PRIVATE PERMUTATIONS IN CARD-BASED CRYPTOGRAPHY

TAKESHI NAKAI

THE UNIVERSITY OF ELECTRO-COMMUNICATIONS

GRADUATE SCHOOL OF INFORMATICS AND ENGINEERING

A DISSERTATION SUBMITTED FOR DOCTOR OF PHILOSOPHY IN ENGINEERING

March, 2021

PRIVATE PERMUTATIONS IN CARD-BASED CRYPTOGRAPHY

SUPERVISORY COMMITTEE

CHAIRPERSON:	ASSOCIATE PROFESSOR MITSUGU IWAMOTO
MEMBER:	PROFESSOR HIROSHI YOSHIURA
MEMBER:	PROFESSOR KAZUO SAKIYAMA
MEMBER:	ASSOCIATE PROFESSOR TAKESHI SUGAWARA
MEMBER:	EMERITUS PROFESSOR KAZUO OHTA

Contents

Abstract

Multi-Party Computation (MPC) is a cryptographic protocol that enables parties to compute a function while keeping their data secret. Although standard MPC is constructed with algebraic procedures and is supposed to be implemented in computers, there are also MPCs that are implemented with physical tools instead of computers. Our study deals with card-based cryptography that realizes MPC using physical cards such as playing cards. In this thesis, we use two types of cards, $\boxed{\clubsuit}$ and $\boxed{\heartsuit}$, whose backsides are the same, for constructing protocols.

Traditional card-based cryptography is based on the operating model that assumes all operations are performed publicly, such as on the table. This model has the advantage of preventing a cheat since all operations are monitored between the players. However, this model also causes the lower bound of the number of cards for a protocol. The assumption that publishes all operations restricts a method of expressing the input value secretly to use face-down cards. Then, at least $2n$ cards are required for an n-bit input protocol since two cards are necessary to arbitrarily express a 1-bit value. This model also requires randomizing operations to be done in public. Traditional card-based cryptography utilizes "shuffle" to achieve confidentiality even under the condition, which publishes all operations. Shuffle is a card-oriented randomizing operation that requires its result cannot be identified by all players, including the player who performed it. On the other hand, algebraic

MPC usually adopts "private randomness" that assumes each player can privately generate and use a random number. Shuffle's assumption has a large gap from private randomness. Thus, card-based cryptography's framework is different from algebraic MPC.

In this thesis, we propose a new card-based cryptography model that introduces "private permutations (PP)," which is similar to private randomness of algebraic MPC. Our model allows a player to use the private area, such as the behind player's back, and PP is an operation to permute a card order privately. Then, PP removes the restriction of having to use face-down cards for input. That is, in our model, there is a possibility to construct a protocol with fewer cards than the lower bound of traditional card-based cryptography. We actually propose several protocols to achieve this.

This thesis is consists of six chapters. Chapter 1 is the descriptions of the background of MPC and card-based cryptography. PP is proposed, and the notations are introduced in Chapter 2. Chapter 6 is the conclusion of this thesis. Chapters 3–5 are summarized as follows.

Chapter 3: Secure Computing Logic Gates – Logic Fate Protocols Whose Number of Cards is Less Than The Lower Limit of Traditional Model

In traditional card-based cryptography, an n-bit input protocol requires a minimum of $2n$ cards. On the other hand, PP enables the input value to be expressed without using face-down cards. Thus PP enables us to construct a protocol with fewer cards than the lower bound in the traditional card-based cryptography. Chapter 3 shows that 2-bit input AND, OR, and XOR protocols can be realized with less than four cards using PP as the input representation. Note that AND protocol is proposed by Marcedone, Wen, and Shi [29].

Chapter 4: Secure Computing Threshold Function – Efficient Card-based Protocols for Majority Votings And The Threshold Function

The three-card AND and OR protocols described in Chapter 3 can be extended to a 2-input 2-output protocol that simultaneously obtains AND and OR results, with four cards. We show that 3-input majority voting can be realized by utilizing this four-card AND/OR protocol without any additional cards. We also propose a (t, n)-threshold function protocol with $n+2$ cards by extending our 3-input majority voting protocols. This protocol is realized with fewer cards than the lower bound $2n$ in traditional card-based cryptography. This result thanks to the input representation by PP, as in Chapter 3. It also reduces the number of PPs and communications from $4n^2$ to n and $2n^2$ to $n-1$. PP achieves efficiency not only for basic functions, such as logic gates but also for advanced functions.

Chapter 5: How to Solve Millionaires' Problem – Efficient Card-based Millionaires' Protocols

To introduce PP, the model of card-based cryptography is closer to that of algebraic MPC, compared to the one based on shuffle. As a result, it is expected that ideas can be mutually utilized between card-based cryptography and algebraic MPC, which have been independently studied. We demonstrate that a new card-based protocol can be obtained by converting Yao's (algebraic) *millionaires' protocol* into a card-based protocol. Millionaires' protocol is a $2m$-bit input protocol that aims at comparing two m-bit values. Our proposed protocol can be easily derived if we understand the essence of his solution and is considerably simplified, thanks to the card's property.

Our protocol, obtained from Yao's protocol, is superior to the existing protocol regarding the number of PPs and communications. However, the number of cards exponentially increases. Thus, we propose another new millionaires' protocol based on the bitwise comparison. This protocol is also interesting in the sense that it uses the famous logic puzzle, "The fork in the road." Although this protocol succeeded in reducing the number of cards from our protocol based on Yao's protocol, it does not reduce below the traditional lower bound of cards, i.e., $4m$.

We can further reduce the number of cards to six by reusing the cards in our proposed protocol. This protocol clarifies that the millionaires' protocol can be realized with only six cards. This protocol is the most straightforward to show the potential of PP power.

Abstract (in Japanese)

マルチパーティ計算 (MPC) は，複数の参加者がそれぞれ持つ情報を秘匿したまま，参加者同士で協調してそれらの情報を入力値とした関数の計算を行う暗号プロトコルである．一般的な MPC は，計算機への実装が想定されている代数的なプロトコル（以下，代数的 MPC）であるが，計算機を用いずに物理的な道具を用いて構成される MPC も提案されている．その中で，本研究はトランプのような物理的なカードを用いて MPC を実現するカードベース暗号を扱う．用いるカードは $\boxed{\clubsuit}$ と $\boxed{\heartsuit}$ の 2 種類で，裏にすると絵柄が同じで区別ができないものとする．

既存のカードベース暗号は，すべての操作を（テーブル上などの）公開の場で行う操作モデルを仮定する．この操作モデルでは，プレイヤの不正なふるまいを考慮する必要がない利点がある一方で，入力値を秘匿して表現する方法が裏にしたカードを用いる方法に限定される．1-bit の表現には 2 枚のカードを用いる必要があるため，n-bit 入力のプロトコルには少なくとも $2n$ 枚のカードが必要である．

また，この操作モデルはランダマイズも公開の場で行うことを要求する．この要求の下で，秘匿性を実現する手法として「シャッフル」と呼ばれるカード特有のランダマイズ操作が用いられている．シャッフルは公開の場で行われる操作であるが，この操作で生成された乱数は操作を行ったプレイヤ本人を含めて，すべてのプレイヤに秘匿されることを要求する．この仮定は，代数的 MPC で通常採用される内部乱数モデル（各プレイヤが内部でプライベートに乱数を生成する）とは構成原理が大きく異なっている．

本論文では，カードベース暗号に，通常の MPC の内部乱数の使用と類似する「秘匿置換」を導入した新たな操作モデルを提案する．秘匿置換は，プレイヤが内部でプライベートに乱数を生成し，他のプレイヤに見えないようにプレイヤの背に隠すなどして，その乱数に応じた置換を行う操作である．このモデルでは，秘匿置換を用いた入力が可能となるため，入力に 2 枚のカードを用いる必要がない．その結果，従来モデルにおけるカード枚数の下限値を下回る枚数でプロトコルを構成できることをいくつかの例で示す．

本稿は全 6 章で構成される．第 1 章は MPC とカードベース暗号の背景説明である．第 2 章は，秘匿置換の導入および記法の定義などの準備である．第 6 章は本論文のまとめである．3 章から 5 章では，秘匿置換を用いて構成したプロトコルの提案を行う．具体的な内容は以下の通りである．

第 3 章: 従来モデルにおける下限を下回るカード枚数の論理演算プロトコルの提案

従来の操作モデルにおけるカード MPC では，入力値の 1-bit 毎に 2 枚のカードを用いるため，n-bit 入力のプロトコルには最低 $2n$ 枚のカードが必要であった．本研究で提案する操作モデルでは，秘匿置換を入力に用いることが可能である．その結果，入力値をカード 2 枚で表現しなければならない制約がなくなり，従来モデルにおけるカード枚数の下限値を下回るプロトコルを構築することが可能となった．第 3 章では，従来モデルでは最低 4 枚必要な，2 入力の論理演算プロトコル AND と OR が，秘匿置換を導入した提案モデルにおいては 3 枚のカードで実現（計算）可能であることを示す（AND は Marcedone–Wen–Shi [29] による）．また，XOR については，2 枚のカードで実現可能であることを示す．

第 4 章: 多数決および閾値関数を計算するカード MPC の提案

第 3 章で示した AND，OR プロトコルは，秘匿置換を用いて入力値表現を工夫することで，使用カード 4 枚で，AND と OR の結果を同時に得る 2 入力 2 出力のプロトコルへ拡張することができる．第 4 章では，この AND/OR 同時計算プロトコルが，カードを追加することなく 3 入力多数決プロトコルへ応用できることを示す．さらに，この 3 入力多数決プロトコルを拡張する

ことで，従来モデルでは $2n+2$ 枚のカードが必要であったしきい値関数計算プロトコルを $n+2$ 枚で実現できることを示す（n は入力数）．従来モデルでは n 入力のプロトコルには最低でも $2n$ 枚のカードが必要であるが，秘匿置換を導入することで，提案のしきい値関数計算プロトコルはその下限値のほぼ半分の枚数で実現することができる．これは，3 章と同様に，入力値をカードで表現するのではなく，秘匿置換で表現したことが効率化の決め手となっている．また，秘匿置換および通信回数に関しても，それぞれ $4n^2$ を n へ，$2n^2$ を $n-1$ へ削減することができる． この結果から，秘匿置換は論理演算のような基礎プロトコルのみでなく，より高度な関数を計算するプロトコルに対しても効率化が達成できることがわかる．

第 5 章: 効率的なカードベース金持ち比べプロトコルの提案

内部乱数より自然に導入される秘匿置換をカードベース暗号に導入したことにより，カードベース暗号は代数的 MPC により近いモデルとなった．それにより，独立的に研究が進められてきたカードベース暗号と代数的 MPC 間で相互にテクニックを活用できるようになることが期待される．本章では実際に，代数的 MPC である 2 つの m-bit 値の Yao の大小比較（金持ち比べ）プロトコルを秘匿置換を用いたカードベース暗号に変換して，新たなプロトコルを構築する．変換は Yao のプロトコルの本質を用いればほぼ自明なものであり，オリジナルのプロトコルを非常に単純化している．

Yao の金持ち比べから得た提案プロトコルは，秘匿置換および通信回数では効率化に成功している一方で，カード枚数が指数的に増えてしまう課題があった．そこで本章では，ビット毎の大小比較に基づく，新しい大小比較プロトコルを提案する．提案プロトコルは，有名な論理パズル “The fork in the road” を用いる点でも興味深い．最終的に，金持ち比べプロトコルをたった 6 枚のカードで実現できることを明らかにした．また，秘匿置換および通信回数に関しても，それぞれ $12m-10$ を $2m+1$ へ，$6m-5$ を $2m$ へ削減する．これは秘匿置換の有効性を示す最も強力な例となっている．

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Mitsugu Iwamoto, for his continuous support and valuable comments. His insightful words always improved my dissertation and presentation. I would also like to thank him for the feedback and support on my career decision. I am also very thankful to Professor Kazuo Ohta for his enthusiastic guidance. He taught me the mind that I should have as a researcher. I am sure that this dissertation would not have been possible without his guidance and persistent help.

Special thanks go to Professor Yohei Watanabe for his sincere support. He sincerely consulted me not only in my research but also in my career path. I am also deeply grateful to Professor Takaaki Mizuki at Tohoku University, who gave me valuable advice for my research.

