively, where $\alpha_{i,l}\alpha_{i,r}, \beta_{i,l}\beta_{i,r} \in \{\clubsuit\heartsuit, \heartsuit\clubsuit\}$. For instance, $a_i = 1$ is represented by cards as $\alpha_{i,l}\alpha_{i,r} = \heartsuit\clubsuit$.

Note that, however, such a two-card representation of binary numbers is redundant in a bit-by-bit comparison since we can represent 0 and 1 by $\clubsuit$ and $\heartsuit$, respectively[9]. In this one-card representation, $\alpha_{i,l}$ and $\beta_{i,l}$ suffice to represent

[8] Private selection of $\boldsymbol{x}'_b$ and making it public are formally realized in this manner.

[9] However, we note that a one-card representation cannot express arbitrary binary numbers. Hence, $4\lceil \log m \rceil$ (i.e., $2\lceil \log m \rceil$ cards for Alice and Bob) cards are at least necessary when comparing two binary numbers less than m.

a_i and b_i, respectively. Further, their negations, $\neg a_i$ and $\neg b_i$, are also represented by $\alpha_{i,r}$ and $\beta_{i,r}$, respectively. In the following, we consider a scenario in which Alice prepares $(a_n, \ldots, a_i)$ by using a two-card representation, and then, Alice and Bob use a one-card representation for comparison.

We compare the bits of Alice and Bob by preparing a device (equipped by a card) called *comparison storage*, denoted by $cs \in \{\clubsuit, \heartsuit\}$, that records the bit-by-bit comparison results. Our idea is roughly as follows: We assume that Bob compares $\beta_{i,l}$ (i.e., b_i) with Alice's card $\alpha_{i,l}$ (i.e., a_i) from $i = 1$ to n, and he overwrites cs with $\beta_{i,l}$ (i.e., b_i) if $\beta_{i,l} \neq \alpha_{i,l}$ (i.e., $b_i \neq a_i$) while cs remains *untouched* if this is not the case (i.e., $b_i = a_i$). Recalling that Bob overwrites the comparison storage with *his* bit, Bob is shown to be *richer* if the comparison storage is $\heartsuit$ (i.e., 1) at the end of the protocol. Similarly, Alice is shown to be *richer* if the comparison storage is $\clubsuit$ (i.e., 0) at the end of the protocol. As is easily understood, however, this rough idea presents two problems:

(P1) If Bob were to directly compare his bits with those of Alice, such a comparison strategy would easily leaks Alice's bits to Bob.
(P2) The fact of overwriting the comparison result or not leaks Bob's bits to Alice.

(P1) can be avoided by considering the following modified randomized strategy: Since Alice prepares $(a_n, \ldots, a_i)$ by two-card representations, she sends Bob $\alpha_{i,l}$ (i.e., a_i) or $\alpha_{i,r}$ (i.e., $\neg a_i$) with probability 1/2. Such a randomization is effective for concealing the value of Alice's bit from Bob, but we encounter another problem:

(P3) Since Alice sends $\alpha_{i,w}$ to Bob $w \in \{l, r\}$ with a probability of 1/2, he cannot tell whether $a_i \neq b_i$ or not.

Problems **(P2)** and **(P3)** are simultaneously resolved by introducing another storage called *dummy storage*, denoted by $ds \in \{\clubsuit, \heartsuit\}$, and communicating the pair of cs and ds between Alice and Bob. Hereafter, we refer to the pair consisting of cs and ds as *storage*.

In order to resolve problem **(P2)**, it suffices for Bob to overwrite cs and ds corresponding to the results of $a_i \neq b_i$ and $a_i = b_i$, respectively, which enables him to hide his bit from Alice. However, due to **(P3)**, Bob cannot determine which one of cs and ds should be overwritten. Hence, we focus on how to resolve problem **(P3)** hereafter.

