How to Solve Millionaires' Problem with Two Kinds of Cards

Takeshi Nakai¹ · Yuto Misawa¹ · Yuuki Tokushige¹ · Mitsugu Iwamoto¹ · Kazuo Ohta^{1,2}

Received: 13 May 2020 / Accepted: 7 October 2020 / Published online: 5 January 2021
© The Author(s) 2021

Abstract

Card-based cryptography, introduced by den Boer aims to realize multiparty computation (MPC) by using physical cards. We propose several efficient card-based protocols for the millionaires' problem by introducing a new operation called Private Permutation (PP) instead of the shuffle used in most of existing card-based cryptography. Shuffle is a useful randomization technique by exploiting the property of card shuffling, but it requires a strong assumption from the viewpoint of arithmetic MPC because shuffle assumes that public randomization is possible. On the other hand, private randomness can be used in PPs, which enables us to design card-based protocols taking ideas of arithmetic MPCs into account. Actually, we show that Yao's millionaires' protocol can be easily transformed into a card-based protocol by using PPs, which is not straightforward by using shuffles because Yao's protocol uses private randomness. Furthermore, we propose entirely novel and efficient card-based millionaire protocols based on PPs by securely updating bitwise comparisons between two numbers, which unveil a power of PPs. As another interest of these protocols, we point out they have a deep connection to the well-known logical puzzle known as "The fork in the road."

Introduction

Background and Motivation

Background Consider a scenario where n players $P_1, P_2, \dots, P_n$ hold private data $x_1, x_2, \dots, x_n$ respectively, and they wish to compute a value of a function $f(x_1, x_2, \dots, x_n)$ without revealing their own data. In this scenario, it is impossible to hide all information of the players' inputs, since the computed value $f(x_1, x_2, \dots, x_n)$

A preliminary version of this article appears in proceedings of 15th International Conference on Cryptology and Network Security (CANS2016) [15].

✉ Takeshi Nakai
t-nakai@uec.ac.jp

Extended author information available on the last page of the article

leaks certain information of their inputs. Hence, we require to hide the information of inputs other than the information of $f(x_1, x_2, \dots, x_n)$. The techniques for realizing such a requirement are called multiparty computation (MPC). MPC is one of the most important research topics in cryptography and information security. Actually, MPC has many cryptographic applications such as electronic auctions and electronic votings.

It is known that MPC can be realized by using several cards, and such a special implementation of MPC is called a card-based cryptography [4, 5]. Unlike general MPC protocols implemented on computers, the card-based cryptography is realized by manual operations. Obviously, manual operations take much longer time than the operations on computers. Hence, it is important to reduce the number of steps in the card-based cryptography. Based on a similar reason, smaller number of cards is desirable for an efficient implementation of card-based cryptography.

Much of the research related to card-based cryptography has been devoted to secure computation of logical gates such as AND and XOR,¹ since any computation can be implemented by their combinations. The central issue when designing efficient card-based cryptography 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 [11] reduced the number of cards to five in AND.² However, [11] assumes the existence of a special shuffle with non-uniform distribution, and hence, it is difficult to be implemented.³ Thus we adopt Mizuki–Sone protocols [9], which are easy to be implemented, as previous research in the efficiency evaluations.

The randomization is also important in order to realize card-based cryptography. In previous works, a randomization operation called shuffle is considered to be useful for card-based protocols with a smaller number of cards, and the usage of this operation has been extensively studied thus far.

We note that shuffles have two special features: The first feature is that a shuffle in a card-based cryptography specifies a certain permutation, whereas a shuffle in ordinary card games permutes the set of cards in a completely random manner. The second feature 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 described as follows: 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 protocols for logical gates utilizing shuffles:

¹ In card-based cryptography, OR operation is easily obtained from an AND operation since NOT operation can be easily implemented.

² We focus only on protocols for logical gates which can be used in combination with other protocols. Without considering the possibility of combination, it is shown in [11] that four cards are sufficient in AND.

³ A method to realize the non-uniform shuffle by using special boxes is proposed in [14].

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. Shuffles are too card-oriented operations. From the viewpoint of arithmetic MPC, they cannot be realized by a single operation but can be realized by at least two players to communicate with each other.⁴

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

(1): It is known that at least one shuffle is necessary for a randomize operation in realizing card-based cryptography. Actually, card-based protocols for logical gates, such as AND, OR and COPY [9], can be realized by using one random bisection cut. This fact implies that it is impossible to make the number of shuffles less than the number of logical gates as long as we construct card-based protocols by the combination of logical gates.⁵ For instance, consider the case of the millionaires' problem initiated by Yao's seminal work [2], 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 $m \in \mathbb{N}$ by logical gates can be realized as in Algo. 1, which involves $2\lceil \log m \rceil - 1$ AND and $2\lceil \log m \rceil - 2$ OR times.⁶ When executing these logical gates, the COPY operation [9] is necessary for copy $\neg a_i$ and b_i in each comparison of bits. Summarizing, $6\lceil \log m \rceil - 5$ random bisection cuts are necessary in total in order to implement Algo. 1.

We can expect this inefficiency to be resolved if we design a card-based cryptography specialized for the function computed in the protocol, although such improvement has not been studied intensively to date. Proceeding with this idea, it is natural to recall Yao's solution to the millionaires' problem [2] since it does not depend on logical gates but specializes in comparing two numbers privately. As we will see in Sect. 3.1, 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! Then, we can show a simple implementation of Yao's protocol by cards if we get rid of restriction of using logical gates.

(2): When we construct a card-based protocol with shuffles, it would be hard to use arithmetic MPC protocols straightforwardly because arithmetic MPCs normally assume private randomness whereas all operations in a card-based protocol with shuffles must be executed in public. This hardness would be an obstacle to construct card-based protocol for millionaires' problem based on Yao's original ideas, and motivates us to also employ the private randomness in card-based cryptography. Hence, in this paper, we explicitly allow such a private operation in card-based

⁴ Although this fact is mentioned in [11], the efficiency of the protocol based on this fact is not discussed by these authors, and hence, they use shuffles as building blocks.