I also would like to express my appreciation for the supervisory committee members, Professor Hiroshi Yoshiura, Professor Kazuo Sakiyama, and Professor Takeshi Sugawara. They gave me a lot of valuable advice and suggestions to improve this dissertation.

I want to offer my special thanks to my collaborated researchers, Mr. Yuuki Tokushige, Mr. Yuto Misawa, and Mr. Satoshi Shirouchi. The wonderful discussion with them gave me an essential inspiration and sophisticated my research. I am also grateful to my laboratory members for providing me right environment for study. Especially, Dr. Kazumasa Shinagawa's comments made me aware of my deficiencies and improved my dissertation and presentation.

I owe my deepest gratitude to the University of Electro-Communications and medical personnel for providing me with an appropriate study environment even in the difficult situation during the COVID-19 pandemic. Finally, it is my pleasure to thank my parents for their support and encouragement for my education.

January, 2021
Takeshi Nakai
Tokyo, Japan

Chapter 1

Introduction

1.1 Background on Multi-Party Computation

Cryptography is one of the essential technologies for utilizing the Internet securely. For instance, the symmetric/public key cryptosystem provides secure communications over the Internet, even in an environment with the adversary. On the other hand, modern cryptography aims to realize not only secure communications but also secure computations among distinct parties.

Consider a scenario where distinct parties wish to compute a function, such as the average and summation, using their private data as the input values. More precisely, we suppose the case where n parties $P_1, P_2, \dots, P_n$ hold private data $x_1, x_2, \dots, x_n$, respectively, and they wish to compute the value of a function $f(x_1, x_2, \dots, x_n)$ without revealing their own data. *Multi-Party Computation* (MPC) is a cryptographic technology that realizes their wishes. MPC has to fulfill the following two requirements: First, every party must receive a correct output (correntness). Second, it is necessary that parties cannot get any information other than the output value (pricacy). Both *correctness* and *privacy* can be easily fulfilled if there is a trusted third party that takes over the functionality. However, the goal of MPC is to fulfill the requirements without it, in other words, even if among distrustful parties. MPC protocols are utilized in various applications, such as electronic auction [18,42] and electronic voting [5,6].

The first MPC protocol is the "mental poker" proposed by Shamir, Rivest, and Adleman in 1979 [53]. It aims to play poker fairly between two people at a distance over the phone. The theoretical foundation of MPC in the two-party setting was established by Yao [61] in 1982. After that, Goldreich, Micali, and Wigderson extended it to the n-party setting [21].

We consider two kinds of adversarial settings, *semi-honest* and *malicious* models.

In the semi-honest adversarial model, it is assumed that adversaries correctly follow the protocol procedures, but they attempt to extract information about the other parties' inputs from legitimately obtained information. Hence, this model is also called *honest but curious*.

On the other hand, the malicious adversarial model allows adversaries to behave out of protocol procedures. More robust security is guaranteed under the malicious adversarial model rather than the semi-honest adversarial model.

It is known that achievability of such models depends on the ratio of corrupted parties, as shown in [4,10,21,50]. These results hold in the *stand-alone* model that assumes parties to participate in only one protocol and run it only once. However, it is usual that several different protocols run at the same time in the modern implementation. It is known that a protocol that is secure in the stand-alone model is not always secure under the composition. A model that guarantees security, even in such a case, is referred to as *Universal Composability* (UC) [7,8]. It is known that any protocols cannot achieve UC security without the setup assumptions, such as a common reference string and a public key infrastructure [9].

There are roughly two directions when constructing an MPC protocol.

One direction is a generic construction, and the other is a specialized construction. The generic construction aims to build a protocol that is available for an arbitrary function. The famous protocol in this direction is the *garbled circuit*, proposed by Yao [62], which is applicable for any Boolean function in the two-party setting. It is based on the fact that a combination of logic gates can compute any Boolean function. In this direction, MPC protocols based on *secret sharing* [52] and *homomorphic encryption* [16,20,49] are also known. For instance, GMW protocol [21], which is available for any functions in the n-party setting, is based on secret sharing. Also, Ben-or, Goldwasser, and Wigderson extended it to the *information-theoretic* model [4].[*1]

On the other hand, specialized construction aims to build a protocol for a specific function straightforwardly. The protocols based on this direction tend to be more efficient than generic solutions. For instance, we consider Yao's solution for the *millionaires' problem* [61], which determines which of two values is greater without revealing them, is included in this direction (see Section 5.2.1 for details). The protocols for mental poker [21,53], electronic auction [18], and electronic voting [5,6] are followed in this direction as well.

1.2 Multi-Party Computation Using Physical Objects

So far, we reviewed the history of standard (or *algebraic*) MPC that is realized by computers algebraically. Although algebraic MPC is constructed to be implemented in computers, there are MPCs that are implemented with physical objects. Such protocols are referred to as *physical cryptography* [24] or *recreational cryptography* [3]. We list several examples of these MPC pro-

[*1]In this model, the adversary has no computational limit. On the other hand, a model that limits the ability of adverwaries in (probabilistic) polynomial time Turing machine is referred to as the *computational model.*

tocols in the following.

- Physical cards (card-based cryptography) [12, 15, 22, 43, 57]
- Envelopes [17, 39, 40]
- Cups [17]
- PEZ dispenser [2, 3, 41]
- Dial lock [33]
- 15 puzzle [34]
- Random-looking images printed on transparencies (visual cryptography) [13, 14]
- Physical coins [28]

Utilizing physical assumptions may enable us to design the protocols that cannot be achieved in the algebraic MPC. Because physical cryptography can be implemented with human hands and are easy to understand, they attract research and are useful in education. In this thesis, we focus on card-based cryptography.

1.3 Background on Card-Based Cryptography

Card-based cryptography, proposed by den Boer in 1989 [15], realizes secure computations by using physical cards. It also employs simple operations as used in general card games such as permutation, turn face-up/down and shuffle. Two types of cards, such as $\boxed{\clubsuit}$ and $\boxed{\heartsuit}$, are used in general card-based cryptography. The backsides of cards are all the same. The number of cards and shuffles are the standard efficiency measures of a card-based protocol.

A Boolean value is usually represented with the format, $0 \mapsto \boxed{\clubsuit}\boxed{\heartsuit}$ and $1 \mapsto \boxed{\heartsuit}\boxed{\clubsuit}$. This format is convenient for NOT operation since it can be easily realized by swapping. An n-bit input protocol requires at least $2n$ cards when we adopt this format to the input representation, such as $\boxed{?}\boxed{?} \in \{0, 1\}$. We call such a method of expressing the input value using face-down cards "commitment."

It is essential in card-based cryptography that all operations are assumed to take place *in public*, like on a table. This assumption has the advantage of preventing a cheat since all operations are monitored between the players. However, this assumption limits the input representation to use the commitment since there is no other way to express an input value secretly. Thus, n-bit input protocol requires at least $2n$ cards since at least two cards are necessary for 1-bit representation, e.g., $0 \mapsto \boxed{\clubsuit}\boxed{\heartsuit}$ and $1 \mapsto \boxed{\heartsuit}\boxed{\clubsuit}$. Even if the Boolean values are represented as $0 \mapsto \boxed{\clubsuit}$ and $1 \mapsto \boxed{\heartsuit}$, each player must possess at least two cards since input values need to be arbitrarily selected.[*2]

Card-based cryptography has been devoted to secure computation of logic gates such as AND, XOR, and COPY. OR operation is easily obtained from an AND operation using NOT operation. Thus, any computation can be implemented by a combinations of these logic gate protocols [51]. When we express a function by a combination of logic gates, it is important whether or not the protocols can be composed with other protocols. Consider the case where card-based protocol X is composed of another protocol Y. Then the output of X must be taken over to Y as the input keeping the value secret, i.e., face-down.

Committed format [12, 43, 57] is a useful definition to simplify the dis-

[*2] A study exists to use information on the top and bottom of marks such as $\clubsuit$ and $\rotatebox{180}{$\clubsuit$}$ [37]. We do not apply this method and assume that there is only one card orientation.

cussion about the composition of card-based protocols. We refer that card-based protocol is in committed format for an encoding ϕ, e.g., $0 \mapsto \boxed{\clubsuit}\boxed{\heartsuit}$ and $1 \mapsto \boxed{\heartsuit}\boxed{\clubsuit}$, if both input and output formats follow ϕ. On the other hand, a protocol with different input and output formats is called a non-committed format protocol. A non-committed format protocol is possible but not appropriate for the composition with other protocols.

Overview of Previous Works: One of the central issue when designing efficient card-based protocols is to minimize the number of cards required in the protocol. den Boer proposed a five-card AND protocol in the non-committed format [15]. Mizuki, Kumamoto, and Sone showed that AND protocol in the non-committed format can be done with four-card [35]. As mentioned above, the input value is represented by two cards. Therefore, their solution is optimal with respect to the number of cards.

The concept of committed format was proposed by Crépeau and Killian with actually showing ten-card AND and fourteen-card XOR protocols in the committed format [12]. The optimal XOR protocol in committed format was proposed by Mizuki and Sone [38]. They proposed six-card AND protocol in the same paper, i.e., there is room for improvement on the AND protocol. It was one of the important open problems in card-based cryptography whether AND protocol can be realized with a smaller number of cards in the committed format. Koch, Walzer, and Härtel proved this problem [27]. In this work, they proposed five-card AND protocol, which always terminates with a fixed number of steps, and four-card AND protocol, which is the Las Vegas algorithm,[*3] in committed format. In addition, they proved that AND protocol in committed format could not be realized with four cards without using the Las Vegas algorithm. In other words, their 5-card AND protocol is optimal as a protocol with a finite number of steps.

[*3]Las Vegas algorithm is a randomized algorithm that outputs either the correct result or information about the failure.

Efficiency has been improveed for each logic gate, but at least one shuffle is required for one logic gate[*4]. Thus, a protocol composed of a combination of logic gates tends to have a large number of shuffles. There may be more efficient protocols that are specialized for each function.

Formulation of Operations in Card-Based Cryptography: There are three operations used in general card-based cryptography: "permutation," "turn face-down/up," and "shuffle." As mentioned above, all operations are assumed to be performed in public, including randomizing operattions. However, in algebraic MPC, we achieve a secure protocol by utilizing the *private randomness*, which assume each player can generate and use random number privately. Thus, it seems that the confidentiality cannot be achieved if all operations are shown to other players. Shuffle is the key technique to solve this problem, which is the randomizing operation performed in public. Then, it is requires that none of the players can identify the result, including the player performed the shuffle. Mizuki–Shizuya [36] formalized this operation as follows:

For a card order $(\alpha_1, \alpha_2, \ldots, \alpha_n)$, the shuffle operation $\mathrm{shuffle}_{\Pi,\mathcal{F}}$ is a random variable defined as

$$\mathrm{shuffle}_{\Pi,\mathcal{F}}(\alpha_1, \alpha_2, \ldots, \alpha_n) = \pi_r(\alpha_1, \alpha_2, \ldots, \alpha_n) \tag{1.1}$$

where $\Pi = \{\pi_1, \pi_2, \ldots, \pi_t\}$ is a set of permutations and $\mathcal{F}$ is a probability distribution on Π. $r \in \{1, \ldots, t\}$ is determined according to $\mathcal{F}$. Then, none of the players identify r that expresses which permutation is selected.

For instance, den Boer [15] used *random cut* that is cyclic shift shuffle

[*4]There is also a method to realize card-based protocol for computing circuits with *one* shuffle [56]. Since this protocol utilized the techniques of garbled circuit [62], 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.

to realize the five-card AND protocol. This shuffle is on the following set of permutations:

$$\Pi = \{\mathrm{id}, (5,1,2,3,4), (4,5,1,2,3), (3,4,5,1,2), (2,3,4,5,1)\}. \tag{1.2}$$

$\mathcal{F}$ is the uniform distribution over Π. Crépeau and Killian [12] also used the random cut to construct ten-card AND and fourteen-card XOR protocols in the committed format. It is known that such shuffles that are closed about permutations and have a uniform distribution can be performed with human hands [23, 26]. Thus, it is desirable that the shuffle is closed and uniform [1, 25]. However, even if the shuffle can be performed with human hands, we need to verify whether it is secure even in public.