Problem **(P3)** can be rephrased using binary numbers as follows: Let $a'_i \in \{0, 1\}$ be a binary number that Bob receives, but he does not know whether $a'_i = a_i$ (in the case of $w = l$) or $a'_i = \neg a_i$ (in the case of $w = r$). Our main object is to find $a_i \neq b_i$ or $a_i = b_i$ even if either one of $a'_i = a_i$ or $a'_i = \neg a_i$ is sent[10].

Our basic idea for resolving **(P3)** is that Bob uses the fact that what he knows is either $\alpha_{i,w} \neq \beta_{i,l}$ or $\alpha_{i,w} = \beta_{i,l}$. Making use of this fact, Alice and Bob

[10] This problem is very similar to the well-known logical problem "*The Fork in the Road*," that is remarked upon later.

treat cs and ds as an *ordered pair* of face-down cards, and assume that either (cs, ds) or (ds, cs) is determined by *Alice's private random choice* $w \in \{l, r\}$ as follows:

- If Alice selects $w = l$ and sends Bob $\alpha_{i,l} \in \{\clubsuit, \heartsuit\}$ (i.e., a_i), then she sends him (cs, ds) with $\alpha_{i,l}$.
- If Alice selects $w = r$ and sends Bob $\alpha_{i,r} \in \{\clubsuit, \heartsuit\}$ (i.e., $\neg a_i$), then she sends him (ds, cs) with $\alpha_{i,r}$.

We can see that the order of cs and ds is synchronized with $w \in \{l, r\}$ (i.e., a_i and $\neg a_i$) in *Alice*. Owing to this synchronization, Bob can correctly overwrite cs only when $a_i \neq b_i$ by implementing the following strategy, even if he does not know which one of cs and ds should be overwritten. Let (σ_l, σ_r) be the storage Bob receives from Alice.

- If $\alpha_{i,w} \neq \beta_{i,l}$ (i.e., $a'_i \neq b_i$) holds, Bob overwrites *the left* element σ_l of the storage (σ_l, σ_r) with $\beta_{i,l}$ (i.e., b_i).
- If $\alpha_{i,w} = \beta_{i,l}$ (i.e., $a'_i = b_i$) holds, Bob overwrites *the right* element σ_r of the storage (σ_l, σ_r) with $\beta_{i,l}$ (i.e., b_i).

Let (σ'_l, σ'_r) be the storage overwritten by Bob, and he returns (σ'_l, σ'_r) to Alice. Then, by using $w \in \{l, r\}$ that Alice generated, she *privately* rearranges (σ'_l, σ'_r) so as to place cs and ds on the left and the right, respectively. After repeating these procedures from $i = 1$ to n, Bob is shown to be richer if $cs = \heartsuit$ (i.e., 1) whereas the contrary is true if $cs = \clubsuit$ (i.e., 0).

Table 2. Synchronization mechanism in the proposed protocol with storage

a_i ($\alpha_{i,l}$)	b_i ($\beta_{i,l}$)	$(cs, ds), w = l$			$(ds, cs), w = r$		
		a'_i ($\alpha_{i,l}$)	$a'_i \neq b_i$	Overwrite	a'_i ($\alpha_{i,r}$)	$a'_i \neq b_i$	Overwrite
0 ($\clubsuit$)	1 ($\heartsuit$)	0 ($\clubsuit$)	True	left $= cs$	1 ($\heartsuit$)	False	right $= cs$
1 ($\heartsuit$)	0 ($\clubsuit$)	1 ($\heartsuit$)	True	left $= cs$	0 ($\clubsuit$)	False	right $= cs$
0 ($\clubsuit$)	0 ($\clubsuit$)	0 ($\clubsuit$)	False	right $= ds$	1 ($\heartsuit$)	True	left $= ds$
1 ($\heartsuit$)	1 ($\heartsuit$)	1 ($\heartsuit$)	False	right $= ds$	0 ($\clubsuit$)	True	left $= ds$