⁵ After the earlier version of this paper [15] being published, a method was proposed to realize card-based cryptography for computing circuits with one shuffle [17]. Since this protocol utilized the techniques of garbled circuit [3], it requires distinct twenty four cards for each gate. Hence, instead of achieving the minimum number of shuffles, this protocol requires much more number of cards compared to the protocols obtained by simply combining the card-based protocols for logical gates.

⁶ Throughout this paper, the logarithmic base is 2.

cryptography.⁷ In most of previous works, all operations are performed in public so as to avoid cheating. On the other hand, since we adopt the operation of executing in player's private area, our card-based protocols are realized under the semi-honest model.

Algorithm 1 Comparing protocol constructed by logical gates

Input: $a = (a_n \cdots a_2 a_1)_2$, $b = (b_n \cdots b_2 b_1)_2$

$f_1 = \bar{a}_1 \wedge b_1$

for $i = 2$ to n **do**

$f_i = \bar{a}_i \wedge b_i \vee (\bar{a}_i \vee b_i) \wedge f_{i-1}$;

end for

Output: f_n

if $f_n = 0$ then $a \geq b$

if $f_n = 1$ then $a < b$

In previous works, a shuffle is considered as a building block for randomization, but actually, it is not a single operation from the viewpoint of arithmetic MPC. For instance, a random bisection cut 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 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 permutations in this example, respectively) are often used in arithmetic MPC. We call such a permutation behind one'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. Introducing PPs, evaluations of card-based protocols are almost parallel to those of arithmetic MPC. Concretely, the number of PPs and communications is considered as computational cost, and the number of cards is considered as memory cost.

Our Contributions

As shown above, the concept of a PP is motivated by (1) and (2). We propose two essentially different protocols for Millionaires' problem based on this concept, and call them proposed protocols I and II. 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.

---

⁷ Private operation in card-based cryptography is introduced independently by Marcedone et al. [8]

Table 1 Comparison of proposed card-based cryptography

Protocols	# of Comm.	# of PP	# of cards
Logical gates (Algorithm 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)	$2\lceil \log m \rceil$	$2\lceil \log m \rceil + 1$	$4\lceil \log m \rceil + 2$
Improvement of Proposed Protocol II	$2\lceil \log m \rceil$	$2\lceil \log m \rceil + 1$	6

We first construct a card-based cryptographic protocol for the millionaires' problem based on Yao's solution for two numbers less than or equal to m (proposed protocol I), which does not include logical gates. In the implementation, shuffles are not used, and PPs play a key role instead of shuffles. Even though this protocol is naïve, only one communication and two PPs are sufficient, which is a considerable improvement of card-based cryptography based on logical gates. On the other hand, the number of cards required by the protocol is $2m$, which is much worse than the protocol based on logical gates ($4\lceil \log m \rceil + 2$).

Not only transforming Yao's solution to card-based cryptography but also we propose an entirely new card-based cryptographic protocol based on PPs (proposed protocol II), which is specially developed to solve the millionaires' problem. Proposed protocol II is more efficient in terms of the number of communications, PPs, and cards. 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 the 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 idea of proposed protocol II is the same as that of the well-known logical puzzle "The fork in the road".⁸ This observation will be introduced when explaining the idea of the proposed protocol II in Sect. 4.1.

Unfortunately, proposed protocol II requires still the same number of cards with the previous work whereas the other measures are evidently improved. Hence, we discuss how to reduce the number of cards in Sect. 4.3. The main idea of this improvement is that inputs are not represented as the sequence of cards but are

⁸ 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? [1, p.25]

memorized in player's mind. This improvement enables the proposed protocol II to realize with only six cards.

This paper is the full version of [15]. We extended the earlier version [15] by proposing a new protocol in Sect. 4.3 that solves the millionaires' problem with only six cards. We also improved presentations, especially surveys are updated in Sect. 2.3.

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 cryptography in Sect. 2. In Sect. 3, the card-based cryptography for the millionaires' problem based on Yao's protocol is presented. Sect. 4 is devoted to the proposal of a new card-based cryptographic protocol with storage. We also show an improvement of the proposed protocol which reduces the number of cards. We summarize our results in Sect. 5.

Preliminaries

Notations and Basic Operations of Cards

In card-based cryptography, we normally use two types of cards such as $\clubsuit$ and $\heartsuit$. 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 $?$. 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 [12]. Note that these operations are executed publicly.

- Face up: $? \mapsto \clubsuit, ? \mapsto \heartsuit$
- Face down: $\clubsuit \mapsto ?, \heartsuit \mapsto ?$
- Permutation: e.g., $\clubsuit\clubsuit\heartsuit \mapsto \heartsuit\clubsuit\clubsuit$

A pair of face-down cards for the Boolean value $x \in \{0, 1\}$, is called commitment. In particular, the permutation for a commitment is referred to as swap.