In order to reduce the number of cards, a different type of shuffle was introduced in [35, 38], called *random bisection cut*. In executing *random bisection cut*, even number of cards are divided into two sets consisting of the same number of cards, and these two sets are exchanged many times until *none* of the players can recognize how many times the two sets of cards are permuted. Hence, π is $\{\mathrm{id}, (v+1, v+2, \ldots, 2v, 1, 2, \ldots, v)\}$ where $2v$ is the number of cards and $\mathcal{F}$ is the uniform distribution over Π. We note that the random bisection cut has only two results, unlike the random cut. Such a shuffle seems unnatural from general card games' viewpoint, and there is room to discuss whether it can be *securely* implemented with human hands [59, 60].

Furthermore, Koch, Walzer, and Härtel introduced unique shuffles whose result is non-uniform to reduce the number of cards in the AND protocols [27]. For instance, they used a shuffle whose set of permutations is similar to random bisection cut, but the probabilities were non-uniform such as 1/3 and 2/3. Although their shuffle certainly contribute to protocols' efficiency,

Table 1.1: Comparison of MPC Models

	Algebraic MPC	Public Model	(Our) Private Model
Enc/Dec	Enc/Dec	Turn Face-down/up	Turn Face-down/up
Randomization	Private	Public	Private
Communication	Use	Not use	Use

it is much harder to be implemented with human hands.[*5]

1.4 Motivation

As explained in the previous section, much of previous works for card-based cryptography are based on the operating model that assumes all operations are performed in public, such as on the table. We call this model *public model*. Public model restricts the expression of input values secretly to use face-down cards. That is, at least $2n$ cards are required for an n-bit input protocol since two cards are necessary to express a 1-bit value arbitrarily. While public model has the advantage of preventing a player's malicious behavior, it causes a lower bound of the number of cards.

Public model also requires randomizing operations to be done in public. Normally, a probabilistic operation must be executed privately since it is necessary to conceal the random number from other players. On the other hand, traditional card-based cryptography achieves confidentiality even under public model by using "shuffle" based on the card-oriented assumption.

Discussion about Shuffles: Shuffle is the most critical operation in card-based cryptography, which is to randomize a card order in public. den Boer showed how to compute securely AND protocol utilizing the random cut, which is one of the shuffles. This shuffle seems practically feasible with human hands. On the other hand, the special shuffles, such as random bisection cut

[*5]A method to realize the non-uniform shuffle by using special boxes is proposed in [46].

and non-uniform shuffles, have a human infeasibility problem. The methods of human hands implementation were proposed using tools, such as boxes [46] and a rubber [60], but it is not preferable if we want to execute protocols only by hands. Such shuffles are mathematically acceptable but not physically acceptable and thus seem not to be included in card-oriented assumptions. Also, shuffle is based on the assumption that its result cannot be identified by human eyes. We note that the protocol's security cannot be guaranteed if video filmed shuffle.

Algebraic MPC realizes secure protocols based on private randomness, which is the assumption that each player can privately generate and use a random number. On the other hand, card-based cryptography realizes MPC protocols without private randomness by utilizing the physical assumption, i.e., by using shuffles. Thus, card-based cryptography framework becomes different from algebraic MPC's one because of this card-oriented assumption. Thus, card-based cryptography has been studied independently to algebraic MPC and has constructed protocols from scratch.

In traditional card-based cryptography, protocols for logic gates have been the central concern. This is because a combination of logic gates can compute any Boolean function, and it is necessary to build a protocol from scratch. The difference in frameworks of algebraic MPC and card-based cryptography under the public model makes card-based cryptography difficult. Thus, almost all traditional card-based cryptography concentrated on logical gates, and there are few studies on the specialized construction for advanced functions.

Although the combination of logic gates has the advantege of being general, there may be a specific and more efficient construction for each function. Thus, we focus on the specialized construction for each function to discover a more efficient protocol.

Proposal of Private Permutations: We propose a new card-based cryptography model that is naturally derived from the private randomness of algebraic MPC. Our model, called *private model*, allows players to use private area, such as under the table or behind the player's back.*6 That is, we introduce the following two operations into card-based cryptography instead of shuffle:

- Private Permutation (PP): Permute card order in private area
- Communication: hand over (or send) cards to another player.

In private model, shuffle can be interpreted as a combination of multiple operations under the semi-honest assumption. For instance, we realize a random bisection cut 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.

Similarly, other shuffles can be achieved with two PPs and one communication. Thus, a protocol constructed on public model can also be realized on private model.

Expected Results on Private model: In private model, operations, including input, can be performed privately like algebraic MPC. As a result, it is possible to break the lower bound of the number of cards in traditional card-based cryptography. However, security is weakened in the sense that semi-honest assumption is required.

Card-based cryptography becomes closer to algebraic MPC by removing the physical assumption of shuffle and introducing private randomness. Table 1.1 summaries the correspondences between the operations of each model and

*6Private randomness in card-based cryptography is introduced independently by Marcedone, Wen, and Shi [29]. Their protocol is described in Section 3.2.

algebraic MPC. From this table, we can see that our private model is closer to algebraic MPC.

As mentioned above, public model requires that all operations, including randomizations, are performed in public. This requirement has a large gap with the private randomness of algebraic MPC. On the other hand, our private model enables the players to perform privately similar to the private randomness. It is noteworthy that public model does not have communication. This is because it is no matter who performs each process in a situation where all operations are performed in public. Surprisingly, all operations can be executed by one player in public model. This property has a large gap with algebraic MPC.

In private model, there is no need to use commitment, i.e., two cards to express one bit, for input by utilizing PP. For instance, in Chapter 3, we will express 0 and 1 by do nothing and permute order by PP, which means no additonal card is necessary to express an input value. Actually, we can construct an n-bit input protocol with less than $2n$ cards by utilizing PP. We show that protocols for logic gates, the threshold function, and the millionaires' problem with less than $2n$ cards in this thesis. These protocols do not rely on the combination of logic gates. They are specialized for each function and are more efficient than generic construction. Unfortunately, our protocols for logic gates cannot be used for the combination with other protocols. It is future work to verify whether efficiency can be improved even in general constructions, such as the combination of logic gates.

It is also expected that algebraic MPC's achievements can be returned to card-based cryptography by closing the model gap between card-based cryptography and algebraic MPC. We show that a new card-based protocol can be obtained by converting (algebraic) Yao's millionaires' protocol to card-based cryptography. On the contrary, it is expected that card-based cryptography can help algebraic MPC, which is the future work.

1.5 Our Results

We propose a new assumption, "private permutation (PP)," as mentioned in the previous section. PP enables us to the input representation without using the commitment. As a result, we can construct the protocols with less than the lower bound of cards in traditional card-based cryptography. We propose several protocols for demonstrating PP's power. We summarize our results in Table 1.2 where n is the number of players and m is the bit length of players' inputs.

Chapter 3: We show that an n-input protocol can be constructed with fewer cards than $2n$. In traditional card-based cryptography, it is assumed that all operations are performed in public. This assumption restricts the input representation to use two face-down cards. PP removes this restriction since the input value can be expressed with PP, e.g., defined by permuting cards if the input value is 0. Doing nothing otherwise.

Chapter 3 shows the following three 2-bit input protocols using PP. Note that the study of constructing logic gate protocols are fundamental in traditional card-based cryptography with shuffle.

- 3-card AND (proposed by Marcedone et al. [29])
- 3-card OR (This work)
- 2-card XOR (This work)

First, we show the protocol that is proposed by Marcedone, Wen, and Shi [29]. They succeeded in reducing the number of cards by the same idea to PP, independently to our work.[*7] The 3-card OR protocol is easily derived from their AND protocol by De Morgan's laws (described in Section 3.3.1). Also,

[*7]In [29], PP is realized PP by permuting cards in an empty room.

we show the XOR protocol can be done with only two cards by a similar technique in Section 3.3.2.

Chapter 4: Chapter 3 showed that PP works effectively for improving efficiency in basic protocols for logic gates. We offer that PP also works effectively for more advanced functions in this chapter. We propose the following two protocols.

- 3-input majority voting protocol with four cards
- n-input threshold function protocol with $n+1$ cards

The 3-input majority voting protocol aims to obtain the voting result with three voters who have Boolean inputs while keeping the input values secret. The threshold function is a Boolean function to determine whether $x_1 + x_2 + \cdots + x_n \geq t$ where the threshold value $0 \leq t \leq n$. Note that the threshold function is a generalization of majority voting.

Our idea is the following: The three-card AND and OR protocols described in Chapter 3 can be extended to a 2-input 2-output protocol with four cards that simultaneously obtains AND and OR results by utilizing the *symmetry* between them. Our 3-input majority voting protocol is based on this four-card AND/OR protocol without any additional cards (described in Section 4.2). We show the outline of this protocol below:

We utilize the following relational expression.

$$\text{If } c = 0: \ a + b + c \geq 2 \iff a + b \geq 2 \iff a \wedge b = 1 \tag{1.3}$$
$$\text{If } c = 1: \ a + b + c \geq 2 \iff a + b \geq 1 \iff a \vee b = 1 \tag{1.4}$$

Four-card AND/OR protocol is used to obtain $a \wedge b$ and $a \vee b$. After that, the third player whose input is c chooses which one outputs by using PP, and this action is her input. Namely, the third player uses no card to express her input value.

The lower bound of the number of cards is six for a 3-bit input protocol in public model. We propose a protocol that beats the lower bound.

Our 3-input majority voting protocol can be extended to more participants. Furthermore, the threshold function can be reduced to the majority voting by fixing some input values. From these facts, we can obtain a protocol for (t, n)-threshold function for any t and n (described in Section 4.3). Our protocol is realized with $n+1$ cards. This result succeeds in reducing the number of cards to below the traditional lower bound. Also, we evaluate a card-based protocol by the number of PPs and communications as the computational cost. As described in Section 1.4, a shuffle is interpreted as two PPs and one communication. We convert a protocol in public model to a protocol in private model when comparing the efficiency between them. Then, our threshold function protocol succeeds in reducing the number of PPs and communications from $4n^2$ to n and $2n^2$ to $n-1$, respectively. We note that this protocol is not the general construction, such as the combination of logic gates but specialized construction for the threshold function.

Chapter 5: By introducing PP, the frame of card-based cryptography becomes closer to that of algebraic MPC. To demonstrate this, we show that a new card-based protocol can be obtained by converting Yao's (algebraic) millionaires' protocol into a card-based protocol (described in Section 5.2) The millionaires' protocol is a comparison protocol for two m-bit values. We note that at least $4m$ cards are necessary to construct the proposed protocol in public model since this protocol has $2m$-bit input. Our proposed protocol can be easily derived if we understand the essence of Yao's protocol. Also, our protocol is considerably simplified thanks to the property of cards, which strips away the complexities of the algebraic process. This is the first result of obtaining a card-based protocol by converting an algebraic protocol. Although this protocol is improved from the viewpoint of the number of PPs and communications, the number of cards exponentially increases from the existing protocol, which is constructed by the combination of logic gates.

Thus, we propose efficient millionaires' protocols in Section 5.3.2 by using bitwise representation of input values. First, we propose a millionaires' protocol with $4m + 2$ cards. Thus, it does not succeed in breaking the lower bound of the number of cards in public model, although it succeeds in reducing the number of cards the above protocol. We show the outline of this protocol below:

Let $a = (a_m, \ldots, a_1)$ and $b = (b_m, \ldots, b_1)$ be binary inputs. Then, input values are represented by two card representation as in public model , i.e., $0 \mapsto \boxed{\clubsuit}\boxed{\heartsuit}$ and $1 \mapsto \boxed{\heartsuit}\boxed{\clubsuit}$. Thus, $4m$ cards are used for the input representation. This protocol adopts the bitwise comparison from the least significant bit. Then, we want to extract information of the most significant bit that holds $a_i \neq b_i$ to determine if $a > b$. To achieve this, we prepare two additional cards for the following two roles.