It is easy to see from Table 2 that our synchronization strategy for storage works well. This is best clarified by discussing the proposed protocol by using binary numbers rather than cards. For instance, consider the case where Alice compares her bit $a_i = 1$ with Bob's bit $b_i = 0$ (the second line in Table 2). If Alice selects $w = l$, Bob receives a bit $a_i = 1$ and compares it with Bob's bit $b_i = 0$. Since $a'_i \neq b_i$, the left-hand side element of the storage, i.e., cs, is overwritten by $b_i = 0$. On the other hand, if Alice selects $w = r$, Bob receives a bit $a'_i = \neg a_i = 0$ and compares it with his bit $b_i = 0$. Since $a'_i = b_i = 0$, the right-hand side element of the storage, i.e., cs, is overwritten by $b_i = 0$. Anyway, cs is correctly overwritten by $b_i = 1$ ($> a_i = 0$) as expected.

Remark. It is interesting to note that the logic of the above synchronization strategy is the same as that of the well-known logical puzzle "*The Fork in the Road,*" [6, p. 25] (see footnote 7). Note that the point of the "The Fork in the Road" is that we need to obtain the correct answer (correct branch) from "*yes-no-questions,*" regardless of whether the native bystander tells the truth. Similarly, in our synchronization strategy, we require the correct compared result ($a_i \neq b_i$ or $a_i = b_i$) from "*same-or-different-questions,*" regardless of whether Bob receives $\alpha_{i,l}$ (i.e., a_i) or $\alpha_{i,r}$ (i.e., $\neg a_i$).

4.2 Proposed Protocol II

Based on the discussion in the previous section, we propose the card-based cryptographic protocol which uses storage and synchronization between the random selection $w \in \{l, r\}$ and the order of cs and ds, for the Millionaires' problem. For the upper bound $m \in \mathbb{N}$ of the wealth of Alice and Bob, let $n := \lceil \log m \rceil$.

Proposed Protocol II (Protocol for Millionaires' Problem with Storage)

(1) Alice prepares a face-down ♣ and a face-down ♡ (This card can be arbitrary since it is a dummy card.) as the output storage cs and the dummy storage ds, respectively. We call the pair consisting of cs and ds *storage.* She also prepares a sequence of $2n$ cards $(\alpha_{1,l}\alpha_{1,r}, \alpha_{2,l}\alpha_{2,r}, \ldots, \alpha_{n,l}\alpha_{n,r})$, which is a binary representation of her wealth $a \in [m]$. Bob also prepares the sequence of $2n$ cards $(\beta_{1,l}\beta_{1,r}, \beta_{2,l}\beta_{2,r}, \ldots, \beta_{n,l}\beta_{n,r})$, which is the binary representation of his wealth $b \in [m]$.

(2) For $i = 1, 2, \ldots, n$, repeat the following operations (2-i)–(2-v):

(2-i) Alice *privately* chooses $w \in \{l, r\}$ uniformly at random. Then, execute the following PP with respect to $\mathcal{R}_2^{\mathsf{ic}}$ which is defined in (3) with $v = 1$:

$$(\sigma_l, \sigma_r) := \mathsf{PP}^{[2]}_{\mathcal{R}_2^{\mathsf{ic}}}((cs, ds), \chi^{\mathsf{eq}}(w, r)) \tag{8}$$

where $\chi^{\mathsf{eq}}(w, r) = 1$ if $w = r$, and $\chi^{\mathsf{eq}}(w, r) = 0$ otherwise.

(2-ii) Alice sends Bob (σ_l, σ_r) in addition to $\alpha_{i,w}$.