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 cryptography based on logical gates, at least one shuffle such as a random bisection cut is necessary for every logical gate except negation, 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 $\mathbf{c} \in \{\clubsuit, \heartsuit\}^t$ be a vector consisting of t face-down cards. For a set $\mathcal{P}_t$ of all permutations over $[t] := \{1, 2, \dots, t\}$, let $\mathcal{R}_t \subset \mathcal{P}_t$ be a set of possible permutations. We also define $\mathcal{R}_t = \{\pi_0, \pi_1, \dots, \pi_{|\mathcal{R}_t|-1}\}$, where π_i denotes a permutation over $[t]$. Then, for a positive integer t and a set of possible permutations $\mathcal{R}_t$, the private permutation is defined as follows:

$$\text{PP}_{\mathcal{R}_t}^{[t]}(\mathbf{c}, s) := \pi_s(\mathbf{c}), \quad s = 0, 1, \dots, |\mathcal{R}_t| - 1. \quad (3)$$

Note that the same function was introduced in the previous works [11, 12] although we impose an additional assumption on this function. Namely, we assume that the player executing $\text{PP}_{\mathcal{R}_t}^{[t]}$ 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 was firstly introduced in the conference version of this paper [15] explicitly. We note that, not only the random bisection cut, but also several different types of *shuffles*, e.g., in [6], can be realized by PPs 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 , let $\mathcal{R}_{2v}^{\text{bc}} := \{\pi_0, \pi_1\} \subset \mathcal{P}_{2v}$ where

$$\pi_0 := (1, \dots, v, v+1, \dots, 2v), \text{ and } \pi_1 := (v+1, \dots, 2v, 1, \dots, v). \quad (4)$$

Eq. 4 means that $\pi_0(\mathbf{c}) = (\mathbf{u}_1, \mathbf{u}_2)$ and $\pi_1(\mathbf{c}) = (\mathbf{u}_2, \mathbf{u}_1)$ for $\mathbf{c} := (\mathbf{u}_1, \mathbf{u}_2)$ given by (1). Then, the random bisection cut for $2v$ cards is represented as $\text{PP}_{\mathcal{R}_{2v}^{\text{bc}}}^{[2v]}(\mathbf{c}, s) = \pi_s(\mathbf{c})$ 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 $\mathbf{c}$, Alice executes $\text{PP}_{\mathcal{R}_{2v}^{\text{bc}}}^{[2v]}(\mathbf{c}, r_A) =: \mathbf{c}'$ by using her private randomness $r_A \in \{0, 1\}$, and $\mathbf{c}'$ is sent to Bob. Bob also executes $\text{PP}_{\mathcal{R}_{2v}^{\text{bc}}}^{[2v]}(\mathbf{c}', r_B)$ by using his private randomness $r_B \in \{0, 1\}$.

⁹ In this paper, we define $[n] := \{1, 2, \dots, n\}$ for an integer $n \in \mathbb{N}$.

Protocol 1 6-card AND protocol [9] (using shuffle)

- 1) Set up the initial value $(a, 0, b)$ represented by the commitments of six cards.
- 2) Apply $\pi := (1, 3, 4, 2, 5, 6)$ to the sequence of cards prepared in the step 1).
- 3) Execute a random bisection cut for these six cards.
- 4) Apply $\pi^{-1} := (1, 4, 2, 3, 5, 6)$ to the sequence of cards obtained in step 3). Note that the result of 2)–4) is either $(a, 0, b)$ or $(-a, b, 0)$ with probability $1/2$.
- 5) Open the first bit. If it is 0, then output the second bit. Otherwise, output the third bit. Graphically, this step is represented as

Efficiency Measures. Most of the previous works, e.g., [12, 13], considered 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 arithmetic 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.

Example: 6-card AND Protocol

In order to clarify the difference between shuffles and PPs, we show two kinds of implementations of 6-card AND protocol, namely, we show the protocol realized by using shuffles (Protocol 1) and PPs (Protocol 2) respectively. Note that, all operations in protocols 1 are executed in public. On the other hand, there are both private and public operations in protocol 2, so that it needs to be clearly distinguished whether the operation is private or public.

We assume that two players, Alice and Bob, hold secret bits $a \in \{0, 1\}$ and $b \in \{0, 1\}$, respectively, and they wish to calculate $a \wedge b$ without revealing information of their inputs. We introduce the 6-card AND protocol [9] realizing this requirement.

Security Notion

Throughout this paper, we assume that both Alice and Bob are semi-honest players. Following [10], we introduce the security notion (perfect secrecy) of card-based cryptography for the millionaires' problem.

Protocol 2 6-card AND protocol [9] (using PPs)

- 1) Set up the initial value $c = (a, 0, b)$ represented by the commitments of six cards. First, one of both players, say Alice, holds these six cards.
- 2) Alice applies $\pi = (1, 3, 4, 2, 5, 6)$ to c in public. Let the sequence executed this permutation be c' .
- 3-i) Alice privately executes the following PP with respect to $\mathcal{R}_6^{\text{bc}}$ which is defined in (4) with $v = 3$.

$$c_A := \text{PP}_{\mathcal{R}_6^{\text{bc}}}^{[6]}(c', r_A) \quad (5)$$

where $\mathcal{R}_6^{\text{bc}} := \{\pi_0, \pi_1\}$ is given by (4) with $v = 3$ and $r_A \in \{0, 1\}$ is chosen uniformly at random.

- 3-ii) Alice sends c_A to Bob.
- 3-iii) Bob privately executes the following PP.

$$c_B := \text{PP}_{\mathcal{R}_6^{\text{bc}}}^{[6]}(c_A, r_B) \quad (6)$$

where $r_B \in \{0, 1\}$ is chosen uniformly at random.

- 4) Bob executes $\pi^{-1} = (1, 4, 2, 3, 5, 6)$ to c_B in public.
- 5) Bob reveals the first bit in public. If it represents 0, then output the 2nd bit. Otherwise, output the 3rd bit.

In defining the security of card-based cryptography, *view* plays a key role. View is roughly defined to be a vector of random variables¹⁰ to the data that each player can obtain in the protocol. Specifically, view includes the 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, which are the input for protocol. The common output of the millionaires' problem for Alice and Bob is represented as $\chi^{\text{ge}}(a, b)$ where

$$\chi^{\text{ge}}(u, v) := \begin{cases} 1 & \text{if } u \geq v \\ 0 & \text{otherwise,} \end{cases} \quad (7)$$

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

¹⁰ Throughout the paper, random variables are represented by capital letters. The probability that a random variable X takes a value x is represented by $\Pr\{X = x\}$ which is also written as $P_X(x)$ for short. Mathematically, a random variable is defined to be a map from probability space to the set of real numbers. However, for simplicity, we allow the cards $\clubsuit, \heartsuit$ to be treated as the values of random variables.