- *output card*: To be overwritten with b_i if $a_i \neq b_i$
- *dummy card*: To be overwritten with b_i if $a_i = b_i$

We note that b_i shows whether $a_i > b_i$ or not if $a_i \neq b_i$. Repeating this recording process up to the most significant bit, the output card has information of the most significant bit that holds $a_i \neq b_i$. Then, the reason why the output and dummy cards are one card respectively is that one-card representation is sufficient, such as $0 \mapsto \boxed{\clubsuit}$ and $1 \mapsto \boxed{\heartsuit}$. Namely, b_i is recorded as one-card representation. To realize this process, a player Alice sends her bit a_i to the other player Bob, and Bob compares and overwrites according to the above rules. Of course, the outline given here does not consider confidentiality. In the actual protocol, Alice sends a_i or $\neg a_i$ to achieve confidentiality. It does not seem that Bob can correctly overwrite in this case. However, our protocol provides both the correct overwriting and confidentiality. The interesting point is that this protocol's main idea is related to the famous logic puzzle, "The fork in the road." [*8]

This protocol does not succeed in breaking the lower bound of the number of cards because input values are expressed as traditional card-based cryptography, i.e., two cards per bit. We show that further improvement is possible for the number of cards by applying the idea of expressing the input values with PP as in Chapters 3 and 4. In the original protocol, overwritten cards, i.e., output/dummy cards, are discarded without opening since these cards have input value information. On the other hand, our improved protocol reuses these discarded cards. Two players randomize the reused cards each other before reusing them to delete the input value information. Then, this protocol adopts two-card representation for output/dummy cards since once-card representation cannot be randomized. The reused cards are utilizes for expressing players' input values. As a result, our improvement enables us to solve the millionaires' problem with only six cards, which is the most significant result to show the power of PP (described in Section 5.3.3).

1.6 Organization of This Thesis

The remaining part of this thesis is organized as follows: We introduce several notations, basic operations of cards, including PP, and the security notion for card-based cryptography in Chapter 2. Chapter 3 presents card-based protocols for logic gates by utilizing PP. In Chapter 4, we first show how to merge the 3-card AND and three-card OR protocols into 4-card AND/OR

[*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? [19, p.25]

protocol. Then, we propose 3-input majority voting protocol and threshold function protocol based on the 4-card AND/OR protocol. Chapter 5 is devoted to the proposal of millionaires' protocol. We first show how to convert algebraic Yao's protocol into a card-based protocol. Then, we propose another millionaires' protocol that is utilizing the idea of the logic puzzle "The fork in the road." We conclude this thesis in Chapter 6.

Table 1.2: Summary of Our Results

Protocol	References	# of PPs	# of Comm.	# of Cards
OR	Mizuki et al. [35]	3	2	4
	Sect. 3.3.1	2	1	3
XOR	Mizuki–Sone [38]	2	1	4
	Sect. 3.3.2	2	1	2
Majority Voting with 3 inputs	Nishida et al. [45]	5	3	8
	Sect. 4.2	3	2	4
Threshold Function	Nishida et al. [44]	$4n^2$	$2n^2$	$2n+2$
	Sect. 4.3	n	$n-1$	$n+1$
Millionaires' Problem	Nishida et al. [44]	$6m-5$	$12m-10$	$4m+2$
	Sect. 5.2	1	2	$2 \cdot 2^m$
	Sect. 5.3.2	$2m$	$2m+1$	$4m+2$
	Sect. 5.3.3	$2m$	$2m+1$	6

Chapter 2

Preliminaries

2.1 Notations and Basic Operations in Card-based Cryptography

In card-based cryptography, we normally use two types of cards such as $\boxed{\clubsuit}$ and $\boxed{\heartsuit}$.[*1] 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 $\boxed{?}$. While some studies uses information on the top and bottom of marks such as $\clubsuit$ and $\rotatebox{180}{$\clubsuit$}$ [37], we do not apply this method and assume that there is only one card orientation. The Boolean values 0 and 1 are encoded as $\boxed{\clubsuit}\boxed{\heartsuit}$ and $\boxed{\heartsuit}\boxed{\clubsuit}$, respectively. Note that we regard a card order as a vector. In this thesis, we use the following fundamental card operations [36]. Note that these operations are executed *publicly*.

- Face up: $\boxed{?} \mapsto \boxed{\clubsuit}$, $\boxed{?} \mapsto \boxed{\heartsuit}$

[*1] There are also researches that uses different cards such as ordinary playing cards [31, 43], polygonal cards [54] and polarizing plates [55].

- Face down: $\boxed{♣} \mapsto \boxed{?}$, $\boxed{♡} \mapsto \boxed{?}$

- Public permutation: e.g., $\boxed{♣}\boxed{♣}\boxed{♡} \mapsto \boxed{♡}\boxed{♣}\boxed{♣}$

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*.

For simplicity, $\boxed{♣}$ and $\boxed{♡}$ are represented as ♣ and ♡, respectively.

2.2 Shuffles and Private Permutation

2.2.1 Shuffles Used in Previous Works

den Boer utilized the shuffle, called *random cut*, to realize five-card AND protocol in non-committed format [15]. The random cut is one of the shuffles that repeats the procedure of moving the first card to the end until all players cannot specify the result. Crepeau–Kilian [12], Niemi–Renvall [43], and Stiglic [57] also used this shuffle to construct card-based AND protocol, in committed format with ten-card, twelve-card, and eight-card respectively. After that, Mizuki and Sone succeeded in reducsing the number of cards by introducing the new shuffle *random bisection cut* [38].

Random Bisection Cut. This is a key technique to realize efficient card-based protocols for logic gates, e.g., four-card AND protocol in non-committed format [35] and six-card AND protocol in committed format [38], 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 $\vec{u}_1$ and $\vec{u}_2$, respectively. Namely, we define

$$\underbrace{\overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{v\text{ cards}}}_{=:\vec{u}_1}\underbrace{\overbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}^{v\text{ cards}}}_{=:\vec{u}_2}. \tag{2.1}$$

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

$$\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\vec{u}_1}\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\vec{u}_2} \quad \text{or} \quad \underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\vec{u}_2}\underbrace{\boxed{?}\boxed{?}\cdots\boxed{?}}_{\vec{u}_1} \tag{2.2}$$

is selected with a probability 1/2. This operation, called *random bisection cut*, is executed in public, but it is assumed that *no player knows* whether one of the above is selected.

Although the random bisection cut contributes to the efficiency of card-based protocols, it is arguable whether this shuffle, which has only two results, can realize uniformly at random with human hands in public [44, 59].

Non-uniform Shuffle. While it is assumed that the results are chosen uniformly at random in the random cut and random bisection cut, there are also shuffles that are assumed to be the results are chosen with non-uniform probability. Koch, Walzer, and Härtel utilized such *non-uniform shuffles* to reduce the number of cards further [27]. However, it is controversial whether non-uniform shuffles can be implemented with only human hands [46].

2.2.2 Proposal of Private Permutations

The shuffle is regarded as a convenient randomization technique for implementing card-based cryptography. However, the assumption of shuffle is *card-oriented*, as is pointed out in Section 1.4. The shuffle is a key technique that realizes card-oriented protocols, but it also causes a gap with algebraic

MPC. Concretely, algebraic MPC cannot accept the following assumptions:

- All players cannot identify the random number even if it is generated in public area.
- Every player cannot know the random number which is generated by him/herself.

Both assumptions are natural for real shuffles, e.g., in playing cards. On the other hand, the shuffle used in card-based cryptography does not completely randomize cards, unlike used in general card games. Thus the shuffle used in card-based cryptography has a problem in them feasibility even if it is accepted that they are based on card-oriented properties.

In order to solve these problems, we introduce new card operations *private permutation* (PP) and *communication* instead of the shuffle. PP allows us to construct card-based protocols with the private randomness like algebraic MPC, e.g., by permuting cards behind the player's back. It becomes necessary to clarify who perform the operation as a result of introducing the private randomness in card-based cryptography. Thus we introduce an operation to hand cards to other players, i.e., communication, in order to make it clear who owns the cards. As a result, shuffles are not interpreted as one operation but as being realized by multiple operations in our model.

Concretely, 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.

We fomalize PP as follows: For a positive integer t, let $\vec{c} \in \{\clubsuit, \heartsuit\}^t$ be a vector consisting of t face-down cards. For a set $\mathcal{P}_t$ of all permutations over $[t] := \{1, 2, \ldots, t\}$, let $\mathcal{R}_t \subset \mathcal{P}_t$ be a set of possible permutations. We also define $\mathcal{R}_t = \{\pi_0, \pi_1, \ldots, \pi_{|\mathcal{R}_t|-1}\}$, 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:

$$\mathsf{PP}^{[t]}_{\mathcal{R}_t}(\vec{c}, s) := \pi_s(\vec{c}), \quad s = 0, 1, \ldots, |\mathcal{R}_t| - 1. \tag{2.3}$$

Note that the same function was introduced in the previous works [27,36] 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. We note that, not only the random bisection cut, but also several different types of *shuffles*, e.g., in [43], 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 $\mathcal{R}^{\mathsf{bc}}_{2v} := \{\pi_0, \pi_1\} \subset \mathcal{P}_{2v}$ where

$$\pi_0 := (1, \ldots, v, v+1, \ldots, 2v), \text{ and } \pi_1 := (v+1, \ldots, 2v, 1, \ldots, v). \tag{2.4}$$

Eq. 2.4 means that $\pi_0(\vec{c}) = (\vec{u}_1, \vec{u}_2)$ and $\pi_1(\vec{c}) = (\vec{u}_2, \vec{u}_1)$ for $\vec{c} := (\vec{u}_1, \vec{u}_2)$ given by (2.1). Then, the random bisection cut for $2v$ cards is represented as $\mathsf{PP}^{[2v]}_{\mathcal{R}^{\mathsf{bc}}_{2v}}(\vec{c}, s) = \pi_s(\vec{c})$ 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 a card order $\vec{c}$, Alice executes $\mathsf{PP}^{[2v]}_{\mathcal{R}^{\mathsf{bc}}_{2v}}(\vec{c}, r_A) =: \vec{c'}$ by using her private randomness $r_A \in \{0, 1\}$, and $\vec{c'}$ is sent to Bob. Bob also executes $\mathsf{PP}^{[2v]}_{\mathcal{R}^{\mathsf{bc}}_{2v}}(\vec{c'}, r_B)$ by using his private randomness $r_B \in \{0, 1\}$.

In most of previous works, all operations are performed *in public* so as to avoid cheating. On the other hand, our card-based protocols are realized under the semi-honest model since we adopt the operation of executing in player's private area.[*2]

Efficiency Measures. Most of the previous work, e.g., [36, 54], considered the number of shuffles as the *computational complexity* since shuffle is the most time-consuming operation. On the other hand, in this thesis we consider that the computational complexity is evaluated by the number of PPs and communications since such measures are suitable for algebraic MPC. In this thesis, 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 Example: Six-card AND Protocol

In order to clarify the difference between shuffles and PPs, we show two kinds of implementations of six-card AND protocol [38], 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 and who perform the operations.

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 six-card AND protocol [38] realizing this requirement.

[*2]It is known that malicious behaviors can be detected by having players act as monitors in a particular protocol [2].

2.4 Security Notion

Throughout this thesis, we assume that both Alice and Bob are semi-honest players. Following [11], we introduce the security notion (perfect secrecy) of card-based cryptography for the millionaires' problem.

In defining the security of card-based cryptography, *view* plays a key role. View is roughly defined to be *a vector of random variables*[*3] *to the data that each player can obtain in the protocol.* Specifically, view includes the randomvariables 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 $i \in [m]$, let x_i be n-bit integers representing the input values of player P_i. The common output of the protocol for all players is represented as $\chi(x_1, x_2, \ldots, x_m)$. The information obtained by each player in the protocol can be classified into *private information* denoted by r_i for each $i \in [m]$, and *public information* denoted by λ.

Hence, view of P_i can be described as the sequence of random variables corresponding to her/his input value x_i, output of the protocol, private information r_i and public information λ. The private information r_i is the random number generated by P_i for PPs. The public information is the cards that the players made public by turning them face-up. Note that, in algebraic 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 X_i, R_i, and Λ be random variables corresponding to the values x_i, r_i, and λ, respectively. Then, the views of P_i

[*3]Throughout the thesis, 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, 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.

are represented as $(X_i, \chi(X_1, X_2, \ldots, X_n), R_A, \Lambda)$.