(2-iii) Bob turns $\alpha_{i,w}$ face up, and he compares $\beta_{i,l}$ with $\alpha_{i,w}$ in *his mind.* If they are *different*, he *privately* overwrites σ_l with $\beta_{i,l}$, otherwise he *privately* overwrites σ_r with $\beta_{i,r}$. This operation can be described as the following PP with respect to $\mathcal{R}_2^{\mathsf{ow1}} := \{\pi_0, \pi_1\}$ where $\pi_0 := (\sigma_l, \beta_{i,l}, \sigma_r)$ and $\pi_1 = (\beta_{i,l}, \sigma_r, \sigma_l)$. :

$$(\sigma'_l, \sigma'_r, \eta) := \mathsf{PP}^{[3]}_{\mathcal{R}_2^{\mathsf{ow1}}}((\sigma_l, \sigma_r, \beta_{i,l}), \overline{\chi^{\mathsf{eq}}}(\beta_{i,l}, \alpha_{i,w})) \tag{9}$$

where $\overline{\chi^{\mathsf{eq}}}(\cdot, \cdot) := 1 - \chi^{\mathsf{eq}}(\cdot, \cdot)$. The extra card η is discarded without turning it face up.

(2-iv) Bob sends Alice (σ'_l, σ'_r).

(2-v) Alice rearranges the storage cards *privately* depending on the random value w chosen in (2-i), i.e., execute the PP such that

$$\mathsf{PP}^{[2]}_{\mathcal{R}^{\mathsf{ic}}_2}((\sigma'_l, \sigma'_r), \chi^{\mathsf{eq}}(w, l)), \tag{10}$$

which is used for the new storage cards (cs, ds).

(3) Alice turns cs face up to output. If the card is $\clubsuit$, then $a \geq b$. Otherwise, $a < b$. After completing the protocol, ds is discarded without revealing.

Example of proposed protocol II. We show a simple example for understanding how the proposed protocol II works correctly. Consider the case where we compare $a = 0$ of Alice and $b = 2$ of Bob, which are represented by $(\alpha_{1,l}\alpha_{1,r}, \alpha_{2,l}\alpha_{2,r}) := (\clubsuit\heartsuit, \clubsuit\heartsuit)$ and $(\beta_{1,l}\beta_{1,r}, \beta_{2,l}\beta_{2,r}) := (\heartsuit\clubsuit, \clubsuit\heartsuit)$, respectively, since $(a_1, a_0) = (0, 0)$ and $(b_1, b_0) = (1, 0)$. We also set $(cs, ds) = (\clubsuit, \heartsuit)$.

We first consider the case of $i = 1$. If Alice chooses $w = l$ in step (2-i), (8) becomes $(\sigma_l, \sigma_r) = (cs, ds) = (\clubsuit, \heartsuit)$ since $\chi^{\mathsf{eq}}(w, r) = \chi^{\mathsf{eq}}(l, r) = 0$. Then, she sends Bob $(\sigma_l, \sigma_r) = (\clubsuit, \heartsuit)$ and $\alpha_{1,l} = \clubsuit$ in step (2-ii). In step (2-iii), Bob compares $\beta_{1,l} = \clubsuit$ with $\alpha_{1,l} = \clubsuit$, which results in $\overline{\chi^{\mathsf{eq}}}(\beta_{1,l}, \alpha_{1,l}) = 0$. Then, he outputs $(\sigma'_l, \sigma'_r) = (\sigma_l, \beta_{1,l}) = (\clubsuit, \clubsuit)$ by overwriting the *right* element of $(\sigma_l, \sigma_r) = (\clubsuit, \heartsuit)$ with $\beta_{1,l} = \clubsuit$ *privately*, since (9) becomes $(\sigma'_l, \sigma'_r, \eta) = (\sigma_l, \beta_{1,l}, \sigma_r)$ due to $\overline{\chi^{\mathsf{eq}}}(\beta_{1,l}, \alpha_{1,l}) = 0$. Bob discards $\sigma_r = \heartsuit$.

On the other hand, consider the case where Alice chooses $w = r$ in step (2-i); Then, (8) in step (2-i) becomes $(\sigma_l, \sigma_r) = (ds, cs) = (\heartsuit, \clubsuit)$ since $\chi^{\mathsf{eq}}(w, r) = \chi^{\mathsf{eq}}(r, r) = 1$. She sends Bob $(\sigma_l, \sigma_r) = (\heartsuit, \clubsuit)$ and $\alpha_{1,r} = \heartsuit$ in step (2-ii). Bob compares $\beta_{1,l} = \clubsuit$ and $\alpha_{1,r} = \heartsuit$, and outputs $(\sigma'_l, \sigma'_r) = (\clubsuit, \clubsuit)$ by overwriting the *left* element of $(\sigma_l, \sigma_r) = (\heartsuit, \clubsuit)$ with $\beta_{1,l} = \clubsuit$ *privately* as a result of (9).