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) for PPs. The public information is the cards that Alice and Bob made public by turning them face up. Note that, in arithmetic MPC, view includes information that each player receives via private channel, but in card-based cryptography, 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 $(A, \chi^{\text{ge}}(A, B), R_A, \Lambda)$ and $(B, \chi^{\text{ge}}(A, B), R_B, \Lambda)$, 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 cryptography for the millionaires' problem as follows:

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

$$S_A(a, \chi^{\text{ge}}(a, b)) \stackrel{\text{perf}}{\equiv} (a, \chi^{\text{ge}}(a, b), R_A, \Lambda) \quad (8)$$

$$S_B(b, \chi^{\text{ge}}(a, b)) \stackrel{\text{perf}}{\equiv} (b, \chi^{\text{ge}}(a, b), R_B, \Lambda) \quad (9)$$

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

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

Our Idea Behind the Proposed Protocol I

We propose a card-based cryptography that resolves the millionaires' problem by cards based on Yao's original solution utilizing PPs.¹¹ Before providing our protocol, we explain Yao's public key based solution [2] as follows:

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,

¹¹ After the earlier version of this paper [15] was published, a shuffle-based protocol implementing Yao's solution was proposed in [18].

where $a, b \in [m]$. Let $\mathcal{X} := [2^N - 1]$ be a set of N -bit integers where, $2^{N/2} > 2m$ is necessary to hold $|z_u - z_v| \geq 2$ in step (4) for all distinct $u, v \in [m]$. $(\text{Enc}_A, \text{Dec}_A)$ is a public-key encryption of Alice. That is, $\text{Enc}_A : \mathcal{X} \rightarrow \mathcal{X}$ is an encryption under Alice's public-key, and Dec_A is a decryption under Alice's private-key.

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

Our Idea Behind Proposed Protocol I. We first point out that the key steps of Yao's protocol are (5)–(7), where Alice privately adds 1 to each of $z_{a+1}, z_{a+2}, \dots, z_m$ 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. This private operation can be implemented by PP in card-based cryptography, which corresponds to the step (3) in the following proposed protocol I.

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

Proposed Protocol I

Based on the ideas discussed in the previous section, we propose a card-based cryptography for the millionaires' problem based on Yao's solution (see Protocol 3). We refer to this protocol proposed protocol I. The definitions of a, b and m are the same as the previous section.

Note that steps 1) and 2) in the proposed protocol I correspond to steps (1)–(5), and the steps 3) and 4) correspond to steps (6) and (7), respectively, which show 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 $(\text{Enc}_A, \text{Dec}_A)$ in Yao's millionaires' protocol must be public-key encryption since a is obtained by Bob in step (6) if $(\text{Enc}_A, \text{Dec}_A)$ 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 $\mathbf{x}'_b$ are completely randomized in public by Alice or Bob at the end of the protocol.

Efficiency of the proposed protocol I In the proposed protocol, $2m$ cards are used. The number of PPs and communications is constant, i.e., it does not depend on the input length. 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 Algorithm 1). We note that the steps 4) and 5) are necessary so that Bob turns $\mathbf{x}'_b$ face up publicly without making b public.¹²

Theorem 1 *The proposed protocol I is perfectly secure; it satisfies (8) and (9) in Definition 1.*

Proof Consider the randomness used by Alice and Bob denoted by R_A and R_B , respectively. In this protocol, no randomness is used by Alice since she only swaps the cards by using a and m . Hence, it is not necessary for the simulator S_A to simulate R_A . We also find that Bob does not use his randomness, and R_B also need not be simulated by S_B .

Regarding the public value Λ , observe that it is only the cards $\mathbf{x}'_b$ revealed in step 5), and the binary value represented by $\mathbf{x}'_b$ is equal to the truth value of $a \geq b$. Hence, Λ is uniquely determined from the output, and it can obviously be simulated, which completes the proof. $\square$

Remark Thanks to the special operations of card-based cryptography, e.g., face up, face down, and swap, etc., proposed protocol I is not only a direct transformation of Yao's protocol, but also is superior to the original one from several aspects. 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 Bob is required to announce his result to Alice in Yao's original protocol (see step (8)).

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

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 number with $\lceil \log m \rceil$ bits (i.e., $2\lceil \log m \rceil$ cards).

¹² Private selection of $\mathbf{x}'_b$ and making it public are formally realized in this manner.

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 bits.

Let $(a_n, \dots, a_1)$ and $(b_n, \dots, b_1)$ be the binary representation of the positive integers a and b , respectively, where $n := \lceil \log m \rceil$ and $a_i, b_i \in \{0, 1\}$, $i = 1, 2, \dots, n$. 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 number is redundant in a bit-by-bit comparison since we can represent 0 and 1 by $\clubsuit$ and $\heartsuit$, respectively.¹³ In this one-card representation, $\alpha_{i,l}$ and $\beta_{i,l}$ suffice to represent 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 $(a_n, \dots, a_1)$ by using a two-card representation, but here, 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 as well) called comparison storage, denoted by $cs \in \{\clubsuit, \heartsuit\}$, that records the bit-by-bit comparison results. Our idea is roughly described 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 has the following a problem:

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.

This problem can be avoided by considering the following modified randomized strategy: Since Alice prepares $(a_n, \dots, a_1)$ 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:

P2) Since Alice sends $\alpha_{i,w}$ to Bob $w \in \{l, r\}$ with a probability of $1/2$, he cannot tell from $\alpha_{i,w}$ whether $a_i \neq b_i$ or not.