Intuitively, if private and public information of P_i can be simulated from her/his input and output for all $i \in [m]$, we can say that no information is contained in the private and public information other than her/his input and output. Hence, we can formulate perfect secrecy of card-based cryptography for the millionaires' problem as follows:

Definition 1 (Perfect secrecy) *Consider a card-based protocol for players $P_1, P_2, \ldots, P_m$. We say that the card-based protocol is perfectly secure if for all $i \in [m]$, there exist simulators S_i such that for all possible input values of the protocol, it holds that*

$$\mathsf{S}_i(x_i, \chi(x_1, x_2, \ldots, x_m)) \overset{\text{perf}}{\equiv} (a, \chi(x_1, x_2, \ldots, x_m), R_A, \Lambda) \tag{2.7}$$

where $U \overset{\text{perf}}{\equiv} V$ means that the each probability distribution Q_U, Q_V corresponding to the random variables U and V, respectively, are perfectly the same.

Protocol 1 Six-card AND Protocol [38] (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 card order 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 card order obtained in step 3). Note that the result of 2)–4) is either $(a, 0, b)$ or $(\neg a, b, 0)$ 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

$$\boxed{\clubsuit}\boxed{\heartsuit}\ \overbrace{\boxed{?}\boxed{?}}^{\text{output}}\ \boxed{?}\boxed{?} \text{ or } \boxed{\heartsuit}\boxed{\clubsuit}\boxed{?}\boxed{?}\ \overbrace{\boxed{?}\boxed{?}}^{\text{output}}.$$

Protocol 2 Six-card AND Protocol [38] (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 $\mathcal{R}_6^{\mathsf{bc}}$ which is defined in (2.4) with $v = 3$.

$$c_A := \mathsf{PP}^{[6]}_{\mathcal{R}_6^{\mathsf{bc}}}(c', r_A) \tag{2.5}$$

where $\mathcal{R}_6^{\mathsf{bc}} := \{\pi_0, \pi_1\}$ is given by (2.4) 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.

$$c_B := \mathsf{PP}^{[6]}_{\mathcal{R}_6^{\mathsf{bc}}}(c_A, r_B) \tag{2.6}$$

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.

Chapter 3

Securely Computing Logic Gates

3.1 Introduction

In traditional card-based cryptography, it is necessary for an n-bit input protocol to use at least $2n$ cards since two cards are required for the arbitrary expression of a 1-bit value. This restriction is due to the assumption that all operations are performed in public, and as a result, face-down cards are the only way to secure input representation. On the other hand, we remove this assumption by introducing PP. PP allows us to represent an input value in another way, e.g., privately permute cards only when the input value is 1 and makes it possible to construct a protocol with fewer cards than the lower bounds of traditional card-based cryptography.

In this chapter, we demonstrate that 2-bit input protocols can be constructed with less than four cards. Proposed protocols are for logic gates, which are the main focus of study in card-based cryptography. First we describe the three-card AND protocol proposed by Marcedone, Wen, and Shi [29]. They introduced the private operation to card-based cryptography

Table 3.1: Summary of Our Results in Chap. 3

Protocol	References	# of PPs	# of comm.	# of Cards
AND	Mizuki et al. [35]	3	2	4
	Marcedone et al. [29]	2	1	3
OR	Mizuki et al. [35]	3	2	4
	This work [Sect. 3.3]	2	1	3
XOR	Mizuki–Sone [38]	2	1	4
	This work [Sect. 3.3]	2	1	2

independently and succeeded in reducing the number of cards. We construct three-card OR protocol by applying De Morgan's law to their protocol. As a result, our protocol becomes symmetrical to three-card AND protocol. Also, we show that XOR protocol can easily be obtained with only two cards.

We summarize our result in Table 3.1. In this chapter, let a and b be the binary inputs of Alice and Bob, respectively.

Organization: In Section 3.2 we introduce the three-card AND protocol proposed by Marcedone, Wen, and Shi [29]. Based on their protocol, we propose three-card AND and two-card OR protocols in Section 3.3. Section 3.4 is the summary of this chapter.

3.2 Our Idea: Three-card AND Protocol

In the Epilogue in [29] (Solution B), the three-card AND protocol is proposed as shown in Protocol 3.[*1] We note that step 2) is realized with following PP:

$$\mathsf{PP}^{[2]}_{\mathcal{R}_2^{\mathsf{bc}}}(\vec{c}, b) := \vec{c'} \tag{3.1}$$

where $\mathcal{R}_2^{\mathsf{bc}} := \{\pi_0, \pi_1\}$, $\pi_0 := (1, 2)$ and $\pi_1 := (2, 1)$.

[*1]Slightly modified for later discussion, but essentially the same as the protocol in [29].

Protocol 3 Three-card AND Protocol [29]

Inputs: Alice has $a \in \{0,1\}$, and Bob has $b \in \{0,1\}$.
Setup: Alice has $\clubsuit\heartsuit$. Bob has $\clubsuit$.

1) Alice performs the following operation.
   - If $a = 0$, sends face-down $\clubsuit$ to Bob.
   - If $a = 1$, sends face-down $\heartsuit$ to Bob.
2) Bob performs the following operation with PP.
   - If $b = 0$, places face-down $\clubsuit$ to the left side of the receiced card.
   - If $a = 1$, places face-down $\clubsuit$ to the right side of the receiced card.
3) Open the right card in public area.
   - If this card is $\clubsuit$, then $a \wedge b = 0$.
   - If this card is $\heartsuit$, then $a \wedge b = 1$.

Table 3.2 shows the correspondence between the card order at the end of step 2) and the output of the protocol. Subscripts of $\clubsuit$ and $\heartsuit$ indicate the player who had the card originally.[*2]

We also note that Bob's input at step 3) in Protocol 3 is not represented by the suit of the card but is represented by the action taken by Bob, i.e., Bob's value corresponds to his choice of left or right where he places his $\clubsuit$.

Table 3.2: Three-card AND protocol

a	b	Step 2)	Output
0	0	$\clubsuit_{\text{Bob}}\ \clubsuit_{\text{Alice}}$	0 ($\clubsuit_{\text{Bob}}$)
0	1	$\clubsuit_{\text{Alice}}\ \clubsuit_{\text{Bob}}$	0 ($\clubsuit_{\text{Alice}}$)
1	0	$\clubsuit_{\text{Bob}}\ \heartsuit_{\text{Alice}}$	0 ($\clubsuit_{\text{Bob}}$)
1	1	$\heartsuit_{\text{Alice}}\ \clubsuit_{\text{Bob}}$	1 ($\heartsuit_{\text{Alice}}$)

[*2]Hereafter, we remove the frame of cards for simplicity.

In this study, we utilize this idea to express a player's input by his/her action and succeed in reducing the number of cards compared to previous work.

Security Proof of three-card AND protocol: We present a brief overview of the security proof for Protocol 3, which will be useful to understand the security of the protocols proposed hereafter.

Since we compute AND, the player who inputs 1 can uniquely determine the other player's input at the end of the protocol. Meanwhile, for the player who inputs 0, no information must leak out to the player, which we have to check. When Alice inputs $a = 0$ ($\clubsuit$), the output is either $\clubsuit_{\text{Alice}}$ or $\clubsuit_{\text{Bob}}$, which is opened by Bob and is indistinguishable from Alice. When Bob inputs $b = 0$, he places his $\clubsuit$ on the left, and he simply shows his card to Alice. Hence, he obtains no information on Alice's input, which is discarded at the end of the protocol.

It is clear that no information is obtained by the players other than Alice and Bob (if such players exist) because the only information they can get is the output. □

3.3 Logic Gate Protocols

In this section, we show the two types of card-based protocol for OR and XOR operations. We first show how to obtain three-card OR protocol based on the three-card AND protocol by De Morgan's law. After that, we propose two-card XOR protocol.

3.3.1 Three-card OR Protocol

Although the concept of PPs is implicitly used in [29], this paper only concentrated on the construction of card-based AND protocols, and no card-based protocols were shown for the other logic gates. Hereafter, we show card-based

Protocol 4 Three-card OR Protocol

Inputs: Alice has $a \in \{0,1\}$, and Bob has $b \in \{0,1\}$.
Setup: Alice has $\clubsuit\heartsuit$. Bob has $\clubsuit$.

1) Alice performs the following operation.
   - If $a = 0$, sends face-down $\underline{\heartsuit}$ to Bob.
   - If $a = 1$, sends face-down $\underline{\clubsuit}$ to Bob.
2) Bob performs the following operation with PP.
   - If $b = 0$, places face-down $\clubsuit$ to the $\underline{\text{right}}$ side of the receiced card.
   - If $a = 1$, places face-down $\clubsuit$ to the $\underline{\text{left}}$ side of the receiced card.
3) Open the left card in public area.
   - If this card is $\underline{\heartsuit}$, then $a \vee b = 0$.
   - If this card is $\underline{\clubsuit}$, then $a \vee b = 1$.

protocols for computing OR and XOR, which are realized with three and two cards, respectively.

To construct card-based OR protocols, we should recall De Morgan's law: $a \vee b = \neg(\neg a \wedge \neg b)$. The card-based OR protocol can be obtained from this identity by negating Alice's input, Bob's input, and the output. Specifically, when Alice inputs $a = 0$, she should use $\heartsuit$ (otherwise $\clubsuit$), and when Bob inputs $b = 0$, he should place $\clubsuit$ to the *right* of the card he received. Finally,

Table 3.3: Three-card OR Protocol

a	b	Step 2)	Output
0	0	$\heartsuit_{\text{Alice}}\ \clubsuit_{\text{Bob}}$	$0\ (\heartsuit_{\text{Alice}})$
0	1	$\clubsuit_{\text{Bob}}\ \heartsuit_{\text{Alice}}$	$1\ (\clubsuit_{\text{Bob}})$
1	0	$\clubsuit_{\text{Alice}}\ \heartsuit_{\text{Bob}}$	$1\ (\clubsuit_{\text{Alice}})$
1	1	$\clubsuit_{\text{Bob}}\ \clubsuit_{\text{Alice}}$	$1\ (\clubsuit_{\text{Bob}})$

Protocol 5 Two-card XOR Protocol

Inputs: Alice has $a \in \{0, 1\}$ and Bob has $b \in \{0, 1\}$.
Initial Setting: Alice has $\clubsuit\heartsuit$. Bob has no card.

1) Alice performs the following operation.
   - If $a = 0$, sends face-down $\clubsuit\heartsuit$ to Bob.
   - If $a = 1$, sends face-down $\heartsuit\clubsuit$ to Bob.
2) Bob performs the following operation with PP.
   - If $b = 0$, do nothing.
   - If $b = 1$, swaps the received cards.
3) Open the two cards in public area.
   - If these cards are $\clubsuit\heartsuit$, then $a \oplus b = 0$.
   - If these cards are $\heartsuit\clubsuit$, then $a \oplus b = 1$.

the output should be negated. Then, we have Protocol 4, where the different parts from Protocol 3 are underlined.

The relation among the inputs, the card order at the end of step 2), and the output is shown in Table 3.3. Security proof is not necessary since this protocol is essentially the same as Protocol 3.

3.3.2 Two-card XOR Protocol

The proposed two-card XOR protocol is shown in Protocol 5. In this protocol, PPs are used in steps 1) and 2). The relationships among the inputs, the pair of cards at the end of step 2), and the output is shown in Table 3.4.

Security of Two-card XOR Protocol: For Alice and Bob, there is no information to be kept secret because, if the value of XOR and one of the two inputs are given, the other input is uniquely determined. Furthermore, no

Table 3.4: Two-card XOR Protocol

a	b	Step 2)	Output
0	0	♣ ♡	0 (♣♡)
0	1	♣ ♡	1 (♡♣)
1	0	♡ ♣	1 (♡♣)
1	1	♡ ♣	0 (♣♡)

information except for the output is known to the players other than Alice and Bob.

It is clear that no information is obtained by the players other than Alice and Bob (if such players exist) because the only information they can get is the output. □

3.4 Results and Discussion

In traditional card-based cryptography, it is necessary to use at least $2n$ cards to realize an n-bit input protocol. This lower bound is due to the constraint that input values must be represented by cards. However, PP removes this restriction and enables us to construct a protocol with less than the lower bound. In Chapter 3, we showed the following 2-bit input protocols could be constructed with less than four cards.