As a result, regardless of the selection of $w \in \{l, r\}$, storage becomes $(cs, ds) = (\clubsuit, \clubsuit)$, which means that the dummy storage is overwritten by the Bob's bit since $a_0 = b_0$. Then, Bob send it to Alice in step in (2-iv). In step (2-v), Alice sets $(cs, ds) := (\clubsuit, \clubsuit)$ due to (10) for the storage sent from Bob.

Next, consider the case of $i = 2$: If Alice selects $w = l$ in step (2-i), she generates $(\sigma_l, \sigma_r) = (cs, ds) = (\clubsuit, \clubsuit)$ from (8), and sends it with $\alpha_{2,l} = \clubsuit$ to Bob in step (2-ii). Then, Bob compares $\beta_{2,l} = \heartsuit$ with $\alpha_{2,l} = \clubsuit$ in step (2-iii). Since $\beta_{2,l} \neq \alpha_{2,l}$, he generates $(\sigma'_l, \sigma'_r) = (\beta_{2,l}, \sigma_r) = (\heartsuit, \clubsuit)$ by overwriting the *left* element of $(\sigma_l, \sigma_r) = (\clubsuit, \clubsuit)$ with $\beta_{2,l} = \heartsuit$ *privately* according to (9). Bob sends $(\sigma'_l, \sigma'_r) = (\heartsuit, \clubsuit)$ to Alice, and she obtains $(cs, ds) := (\heartsuit, \clubsuit)$ due to (10). Similar argument holds when Alice selects $w = r$, which is omitted here.

Finally, the output value correctly becomes $cs = \heartsuit$ as $a < b$ regardless of random choices of Alice.

Efficiency of the proposed protocol II. This protocol requires two communications for every bit therefore it requires $2\lceil \log m \rceil$ communications. We note that the sequence of PPs executed in steps (2-iii) and (2-iv) can be regarded as one PP. Similarly, steps (2-v) and (2-i), when i is incremented, can also be regarded as one PP. Hence, this protocol requires $2\lceil \log m \rceil$ PPs. The number of cards is $4\lceil \log m \rceil + 2$.

Theorem 2. *The proposed protocol II is perfectly secure; it satisfies* (5) *in Definition 1.*

Proof. First, consider the randomness used by Alice and Bob denoted by R_A and R_B, respectively. From step (2-i), the value of R_A is the choice of w which is randomly selected from $\{l, r\}$ with a probability of 1/2. Hence, R_A can obviously be simulated by S_A by using n independent uniform binary numbers. Similar to proposed protocol I, Bob does not use any randomness, and hence, S_A does not have to simulate R_B.

Then, considering the simulation of public information Λ which corresponds the face-up cards in step (2-iii), it is easy to see that Alice can generate λ by using a, i.e., her $2n$ cards, and the selection w. Hence, Λ is easily simulated by S_A. For Bob, $\alpha_{i,w}$ seems to be uniform over $\{\heartsuit, \clubsuit\}$ since he does not know the value of w selected randomly by Alice.

Therefore, simulators S_A and S_B exist, which completes the proof. □

5 Concluding Remarks

In this paper, we proposed two efficient card-based cryptographic protocols (called proposed protocols I and II) for the millionaires' problem by introducing a new operation called *private permutation* (PP). Proposed protocol I is constructed based on Yao's solution. This solution was realized by using public key encryption instead of logical gates, and hence, it could not be straightforwardly implemented to card-based cryptographic protocols based on logical gates. However, we show that Yao's solution can be easily implemented by using cards if we do not restrict ourselves by logical gates and use PPs instead. This protocol could be realized with one communication and two PPs, and is therefore much more effective than the existing protocol (see Table 1). However, the number of cards increases. It is worth mentioning that proposed protocol I is not only a direct transformation of Yao's protocol, but is also superior to the original protocol in the sense that randomness and the announcement of the result are not required as opposed to Yao's original protocol.