Problem **P2)** is 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.

¹³ 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 arbitrary two binary numbers less than m .

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

Hereafter, we refer to the pair consisting of cs and ds as storage. Bob overwrites cs and ds corresponding to the result of $a_i \neq b_i$ and $a_i = b_i$. However, just adding a new storage is insufficient to resolve the problem that Bob cannot determine whether $a_i \neq b_i$ or $a_i = b_i$, i.e., he cannot determine which one of cs and ds should be overwritten.

Problem **P2**) 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.¹⁴

Basic idea for resolving **P2**) is that Bob uses the fact that what he knows is either $\alpha_{i,w} \neq \beta_{i,l}$ (i.e., $a'_i \neq b_i$) or $\alpha_{i,w} = \beta_{i,l}$ (i.e., $a'_i = b_i$). Making use of this fact, Alice and Bob 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 $\alpha_{i,r}$.

Note that $\alpha_{i,w}$ is not necessary to be face-down when she sends it since no information on a leaks to Bob from $\alpha_{i,w}$. 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. Then Bob's behavior after receiving $\alpha_{i,w}$ from Alice is shown below.

- 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).

¹⁴ This problem is very similar to the well-known logical problem "The fork in the road," that is remarked upon later.

- 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).

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 $a'_i = a_i = 1$ 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., ds , is overwritten by $b_i = 0$. Anyway, cs is correctly overwritten by $b_i = 0$ ($< a_i = 1$) 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,” [1, p.25] (see footnote *8). 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. The one of the well-known answers to this puzzle is that the logician should ask “if I ask the right way goes to the village, then do you answer YES?.” This question consists of two propositions as follows:

- Q1)** The right way goes to the village.
Q2) The bystander answers YES.

Suppose that the right way goes to the village. If the native bystander is a truth-teller, then obviously the logician receives YES. On the other hand, the logician have the same answer (YES) even if the bystander is a liar because of the following double negation:

- L1)** The liar wants to say NO to **Q1**) because **Q1**) is true.
L2) The liar has to say YES because **Q2**) is false (due to **L1**)).

Namely, telling lies twice for **Q1**) and **Q2**), the liar says YES if the right way goes to the village; NO otherwise. Our synchronization strategy has the same structure with this puzzle. In proposed protocol II, Alice chooses whether she sends a_i (i.e., truth) or $\neg a_i$ (i.e., lie), which corresponds to **L1**). If she chooses the lie, then she reverses the order of storage cards cd and ds , which has the same effect with **L2**).

Due to this structure of double negation, Bob can correctly record the comparison result regardless of Alice's choice. Therefore, we can verify the correctness of proposed protocol II.

Proposed Protocol II

Based on the discussion in the previous section, we propose the card-based cryptography which uses storage and synchronization between the random selection $w \in \{l, r\}$ and the order of cs and ds, for the Millionaires' problem (see Protocol 4). For the upper bound $m \in \mathbb{N}$ of the wealth of Alice and Bob, let $n := \lceil \log m \rceil$.¹⁵

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 $(\alpha_{2,l}\alpha_{2,r}, \alpha_{1,l}\alpha_{1,r}) := (\clubsuit\heartsuit, \clubsuit\heartsuit)$ and $(\beta_{2,l}\beta_{2,r}, \beta_{1,l}\beta_{1,r}) := (\heartsuit\clubsuit, \clubsuit\heartsuit)$, respectively, since $(a_2, a_1) = (0, 0)$ and $(b_2, b_1) = (1, 0)$. We also set $(cs, ds) = (\clubsuit, \heartsuit)$.

We first consider the case of $i = 1$. If Alice chooses $w = l$ in step (2-i), (12) becomes $(\sigma_l, \sigma_r) = (cs, ds) = (\clubsuit, \heartsuit)$ since $\chi^{\text{eq}}(w, r) = \chi^{\text{eq}}(l, r) = 0$. 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 $\chi^{\text{eq}}(\beta_{1,l}, \alpha_{1,l}) = 0$. 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 (13) becomes $(\sigma'_l, \sigma'_r, \eta) = (\sigma_l, \beta_{1,l}, \sigma_r)$ due to $\chi^{\text{eq}}(\beta_{1,l}, \alpha_{1,l}) = 0$. Bob discards σ_r without face up $\sigma_r = \heartsuit$.

On the other hand, consider the case where Alice chooses $w = r$ in step (2-i); Then, (12) in step (2-i) becomes $(\sigma_l, \sigma_r) = (ds, cs) = (\heartsuit, \clubsuit)$ since $\chi^{\text{eq}}(w, r) = \chi^{\text{eq}}(r, r) = 1$. She sends Bob $(\sigma_l, \sigma_r) = (\heartsuit, \clubsuit)$ and $\alpha_{1,r} = \heartsuit$ in step (2-ii). Bob compares $\beta_{1,l} = \clubsuit$ with $\alpha_{1,r} = \heartsuit$, 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 (13).

Summarizing the case of $i = 1$, 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 $a_1 = b_1$. Then, Bob sends it to Alice in step (2-iv). In step (2-v), Alice sets $(cs, ds) := (\clubsuit, \clubsuit)$ due to (14) 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 (12), 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 (13). Bob sends $(\sigma'_l, \sigma'_r) = (\heartsuit, \clubsuit)$ to Alice, and she obtains $(cs, ds) := (\heartsuit, \clubsuit)$ due to (14).