- Section 3.2: three-card AND protocol proposed by Marcedone et al. [29]
- Section 3.3.1: three-card OR protocol
- Section 3.3.2: two-card XOR protocol

In this chapter, we focus on logic gates, which are the mainstream in previous works. We succeeded in reducing the number of cards in these basic protocols. The remaining chapter shows that more advanced protocols, such as the threshold function and millionaires' protocol, can also be improved utilizing PP.

Chapter 4

Securely Computing Threshold Function

4.1 Introduction

Chapter 3 showed that 2-bit input logic gates could be securely computed with less than four cards. This result offers that PP is useful for reducing the number of cards, using the basic protocols as an example. On the other hand, the number of PPs and communications is not reduced since the logic gates are basic functions, and thus there is no room for the reduction. In this chapter, we propose a protocol for more advanced functions such as the threshold function, which is efficient not only for the number of cards but also for the number of PPs and communications.

Our Idea: The interesting points of card-based AND and OR protocols, described in Chapter 3, are not only that we can substantially reduce the number of cards, but we can also simultaneously realize AND and OR operations. This simultaneous realization enables us to implement the 3-input majority voting protocol with only *four* cards.

Our main idea for 3-input majority voting is to utilize the simultaneous realization of AND and OR operations. Observing the relations for $a, b \in \{0, 1\}$,

$$\begin{aligned} a \wedge b = 1 &\iff a + b \geq 2 \\ a \vee b = 1 &\iff a + b \geq 1 \end{aligned}$$

it seems that $a \wedge b$ and $a \vee b$ can be interpreted as the interim result of the majority voting, respectively. Here, we consider the strategy that $a \wedge b$ and $a \vee b$ is passed to the third player, who holds $c \in \{0, 1\}$, for computing the result of three-inputs majority voting. Then, we can understand that it is different whether the desired value for the third player is $a \wedge b$ or $a \vee b$ depending on c from the following trivial relations,

$$\begin{aligned} \text{if } c = 0 \text{ then, } a + b + c \geq 2 &\iff a + b \geq 2, \\ \text{if } c = 1 \text{ then, } a + b + c \geq 2 &\iff a + b \geq 1. \end{aligned}$$

Therefore, the third party should choose one of them depending on her input c to obtain the result of the majority voting. Then, we note that the third player needs not to use any card since she plays only the role of selecting $a \wedge b$ or $a \vee b$, which are created by the other two players. In other words, we can obtain a protocol for three-input majority voting without adding any cards from simultaneous AND and OR protocol, i.e., we can construct it using only *four* cards.

Our protocol for 3-input majority voting can be extended for more voters. Utilizing this fact, we construct a threshold function protocol by the reduction to the majority votings. More formally, we utilize the following property:

Let $\Pi_{t,n}$ be a protocol for threshold function where t is a threshold value and n is the number of participants. Then $\Pi_{t,n} = \Pi_{t,n-1}$ holds if one of the input values is fixed to 1.

Table 4.1: Summary of Our Results in Chap. 4

Protocol	References	# of PPs	# of comm.	# of Cards
Majority Voting	Nishida et al. [45]	5	3	8
with 3 inputs	This work [Sect. 4.2]	3	2	4
Threshold	Nishida et al. [44]	$4n^2$	$2n^2$	$2n+2$
Function	This work [Sect. 4.3]	n	$n-1$	$n+1$

This reduction is realized by setting up *dummy participants* whose inputs are fixed to 1. As a result, we obtain threshold function protocol with $n+1$ cards, which is fewer than the lower bound of traditional card-based cryptography.

Organization: In Section 4.2, we first show how to obtain AND and OR results simultaneously with four cards. We propose 3-input majority voting protocol based on this protocol without additional cards. Also, we show that this protocol can be extended to a protocol for the threshold function in Section 4.3. Section 4.4 is the summary of this chapter.

4.2 Three-input Majority Voting Protocol with Four Cards

Based on the observations on the three-card AND/OR protocols, we propose a three-input majority voting protocol that uses only four cards. Consider the scenario such that Alice, Bob, and Carol have their binary values a, b, and c, respectively. They want to know the result of majority voting without revealing their individual inputs.

Two types of realizations of such a majority voting protocol can be considered. One realization is computing the summation $s := a+b+c$ and then output s, which tells us which is the majority [32]. The other realization is to output 0 if the majority is 0, otherwise output 1 [58]. In this study, we focus on the latter since it is more secure and theoretically interesting. Specifically, we want to compute the following function $\mathsf{maj}(a, b, c) \in \{0, 1\}$ securely:

$$\mathsf{maj}_3(a, b, c) = \begin{cases} 0, & \text{if} \quad a + b + c \leq 1 \\ 1, & \text{if} \quad a + b + c \geq 2. \end{cases} \tag{4.1}$$

4.2.1 Idea behind Our Three-input Majority Voting Protocol

Assume that Alice, Bob, and Carol vote a, b, and c, respectively, in this order. We focus on the Carol's vote $c \in \{0, 1\}$.

In the case of $c = 0$, the following relationship holds.

$$a + b + c \geq 2 \iff a + b \geq 2 \iff a \wedge b = 1 \tag{4.2}$$

This relationship implies that $a \wedge b$ is the result of the majority voting when $c = 0$.

Meanwhile, in the case of $c = 1$, we have the following relationship:

$$a + b + c \geq 2 \iff a + b \geq 1 \iff a \vee b = 1 \tag{4.3}$$

Hence, $a \vee b$ is the result of the majority voting when $c = 1$.

Summarizing, we have

$$\mathsf{maj}_3(a, b, c) = \begin{cases} a \wedge b, & \text{if} \quad c = 0 \\ a \vee b, & \text{if} \quad c = 1, \end{cases} \tag{4.4}$$

which can be calculated securely if we can merge the AND and OR protocols in Protocols 3 and 4, respectively. In fact, such unification is possible by

Protocol 6 Modified Three-card OR Protocol

Inputs: Alice has $a \in \{0, 1\}$ and Bob has $b \in \{0, 1\}$.
Setup: Alice has $\clubsuit\heartsuit$ and Bob has $\heartsuit$.

1) Alice performs the following operation.
   - If $a = 0$, sends face-down $\clubsuit$ to Bob.
   - If $a = 1$, sends face-down $\heartsuit$ to Bob.
2) Bob performs the following operation with PP.
   - If $b = 0$, places face-down $\heartsuit$ to the left side of the receiced card.
   - If $b = 1$, places face-down $\heartsuit$ to the right side of the receiced card.
3) Open the right card in public area.
   - If this card is $\clubsuit$, then $a \vee b = 0$.
   - If this card is $\heartsuit$, then $a \vee b = 1$.

using four cards, which will be explained in the next subsection.

4.2.2 Unifying AND and OR Operations

The unrevealed card in step 3) is not utilized in this step for the three-card AND and OR protocols in Protocols 3 and 4. Our main idea to simultaneously obtain $a \wedge b$ and $a \vee b$ is that the wasted card is also effectively used for representing output value.

Table 4.2: Modified Three-card OR Protocol

a	b	Step 2)	Output
0	0	$\heartsuit_{\text{Bob}}\ \clubsuit_{\text{Alice}}$	0 ($\clubsuit_{\text{Alice}}$)
0	1	$\clubsuit_{\text{Alice}}\ \heartsuit_{\text{Bob}}$	1 ($\heartsuit_{\text{Bob}}$)
1	0	$\heartsuit_{\text{Bob}}\ \heartsuit_{\text{Alice}}$	1 ($\heartsuit_{\text{Alice}}$)
1	1	$\heartsuit_{\text{Alice}}\ \heartsuit_{\text{Bob}}$	1 ($\heartsuit_{\text{Bob}}$)

These protocols are essentially the same based on De Morgan's law. Hence, they are symmetric form, and so, the unrevealed card is on the same side, i.e., the right side. We first modify our three-card OR protocol to resolve this match since it will hinder the simultaneous realization of them.

Modification of Three-card OR Protocol.

To obtain the unified protocol, we should reverse the left and right of the card Bob chose for output in step 3) of Protocol 4. Then, we note that the format of output values for Protocols 3 and 4 must be the same, i.e., we should also resolve the problem which the correspondence between the suit and output value is reversed in this protocol as seen in Tables 3.2 and 3.3.

We exchange ♣ and ♡ in Protocol 4 based on the above discussion. Moreover, we swap the left and right side Bob selected in step 2) of Protocol 4 in order to make $a \vee b$ place on the right side, i.e., the opposite side of Protocol 3. Then, we obtain Protocol 6 from Protocol 4. The relationships among the input and the output are shown in Table 4.2.

Four-card AND/OR Protocol.

Observe that the right card and the left card are discarded at the end of the protocol in both Protocols 3 and 6, respectively. We also observe that Bob has ♣ and ♡ at step 1) in both Protocols 3 and 6, respectively. From these observations, we can merge Protocols 3 and 6 by letting Bob have both of them, i.e., ♣ and ♡, in the initial setup. Then, we can implement the results of AND and OR simultaneously with four cards, as shown in Protocol 7.

We show in the next section that this four-card AND/OR protocol is useful in calculating the three-inputs majority voting with only *four* cards.

Protocol 7 Four-card AND/OR protocol

Inputs: Alice has $a \in \{0,1\}$ and Bob has $b \in \{0,1\}$.
Setup: Each of Alice and Bob has $\clubsuit\heartsuit$.

1) Alice performs the following operation.
   - If $a = 0$, sends face-down $\clubsuit$ to Bob.
   - If $a = 1$, sends face-down $\heartsuit$ to Bob.
2) Bob performs the following operation with PP.
   - If $b = 0$, places face-down $\clubsuit$ on the left side of the received card.
   - If $b = 1$, places face-down $\heartsuit$ on the right side of the received card.
3) Let $0 \mapsto \clubsuit$ and $1 \mapsto \heartsuit$. Then, the left card expresses $a \wedge b$ and the right card expresses $a \vee b$.

4.2.3 Three-input Majority Voting Protocol with Four Cards

Based on the four-card AND/OR protocol, it is easy to compute the majority voting protocol. First, Alice and Bob compute $a \wedge b$ and $a \vee b$ simultaneously, where the result is concealed. Then, Carol chooses $a \wedge b$ or $a \vee b$ depending on $c = 0$ or $c = 1$, respectively, behind her back. The detailed algorithm is shown in Protocol 8. Formally, step 4) is performed with the same PP as (3.1) in Chapter 3. Then, the left card is the picked out card.

Table 4.3 shows the pair of cards at the end of step 2) and the output. Note that the third player, Carol, has no card for her input since her role is to choose $a \wedge b$ or $a \vee b$ by PP. Thus, our protocol for the three-inputs majority voting does not require any additional card from four-card AND/OR protocol.

Protocol 8 Three-input Majority Voting Protocol

Inputs: Alice has $a \in \{0, 1\}$, Bob has $b \in \{0, 1\}$, and Carol has $c \in \{0, 1\}$.
Setup: Alice and Bob each have a pair $\clubsuit\heartsuit$. Carol has no card.

1) Alice performs the following operation.
   - If $a = 0$, sends face-down $\clubsuit$ to Bob.
   - If $a = 1$, sends face-down $\heartsuit$ to Bob.
2) Bob performs the following operation with PP.
   - If $b = 0$, places face-down $\clubsuit$ on the left side of the received card.
   - If $b = 1$, places face-down $\heartsuit$ on the right side of the received card.
3) Bob sends the two cards to Carol.
4) Carol performs the following operation with PP.
   - If $c = 0$, picks out the left card of the received card.
   - If $c = 1$, picks out the right card of the received card.
5) Open the picked out card in public area.
   - If this card is $\clubsuit$, then the output value is 0.
   - If this card is $\heartsuit$, then the output value is 1.