Proposed protocol II is entirely novel. It constitutes the communication of two types of storage for recording the compared result between two players. This proposed protocol is superior to the existing protocol based on logical gates with respect to the number of communications and PPs, whereas the number of cards is the same as the existing protocol. Furthermore, it is interesting to remark that proposed protocol II and the well-known logical puzzle known as "The Fork in the Road," are deeply related.

In the following, we briefly mention that proposed protocol II can be improved in two directions. Due to space limitations, the detailed explanation will appear at in the full version of the paper.

The first direction of improvement is the following: According to Table 1, proposed protocol II has not improved in terms of the number of cards. Hence, the first improvement is that proposed protocol II can be realized with only six cards. Our idea is that we do not need to represent the input as binary numbers by using $2\lceil \log m \rceil$ cards, but that it is sufficient to remember the input *in the player's mind*. Then, two cards are sufficient to represent the player's input since these two cards can be *reused*.

The second improvement is as follows: Proposed protocol II cannot be used for composing the other protocol[11] since each player is required to know his/her inputs. In order to resolve this, we can use an improved technique called *selection and substitution protocols* inspired by 6-card AND protocol [9]. Introducing this idea enables us to propose the card-based millionaires' problem while concealing the input and the output where the number of communications and PPs are almost 1/2 compared to the card-based cryptographic protocol for the millionaires' problem based on logical gates.

Acknowledgement. We are grateful to the anonymous reviewers for their careful reading of our manuscript and their insightful comments, which greatly improved the paper. This work was supported by JSPS KAKENHI Grant Numbers JP15H02710 and JP26420345.

References

1. Yao, A.: Protocols for secure computations. In: IEEE Symposium on FOCS, vol. 23, pp. 160–164. IEEE (1982)
2. Boer, B.: More efficient match-making and satisfiability *The Five Card Trick*. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 208–217. Springer, Heidelberg (1990). doi:10.1007/3-540-46885-4_23
3. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. of Cryptology **13**, 143–202 (2000)
4. Cramer, R., Dåmgard, I., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing. Cambridge University Press, Cambridge (2015)
5. Crépeau, C., Kilian, J.: Discreet solitary games. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 319–330. Springer, Heidelberg (1994). doi:10.1007/3-540-48329-2_27
6. Gardner, M.: Hexaflexagons and Other Mathematical Diversions: The First Scientific American Book of Puzzles and Games. University of Chicago Press, Chicago (1956)
7. Koch, A., Walzer, S., Härtel, K.: Card-based cryptographic protocols using a minimal number of cards. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 783–807. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48797-6_32
8. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. Int. J. Inf. Secur. **13**(1), 15–23 (2014)
9. Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: Deng, X., Hopcroft, J.E., Xue, J. (eds.) FAW 2009. LNCS, vol. 5598, pp. 358–369. Springer, Heidelberg (2009). doi:10.1007/978-3-642-02270-8_36

[11] Note that the term "composing" used here does not imply *composable security* [3].

10. Niemi, V., Renvall, A.: Secure multiparty computations without computer. Theoret. Comput. Sci. **191**(1, 2), 173–183 (1998)
11. Shinagawa, K., Mizuki, T., Schuldt, J.C.N., Nuida, K., Kanayama, N., Nishide, T., Hanaoka, G., Okamoto, E.: Multi-party computation with small shuffle complexity using regular polygon cards. In: Au, M.-H., Miyaji, A. (eds.) ProvSec 2015. LNCS, vol. 9451, pp. 127–146. Springer, Heidelberg (2015). doi:10.1007/978-3-319-26059-4_7