On the other hand, consider the case where Alice chooses $w = r$ in step (2-i); Then, she generates $(\sigma_l, \sigma_r) = (ds, cs) = (\clubsuit, \clubsuit)$ from (12), and sends it with $\alpha_{2,r} = \heartsuit$ in step (2-iii). Since $\beta_{2,l} = \alpha_{2,r}$, he generates $(\sigma'_l, \sigma'_r) = (\sigma_l, \beta_{2,l}) = (\clubsuit, \heartsuit)$ by overwriting the right element of $(\sigma_l, \sigma_r) = (\clubsuit, \clubsuit)$ with $\beta_{2,l} = \heartsuit$ *privately* according to (13). Bob sends $(\sigma'_l, \sigma'_r) = (\clubsuit, \heartsuit)$ to Alice, and she obtains $(cs, ds) := (\heartsuit, \clubsuit)$ due to (14).

¹⁵ This card can be arbitrary since it is a dummy card.

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 steps (2-v) and (2-i), when i is incremented, can also be regarded as one PP. Hence, this protocol requires $2\lceil \log m \rceil + 1$ PPs. The number of cards is $4\lceil \log m \rceil + 2$.

Theorem 2 *The proposed protocol II is perfectly secure; it satisfies (8) and (9) in Definition 1.*

Proof Consider the randomness used by Alice and Bob denoted by R_A and R_B , respectively. From step (2-i), the value of $R_A = (W_1, W_2, \dots, W_n)$ where W_i is the random variable corresponding to w in the i -th loop in step 2). Each random variable W_i , $i = 1, 2, \dots, n$ takes the value in $\{l, r\}$ with probability $1/2$, and it is independent from the other random variables. Hence, R_A can obviously be simulated by S_A by using n independent uniform binary numbers. Similarly to the proposed protocol I, Bob does not use any randomness, and hence, S_B does not have to simulate R_B .

Regarding the simulation of public information Λ , observe that Λ is the n values represented by the face-up cards in step (2-iii), i.e., $\Lambda = (\alpha_{1,W_1}, \alpha_{2,W_2}, \dots, \alpha_{n,W_n})$. Hence, Λ is easily simulated by S_A by a which is represented by her $2n$ cards, and the random variable $R_A = (W_1, W_2, \dots, W_n)$. On the other hand, for Bob, $\Lambda = (\alpha_{1,W_1}, \alpha_{2,W_2}, \dots, \alpha_{n,W_n})$ seems to be uniformly distributed over $\{\heartsuit, \clubsuit\}^n$ since he does not know the value of $W_i = w_i$, which is selected randomly by Alice. Hence, Λ is easily simulated by S_B .

Since the simulators S_A and S_B can be explicitly constructed as above, we complete the proof of the theorem. $\square$

Millionaires' Problem Can Be Solved with Only Six Cards

Although the proposed protocol II was improved in terms of the numbers of PPs and communications, as is shown in Table 1, it still requires the same number of cards with the previous work based on logical gates. However, we show that the proposed protocol II can be realized with only six cards in this section.

Our main idea is to reuse the card η discarded by Bob in step (2-iii) of Protocol 4.¹⁶ We note that, however, η cannot be simply reused. If η is reused simply and accessed by Alice and/or Bob, they can obtain information about η which holds information on their inputs. For instance, in Protocol 4, suppose that Bob could look at the front of η in step (2-iii) for $i = 1$. Noticing that $(cs, ds) = (\clubsuit, \heartsuit)$ is public information when $i = 1$, a_1 completely leaks to Bob since he can tell whether $w = l$

¹⁶ In our setting, the randomization for reusing η is executed by participants, Alice and Bob. If we are allowed to outsource this randomization to a trusted third party, the number of cards can further reduced, which was pointed out in [16].

or not. Inductively, η should not be simply reused when $i \geq 1$ because η also holds information about Alice's choice of w .

Therefore, we need to erase information about η for reusing it. However, it is impossible to erase the information about η as long as we adopt the one-card representation since a single card cannot be randomized as opposed to the case of two-card representation. Hence, we should employ two-card representation for the storage (and the input) in Protocol 4. Concretely, if η is in two-card representation, Bob returns randomized η to Alice instead of discarding it by Bob in Protocol 4. Then, she can reuse η .

One may think that this modification makes the protocol inefficient since the number of storage cards increases. However, surprisingly, this modification allows Alice and Bob to use η as his/her inputs if they hold inputs in their mind! Namely, at the cost of using six cards for (cs, ds) and η , Alice and Bob do not necessary to have their cards to represent their inputs (if they can remember the inputs). Therefore, six cards are sufficient to realize a card-based protocol for millionaires' problem with efficient memory and communication cost. The improved protocol shown in Protocol 5. As shown in the step 1), the storage cards are represented by two-card representation. The steps (2-v) and (2-viii) are executed for erasing information of η by Bob and Alice.

Efficiency of the improvement of 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-iv) and (2-v) can be regarded as one PP. Similarly, steps (2-vii) to (2-i), when i is incremented, can also be regarded as one PP. Hence, this protocol requires $2\lceil \log m \rceil + 1$ PPs.

Concluding Remarks

In this paper, we proposed two efficient card-based cryptography (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 cryptography 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 consists of 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. But this protocol is not made efficient from the view point of the number of cards (see Table 1). Hence we proposed a method to reduce the number of cards. The improved protocol works only six cards by memorizing the inputs in player’s mind without representing them using cards.

Protocol 3 Proposed Protocol I: Card-based Yao’s Solution

- 1) Alice prepares m pairs of $\clubsuit\heartsuit$ and turn them all face down. This preparation is represented in a vector form as $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m)$ where $\mathbf{x}_1 = \mathbf{x}_2 = \dots = \mathbf{x}_m = (\clubsuit, \heartsuit)$.
- 2) For $i = 1, 2, \dots, m$, repeat the operation in which Alice swaps $\mathbf{x}_i$ if $i > a$; otherwise does not. Each swap operation must be executed *privately*, and it is described as the following PP with respect to $\mathcal{R}_2^{bc} := \{\pi_0, \pi_1\}$ which is given by (4) with $v = 1$:

$$\text{PP}_{\mathcal{R}_2^{bc}}^{[2]}(\mathbf{x}_i, \chi^{\text{ge}}(i-1, a)), \quad i = 1, 2, \dots, m, \quad (10)$$

where $\chi^{\text{ge}}(\cdot, \cdot)$ is defined in (7), i.e., $\chi^{\text{ge}}(i-1, a) = 1$ iff $i > a$. As a result, Alice *privately* generates the sequence of cards $\mathbf{x}' := (\mathbf{x}'_1, \mathbf{x}'_2, \dots, \mathbf{x}'_m)$ where $\mathbf{x}'_i := \text{PP}_{\mathcal{R}_2^{bc}}^{[2]}(\mathbf{x}_i, \chi^{\text{ge}}(i-1, a))$.