4.3 Card-based Threshold Function Protocol

In this section, we show a new protocol for the threshold function by generalizing Protocol 8. Let $x_1, x_2, \ldots, x_n$ be Boolean inputs of n players $P_1, \ldots, P_n$ respectively. Then, our (t, n)-threshold function protocol aims to compute the following function without revealing input values.

$$f_{(t,n)}(x_1, \ldots, x_n) = \begin{cases} 0, & \text{if } \sum_{i=1}^{n} x_i < t \\ 1, & \text{otherwise} \end{cases} \tag{4.5}$$

Table 4.3: Three-input Majority Voting

a	b	c	Step 2)	Output
0	0	0	$\clubsuit_{\text{Bob}}$ $\clubsuit_{\text{Alice}}$	0 ($\clubsuit_{\text{Bob}}$)
0	1	0	$\clubsuit_{\text{Alice}}$ $\heartsuit_{\text{Bob}}$	0 ($\clubsuit_{\text{Alice}}$)
1	0	0	$\clubsuit_{\text{Bob}}$ $\heartsuit_{\text{Alice}}$	0 ($\clubsuit_{\text{Bob}}$)
1	1	0	$\heartsuit_{\text{Alice}}$ $\heartsuit_{\text{Bob}}$	1 ($\heartsuit_{\text{Alice}}$)
0	0	1	$\clubsuit_{\text{Bob}}$ $\clubsuit_{\text{Alice}}$	0 ($\clubsuit_{\text{Alice}}$)
0	1	1	$\clubsuit_{\text{Alice}}$ $\heartsuit_{\text{Bob}}$	1 ($\heartsuit_{\text{Bob}}$)
1	0	1	$\clubsuit_{\text{Bob}}$ $\heartsuit_{\text{Alice}}$	1 ($\heartsuit_{\text{Alice}}$)
1	1	1	$\heartsuit_{\text{Alice}}$ $\heartsuit_{\text{Bob}}$	1 ($\heartsuit_{\text{Bob}}$)

4.3.1 Extending to n-input Majority Voting Protocol

We first show that our three-input majority voting protocol, shown in Section 4.2.3, can be extended to n-input majority voting protocol. Here, we define the function for n-input majority voting as follows. [*1]

$$\mathsf{maj}_n(x_1, \ldots, x_n) = \begin{cases} 0, & \text{if } \sum_{i=1}^{n} x_i < n/2 \\ 1, & \text{otherwise} \end{cases} \tag{4.6}$$

The first step for generalizing the number of inputs is to focus on the three players, Alice, Bob, and Carol, in our three-input majority voting protocol. We will discuss the roles of the first half players (Alice, Bob) who use cards for input and the second half (Carol) who does not use cards for input. Here, Table 4.4 summarizes the relationship between input and output focus on the input values' sum.

In Section 4.2, Carol's operation was explained as an operation of "selecting an output card," but in this section, it is interpreted as the operation of "removing unnecessary cards" for generalization. In other words, we interpret Carol's role as removing the right card if $c = 0$ and removing the left

[*1]Note that 1 is output if n is even and the number of inputs of 0 and 1 are the same.

Table 4.4: Mechanism of Three-input Majority Voting Protocol

$a+b$	Received Cards of Carol	Output ($c=0$)	Output ($c=1$)
0	♣♣	♣	♣
1	♣♡	♣	♡
2	♡♡	♡	♡

Table 4.5: Mechanism of n-input Majority Voting Protocol (Case of $n = 5$)

$x_1 + x_2 + x_3$	Received Cards of P_4	Output ($x_4 + x_5 = 0$)	Output ($x_4 + x_5 = 1$)	Output ($x_4 + x_5 = 2$)
0	♣♣♣	♣	♣	♣
1	♣♣♡	♣	♣	♡
2	♣♡♡	♣	♡	♡
3	♡♡♡	♡	♡	♡

card if $c = 1$. In the framework of n-input majority voting, the discussion is based on the policy that the first half players P_i ($1 \leq i \leq \lceil n/2 \rceil$) is "the players who have the input cards and adds one card" and the second half players P_j ($\lceil n/2 \rceil < j \leq n$) is "the players who have no input card and removes one card." As a result, the last remaining card is the output.

Case Where n Is An Odd Number

We suppose that the first half of players P_i ($i = 1, \ldots, n$) perform the same operations as Alice and Bob in our three-input majority voting protocol. Namely, for $1 \leq i \leq \lceil n/2 \rceil$, suppose that P_i performs the following operations with PP sequentially.

- If $x_i = 0$, then P_i places face-down ♣ on the leftmost of the received cards, and sends the cards after processing to P_{i+1}.
- If $x_i = 1$, then P_i places face-down ♡ on the rightmost of the received

cards, and sends the cards after processing to P_{i+1}.

However, P_1 only sends $\clubsuit$ or $\heartsuit$ to P_2, like Alice, since there is no card sent from the previous player.

We do not interpret Carol's operation as "selecting an output card" but "removing an unnecessary card," and suppose the second half players $P_i\,(i = 1, \ldots, n)$ perform the same operation as her. Namely, for $P_j\ (\lceil n/2 \rceil < j \leq n)$, suppose that P_j performs the following operations with PP sequentially.

- If $x_j = 0$, then P_j removes the rightmost card of the received cards, and sends the cards to P_{j+1}.
- If $x_j = 1$, then P_j removes the leftmost card of the received cards, and sends the cards to P_{j+1}.

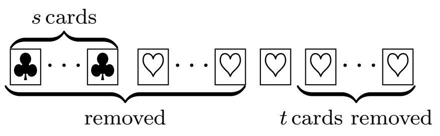
However, P_n only removes the rightmost or leftmost card and outputs the remaining card.

For instance, in the case where $n = 5$, the input/output relationship that the above procedures are applied is as shown in Table 4.5. This is an extension of Table 4.4, and it can be seen that the correct output is obtained. On the other hand, we can easily understand that the above procedures cannot obtain correct output if n is an even number because the number of "players who add one card" and "players who remove one card" is the same, and no output card remains. The case where n is an even number is explained in the next section. Here, we confirm that the above procedures satisfy correctness and security when n is an odd number.

Correctness: Let s be the number of players whose input value is 0 among the first half players $P_i\,(1 \leq i \leq (n+1)/2)$. Then, the card order received by $P_{(n+1)/2+1}$ is s cards are $\clubsuit$ and $(n+1)/2 - s$ cards are $\heartsuit$ from the left. Also, let t be the number of players whose input value is 0 among the second half players $P_j\,((n+1)/2 < j \leq n)$.

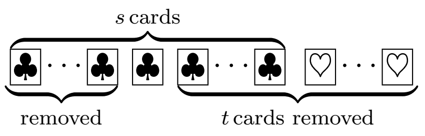
- If $s + t < (n+1)/2$

  Then, the number of players t to remove the rightmost card is less than $(n+1)/2 - s$. Therefore, the final remaining card is $\heartsuit$ representing 1 as shown below, and it is confirmed that the correct output can be obtained.



- If $s + t \geq (n+1)/2$

  Then, the number of players t to remove the rightmost card is $(n+1)/2 - s$ or more. Therefore, the final remaining card is $\clubsuit$ representing 0 as shown below, and it is confirmed that the correct output can be obtained.



□

Security: It is trivial that no information beyond the output is leaked since only the ourput card is opened and players' operations are hidden by the assumption of PP. □

Case Where n Is An Even Number

In order to realize n-input majority voting protocol where n is an even number, we utilize the following equivalence relation.

$$\mathsf{maj}_n(x_1, \ldots, x_n) = 1 \iff \mathsf{maj}_{n+1}(1, x_1, \ldots, x_n) = 1 \tag{4.7}$$

Namely, in the case where n is an even number, we realize n-input majority voting protocol by reducing $n+1$-input majority voting protocol. The specific procedure is the following:

Suppose the first player P_1' is a dummy player whose input is fixed at 1 in the $n+1$-input majority voting protocol. The result of this protocol is the same as the result of n-input majority voting by n players, excluding P_1', from the above equivalence relation. Thus, we can realize n-input majority voting protocol by executing $n+1$-input majority voting protocol from the state where P_1' inputs 1.

Note that the dummy player P_1' needs only $\heartsuit$ since she does not input 0. The reason for making the P_1', who is the role of adding a card, a dummy is this reduction of the number of cards. It is possible to make a player in the role of removing a card, e.g., P_n', as a dummy. However, in this case, one more card is required than when P_1' is used as a dummy.

Graphical Interpretation: Figure 4.1 shows that the mechanism of how to implement our majority voting protocol with even number inputs, using $n = 4$ as an example. The vertical axis means the threshold value and the horizontal one means the number of inputs. This figure shows the flow that reduces to the next majority voting (or threshold) function each time an input value is determined. The protocol is executed as a five-input majority voting, but the result is the same as a four-input majority voting since the input value of one player is fixed at 1 in advance.

4.3.2 Secure Computation for (t, n)-threshold Function with $n+1$ Cards

The idea to construct the threshold function protocol is essentially the same as the n-input majority voting protocol when n is even. We utilize the following equivalence relation. Here, we can assume $t < n/2$ without loss of

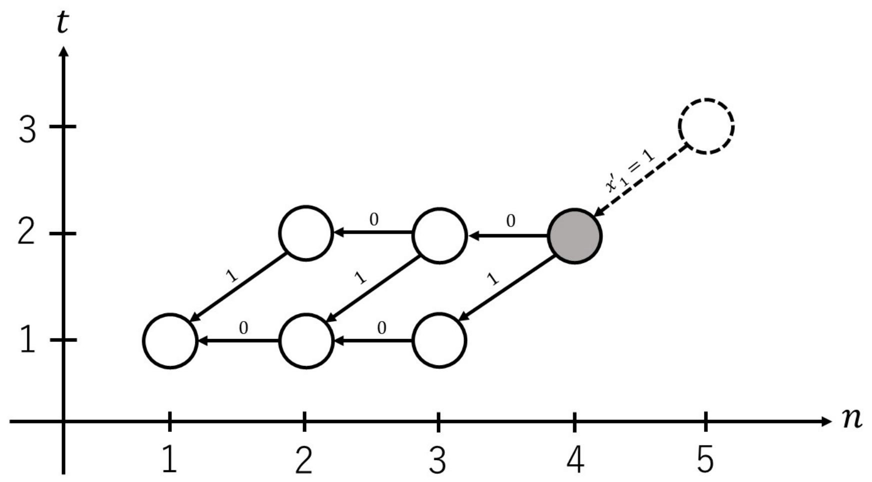


Figure 4.1: Graphical Interpretation of Four-input Majority Voting Protocol

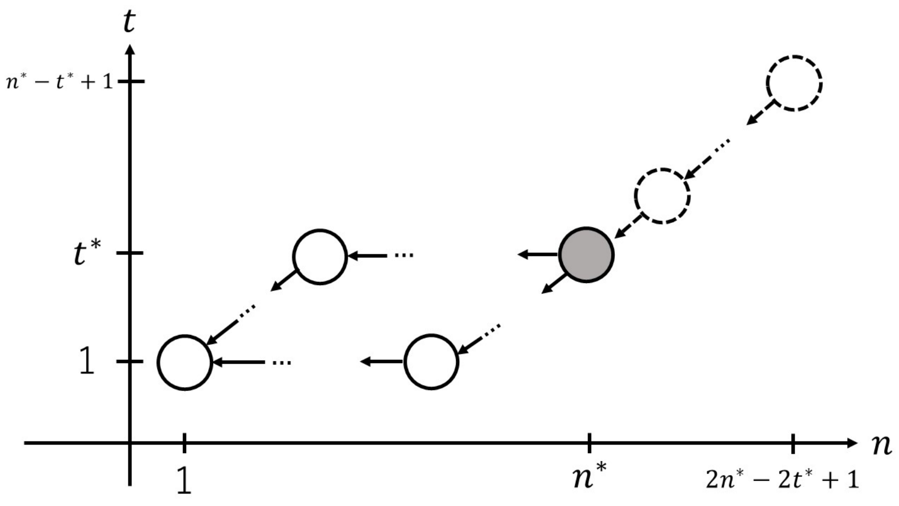


Figure 4.2: Graphical Interpretation of Threshold Function Protocol

generality since inputs 0 and 1 can be reversed if $t \geq n/2$. Also, even if 0 and 1 are not reversed, it is possible to realize the reduction by fixing the dummy input value to 0.

$$f_{(t,n)}(x_1, \dots, x_n) = 1 \iff f_{(t+1,n+1)}(1, x_1, \dots, x_n) = 1 \tag{4.8}$$

In other words, we realize the threshold function protocol by selecting an integer d such that $f_{(t+d,n+d)}$ is the odd number input majority function and reducing to $(n+d)$-input majority voting protocol.