- 3) Alice sends $\mathbf{x}'$ to Bob.
- 4) Bob *privately* moves $\mathbf{x}'_b$ to the first element of $\mathbf{x}'$, which is described as the following PP:

$$\text{PP}_{\mathcal{R}_{2m}^{\text{mf}}}^{[2m]}(\mathbf{x}', b-1) = \pi_{b-1}(\mathbf{x}') \quad (11)$$

where $\mathcal{R}_{2m}^{\text{mf}} := \{\pi_i\}_{i=0}^{m-1}$ such that $\pi_i : (1, 2, \dots, m) \mapsto (i+1, 1, 2, \dots, i, i+2, \dots, m)$.

- 5) Bob reveals the left most commitment of $\text{PP}_{\mathcal{R}_{2m}^{\text{mf}}}^{[2m]}(\mathbf{x}', b-1)$, i.e., $\mathbf{x}'_b$. If the value represented by $\mathbf{x}'_b$ 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 $\mathbf{x}'$ except for $\mathbf{x}'_b$. We call this operation “the remaining cards are *discarded*,” hereafter.

Protocol 4 Proposed Protocol II: Protocol for Millionaires' Problem with Storage

---

- 1) Alice prepares a face-down $\clubsuit$ and a face-down $\heartsuit^{*15}$ as the comparison 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 $(\alpha_{n,l}\alpha_{n,r}, \dots, \alpha_{2,l}\alpha_{2,r}, \alpha_{1,l}\alpha_{1,r})$, which is a binary representation of her wealth $a \in [m]$. Bob also prepares the sequence of $2n$ cards $(\beta_{n,l}\beta_{n,r}, \dots, \beta_{2,l}\beta_{2,r}, \beta_{1,l}\beta_{1,r})$, which is the binary representation of his wealth $b \in [m]$.

- 2) For $i = 1, 2, \dots, n$, 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 $\mathcal{R}_2^{bc}$ which is defined in (4) with $v = 1$:

$$(\sigma_l, \sigma_r) := \text{PP}_{\mathcal{R}_2^{bc}}^{[2]}((cs, ds), \chi^{\text{eq}}(w, r)) \quad (12)$$

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

- (2-ii) Alice sends Bob (σ_l, σ_r) in addition to $\alpha_{i,w}$. Here, $\alpha_{i,w}$ need not be face down.

- (2-iii) Bob 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 $\beta_{i,l}$. This operation can be described as the following PP with respect to $\mathcal{R}_3^{\text{owl}} := \{\pi_0, \pi_1\}$ where $\pi_0 : (1, 2, 3) \mapsto (1, 3, 2)$ and $\pi_1 : (1, 2, 3) \mapsto (3, 2, 1)$:

$$(\sigma'_l, \sigma'_r, \eta) := \text{PP}_{\mathcal{R}_3^{\text{owl}}}^{[3]}((\sigma_l, \sigma_r, \beta_{i,l}), \overline{\chi^{\text{eq}}}(\beta_{i,l}, \alpha_{i,w})) \quad (13)$$

where $\overline{\chi^{\text{eq}}}(\cdot, \cdot) := 1 - \chi^{\text{eq}}(\cdot, \cdot)$. 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., executes the PP such that

$$\text{PP}_{\mathcal{R}_2^{bc}}^{[2]}((\sigma'_l, \sigma'_r), \chi^{\text{eq}}(w, l)), \quad (14)$$

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.
-

Protocol 5 Improvement of Proposed Protocol II

- 1) Alice prepares two face-down ♣ and two face-down ♥ as the storage. First, let $cs = 0, ds = 0$, i.e., cs and ds are expressed with two cards respectively. She also prepares one ♣ and one ♥.
- 2) For $i = 1, 2, \dots, n$, repeat the following operations.
  - (2-i) Alice chooses $\alpha_i \in \{0, 1\}$ uniformly at random privately. Then, execute the following permutation:

$$(\sigma_l, \sigma_r) := \text{PP}_{\mathcal{R}_4^{\text{bc}}}^{[4]}((cs, ds), \overline{\chi^{\text{eq}}}(\alpha_i, a_i)) \quad (15)$$

- (2-ii) Alice turns remaining two cards face up and makes α_i with them keeping on face up.
  - (2-iii) Alice sends Bob (σ_l, σ_r) in addition to the two cards representing α_i .
  - (2-iv) Bob compares b_i with α_i in his mind. If they are different, he replaces σ_l with b_i privately, otherwise he replaces σ_r with b_i privately. Then, b_i is made of the two cards which is used for representing α_i . This operation can be described as the following PP with respect to $\mathcal{R}_6^{\text{ow2}} := \{\pi_0, \pi_1\}$ where $\pi_0 : (1, 2, 3, 4, 5, 6) \mapsto (1, 2, 5, 6, 3, 4)$ and $\pi_1 : (1, 2, 3, 4, 5, 6) \mapsto (5, 6, 3, 4, 1, 2)$:

$$(\sigma'_l, \sigma'_r, \eta) := \text{PP}_{\mathcal{R}_6^{\text{ow2}}}^{[6]}((\sigma_l, \sigma_r, b_i), \overline{\chi^{\text{eq}}}(b_i, \alpha_i)) \quad (16)$$

- (2-v) Bob executes the following operation for erasing information about the storage.

$$\eta' := \text{PP}_{\mathcal{R}_2^{\text{bc}}}^{[2]}(\eta, r_b) \quad (17)$$

where $r_b \in \{0, 1\}$ is chosen uniformly at random.

- (2-vi) Bob sends $(\sigma'_l, \sigma'_r, \eta')$ to Alice.
  - (2-vii) Alice rearranges storage cards privately depending on the random value α_i chosen in (2-i) as

$$\text{PP}_{\mathcal{R}_4^{\text{bc}}}^{[4]}((\sigma'_l, \sigma'_r), \overline{\chi^{\text{eq}}}(\alpha_i, a_i)), \quad (18)$$

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

- (2-viii) Alice executes the following operation for erasing information about the storage.

$$\eta'' := \text{PP}_{\mathcal{R}_2^{\text{bc}}}^{[2]}(\eta', r_a) \quad (19)$$

where $r_a \in \{0, 1\}$ is chosen uniformly at random.

- 3) Alice turns cs face up to output. If $cs = 0$, then $a \geq b$. Otherwise, $a < b$. After completing the protocol, ds is discarded without revealing.
-

Acknowledgements We are grateful to Prof. Takaaki Mizuki and Dr. Kazumasa Shinagawa for their valuable comments and discussion. We thank the anonymous reviewers, whose comments have helped us to improve the presentation of this paper. This work was supported by JSPS KAKENHI Grant numbers JP26420345, JP17H01752, JP18H05289, JP18K19780, JP18K11293, JP18H03238 and JP20J21248.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Gardner, M.: Hexaflexagons and Other Mathematical Diversions: The First Scientific American Book of Puzzles and Games. Univ of Chicago Press, Chicago (1956)
2. Yao, A.: Protocols for secure computations. In: IEEE Symposium on FOCS, vol. 23, pp. 160–164, IEEE (1982)
3. Yao, A.: How to generate and exchange secrets (extended abstract). In: 27th Annual Symposium on Foundations of Computer Science, pp. 162–167 (1986)
4. den Boer, B.: More efficient match making and satisfiability: the five card trick. Lecture Notes in Computer Science, vol. 434. Springer, pp. 208–217 (1990)
5. Crépeau, C., Killian, J.: Discreet solitary games. In: CRYPTO'93, Proceedings of the 13th annual international cryptology conference on Advances in cryptology. vol. LNCS773. Springer, pp. 319–330 (1994)
6. Niemi, V., Renvall, A.: Secure multiparty computations without computer. Theor. Comput. Sci. **191**(1, 2), 173–183 (1998)
7. Canetti, R.: Security and composition of multiparty cryptographic protocols. J. Cryptol. **13**, 143–202 (2000)
8. Marcedone, A., Wen, Z., Shi, E.: Secure Dating with Four or Fewer Cards. IACR Cryptology ePrint Archive, vol. 2015/1031, <https://eprint.iacr.org/2015/1031> (2015)
9. Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: FAW 2009, vol. LNCS 5598. Springer, pp. 358–369 (2009)
10. Cramer, R., Dămgård, I., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing. Cambridge University Press, Cambridge (2015)
11. Koch, A., Walzer, S., Härtel, K.: Card-based cryptographic protocols using a minimal of cards. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology–ASIACRYPT 2015, LNCS, vol. 9452 and 9453. Springer, pp. 783–807 (2015)
12. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. Int. J. Inf. Secur. **13**(1), 15–23 (2014)
13. Shinagawa, K., Mizuki, T., Schuldt, J., Nuida, K., Kanayama, N., Nishide, T., Hanaoka, G., Okamoto, E.: Multi-party computation with small shuffle complexity using regular polygon cards. In: International Conference on Provable Security, ProvSec2015, LNCS, vol. 9451. Springer, pp. 127–146 (2015)
14. Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: An Implementation of Non-Uniform Shuffle for Secure Multi-Party Computation. In: International Workshop on ASIA Public-Key Cryptography, pp. 49–55 (2016)
15. Nakai, T., Tokushige, Y., Misawa, Y., Iwamoto, M., Ohta, K., Efficient Card-based Cryptographic Protocols for Millionaires' Problem Utilizing Private Permutations. In 15th International Conference on Cryptology and Network Security, CANS2016, LNCS, vol. 10052. Springer, pp. 500–517 (2016)

16. Ono, H., Manabe, Y., Efficient card-based cryptographic protocol for the millionaires' problem using private input operations. In: 13th Asia Joint Conference of Information Security, AsiaJ-CIS2018, IEEE (2018)
17. Shinagawa, K., Nuida, K., A single Shuffle Is Enough for Secure Card-based Computation of Any Circuit. In Cryptography ePrint Archive, vol. 2019/380. <https://eprint.iacr.org/2019/380> (2019)
18. Miyahara, D., Hayashi, Y., Mizuki, T., Sone, H.: Practical card-based implementations of Yao's millionaire protocol. Theor. Comput. Sci. **803**, 207–221 (2020)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

**Takeshi Nakai¹ · Yuto Misawa¹ · Yuuki Tokushige¹ · Mitsugu Iwamoto¹ · Kazuo Ohta^{1,2} **

Yuto Misawa
m.yuto@uec.ac.jp

Yuuki Tokushige
yuuki.tokushige@uec.ac.jp

Mitsugu Iwamoto
mitsugu@uec.ac.jp

Kazuo Ohta
kazuo.ohta@uec.ac.jp

¹ Graduate School of Informatics and Engineering, The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan

² Cyber Physical Security Research Center, National Institute of Advanced Industrial Science and Technology, Tokyo, Japan