Let $f_{(t,n)}$ be the threshold function to be computed. Then, we reduce this function to $f_{(n-t+1,2n-2t+1)} = \mathsf{maj}_{2n-2t+1}$. Thus, we can obtain the result of $f_{(t,n)}$ by executing the protocol for $\mathsf{maj}_{2n-2t+1}$ with $n-2t+1$ dummy players whose input values are 1. The specific procedure is shown in Protocol 9. $\heartsuit \times n-2t+1$ possessed by P_1 in setup are the inputs of dummy players. This protocol is constructed with $\clubsuit \times t$ and $\heartsuit \times n-t+1$.

Graphical Interpretation: Figure 4.2 shows that the mechanism of how to implement our threshold function protocol. This figure shows the flow that reduces to the next threshold function each time an input value is determined. The protocol is executed as a $(n-t+1, 2n-2t+1)$-threshold function, i.e., $2n-2t+1$-input majority voting, but the result is the same as (t,n)-threshold function since the input values of $n-2t+1$ players are fixed at 1 in advance.

4.4 Results and Discussion

In the previous chapter, we succeeded in reducing the number of cards in the basic protocols. Thus, this chapter showed that PP also works effectively in constructing protocols for more advanced functions.

Chapter 4 proposed the following three protocols.

- Section 4.2: Simultaneous AND/OR protocol with four cards

- Section 4.2: 3-input majority voting protocol with four cards
- Section 4.3: (t, n)-threshold function protocol with $n - 1$ cards

We first showed that the AND and OR protocols described in Chapter 3 could be improved to four-card AND/OR protocol, which can be simultaneously obtained AND and OR result in order to construct a protocol for 3-input majority based on this protocol. Our protocol for 3-input majority voting can be realized based on the AND/OR protocol without additional cards. As a result, we realize the protocol for majority voting with only four cards. This protocol can be extended to protocols for more voters. Utilizing this fact, we construct a threshold function protocol with $n - 1$ cards, which is fewer than the lower bound of traditional card-based cryptography. This protocol is more efficient in terms of not only the number of cards but also the number of PPs and communications than the existing protocol.

Protocol 9 (t, n)-threshold Function Protocol

Inputs: Let $x_1, \dots x_n \in \{0, 1\}$ be input values of each player.
Setup: $P_1, \dots, P_t$ each have a pair $\clubsuit\heartsuit$. In addition, P_1 has $\heartsuit \times n - 2t + 1$.

1) For $i = 1, \dots, t$, repeat the following operation with PP to the received cards.

    - If $x_i = 0$, P_i places face-down $\clubsuit$ on the leftmost, and sends the cards after processing to P_{i+1}.
    - If $x_i = 1$, P_i places face-down $\heartsuit$ on the rightmost, and sends the cards after processing to P_{i+1}.

    Suppose that P_1 only performs the operation to place a card.

2) For $j = t+1, \dots, n$, repeat the following operation with PP to the received cards.

    - If $x_j = 0$, then P_j removes the rightmost card, and sends the cards after processing to P_{j+1}.
    - If $x_j = 1$, then P_j removes the leftmost card, and sends the cards after processing to P_{j+1}.

    Suppose that P_n only performs the removal operation.

3) Open the remaining one card in public area.

    - If this card is $\clubsuit$, then the output value is 0.
    - If this card is $\heartsuit$, then the output value is 1.

Chapter 5

How to Solve Millionaires' Problem

5.1 Introduction

In traditioanal card-based-cryptography adopts the private model that assumes all operations are performed in public. Shuffle is the critical operation to achieve confidentiality even in this condition. However, card-based cryptography principle becomes different from algebraic MPC shuffle is too card-oriented operation.

The introduction of PP makes card-based cryptography closer to algebraic MPC. As a result, it is easier to mutually utilize knowledge between them. We demonstrate that a new card-based protocol can be obtained by converting Yao's (algebraic) millionaires' protocol into a card-based protocol (Millionaires' protocol I). This conversion can be easily derived if we understand how his protocol works. Thus, we explain the outline of Yao's protocol before describing the proposed protocol.

It is the mainstream to propose logic gate protocols in traditional card-based cryptography since it is known that every Boolean function can be computed with a combination of logic gates. Millionaires' protocol can also be constructed with a combination of logic gates. We compare our protocols and the protocol based on Algorithm 10. This algorithm involves $2m-1$ AND and $2m-2$ OR times. When executing these logic gates, the COPY operation [38] is necessary for copy $\neg a_i$ and b_i in each comparison of bits. Summarizing, $6\lceil \log d \rceil - 5$ shuffles are necessary for total to implement Protocol 10. We summarize our result in Table 5.1.

Although our protocol based on Yao's solution succeeds in improving the number of PPs and communications, the number of cards exponentially increases from the existing protocol. Thus, we propose another millionaires' protocol that also improves the number of cards (Millionaires' protocol II). We adopt the bitwise comparison for reducing the number of cards. Millionaires' protocol II compares bit by bit 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. Hence, we show how to manipulate the storage privately by using PPs. It is very interesting to note that the idea of Millionaires' protocol II is the same as that of the well-known logic puzzle "*The fork in the road.*" This observation will be introduced when explaining the idea of Millionaires' protocol II in Section 5.3.1.

Unfortunately, Millionares' protocol II still requires the same number of cards as the previous work, whereas the other measures are evidently improved. Hence, we discuss how to reduce the number of cards in Section 5.3.3. The main idea of this improvement is that inputs are not represented as the sequence of cards but are memorized in players' mind. This improvement enables Millionaires' protocol II to realize with only *six* cards.

The remaining part of this chapter is organized as follows: In Section 5.2, the card-based cryptography for the millionaires' problem based on Yao's protocol is presented. Section 5.3 is devoted to the proposal of a new card-

Protocol 10 Comparing Protocol Constructed by Logic Gates

```
Input: $a = (a_m \cdots a_2\, a_1)_2,\ b = (b_m \cdots b_2\, b_1)_2$
  $f_1 = \bar{a}_1 \wedge b_1$
  for $i = 2$ to $m$ do
    $f_i = \bar{a}_i \wedge b_i \vee (\bar{a}_i \vee b_i) \wedge f_{i-1}$ ;
  end for
Output: $f_m$
  if $f_n = 0$ then $a \geq b$
  if $f_n = 1$ then $a < b$
```

Table 5.1: Summary of Our Results in Chap. 5

Protocols	# of Comm.	# of PP	# of cards
logic gates (Algo. 10)	$6m-5$	$12m-10$	$4m+2$
Millionaires' protocol I (Yao)	1	2	$2 \cdot 2^m$
Millionaires' protocol II (storage)	$2m$	$2m+1$	$4m+2$
Improvement of millionaires' protocol II	$2m$	$2m+1$	6

based cryptographic protocol *with storage.* We also show an improvement in the proposed protocol that reduces the number of cards. We summarize this chapter in Section 5.4.

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

5.2.1 Our Idea Behind the Millionaires' Protocol I

We propose a card-based cryptography that resolves the millionaires' problem by cards based on Yao's original solution utilizing PPs. Before providing our protocol, we explain Yao's public key based solution [61] 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, where $a, b \in [d]$. Let $\mathcal{X} := [2^N - 1]$ be a set of N-bit integers where, $2^{N/2} > 2d$ is necessary to hold $|z_u - z_v| \geq 2$ in step $\langle 4 \rangle$ for all distinct $u, v \in [d]$. $(\mathsf{Enc}_A, \mathsf{Dec}_A)$ is a public-key encryption of Alice. That is, $\mathsf{Enc}_A : \mathcal{X} \to \mathcal{X}$ 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 the number $c - b + 1$ in the mod 2^N sense to Alice.

$\langle 3 \rangle$ For $i = 1, 2, \ldots, d$, Alice computes privately the values of $y_i = \mathsf{Dec}_A(c - b + i)$; each value $c - b + i$ is in the mod 2^N sense.

$\langle 4 \rangle$ Alice generates a random prime $p \in [2^{N/2} - 1]$, and computes the values $z_i := y_i \bmod p$, for $i = 1, 2, \ldots, d$. If $|z_u - z_v| \geq 2$ for all distinct $u, v \in [d]$, 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.

$\langle 5 \rangle$ Alice makes $z' = (z_1, z_2, \ldots, z_a, z_{a+1}+1, z_{a+2}+1, \ldots, z_d+1)$; each value is in the mod p sense.

$\langle 6 \rangle$ Alice sends p and the vector z' to Bob.

$\langle 7 \rangle$ Bob looks at the b-th number in z'. If it is equal to x mod p, then $a \geq b$, otherwise $a < b$.

$\langle 8 \rangle$ Bob sends the result to Alice.

Our Idea Behind Millionaires' 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 each of $z_{a+1}, z_{a+2}, \ldots, z_d$ 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 $\langle 3\rangle$ in the following millioniares' 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 cryptography, these steps can be replaced with *single step* since face down plays 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.

5.2.2 Millionaires' Protocol I

Based on the ideas discussed in the previous section, we propose a card-based protocol for the millionaires' problem based on Yao's solution (see Protocol 11). We refer to this protocol *Millionaires' protocol I.* The definitions of a, b and m are the same as the previous section. Let $\chi^{\mathsf{ge}}(\cdot,\cdot)$ be a function such that

$$\chi^{\mathsf{ge}}(u,v) := \begin{cases} 1 & \text{if } u \geq v \\ 0 & \text{otherwise,} \end{cases} \tag{5.1}$$

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

Note that steps 1) and 2) in Millionaires' protocol I correspond to steps $\langle 1\rangle$–$\langle 5\rangle$, and the steps 3) and 4) correspond to steps $\langle 6\rangle$ and $\langle 7\rangle$, 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 $(\mathsf{Enc}_A, \mathsf{Dec}_A)$ in Yao's millionaires' protocol must be public-key encryption since a is obtained by Bob in step $\langle 6\rangle$ if $(\mathsf{Enc}_A, \mathsf{Dec}_A)$ is a private key encryption. On the other hand, in Millionaires' protocol I, such leakage

Protocol 11 Millionaires' 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 $(\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_d)$ where $\vec{x}_1 = \vec{x}_2 = \cdots = \vec{x}_d = (♣, ♡)$.

2) For $i = 1, 2, \ldots, m$, repeat the operation in which Alice swaps $\vec{x}_i$ if $i > a$; otherwise does not. Each swap operation must be executed *privately*, and it is described as the following PP with respect to $\mathcal{R}_2^{\mathsf{bc}} := \{\pi_0, \pi_1\}$ which is given by (2.4) with $v = 1$:

$$\mathsf{PP}^{[2]}_{\mathcal{R}_2^{\mathsf{bc}}}(\vec{x}_i, \chi^{\mathsf{ge}}(i-1, a)),\ i = 1, 2, \ldots, d, \tag{5.2}$$

where $\chi^{\mathsf{ge}}(\cdot, \cdot)$ is defined in (5.1), i.e., $\chi^{\mathsf{ge}}(i-1, a) = 1$ iff $i > a$. As a result, Alice *privately* generates the sequence of cards $\vec{x}' := (\vec{x}'_1, \vec{x}'_2, \ldots, \vec{x}'_m)$ where $\vec{x}'_i := \mathsf{PP}^{[2]}_{\mathcal{R}_2^{\mathsf{bc}}}(\vec{x}_i, \chi^{\mathsf{ge}}(i-1, a))$.

3) Alice sends $\vec{x}'$ to Bob.

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

$$\mathsf{PP}^{[2d]}_{\mathcal{R}_{2d}^{\mathsf{mf}}}(\vec{x}', b-1) = \pi_{b-1}(\vec{x}') \tag{5.3}$$

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

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

of a to Bob is prevented by requiring that all cards except $\vec{x}'_b$ are completely randomized in public by Alice or Bob at the end of the protocol.

Efficiency of Millionaires' protocol I. In the proposed protocol, $2d$ 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 logic gates (see Algo. 10). We note that the steps 4) and 5) are necessary so that Bob turns $\vec{x}'_b$ face up publicly without making b public.[*1]

Theorem 1 *Millionaires' protocol I is perfectly secure; it satisfies* (2.7) *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 $\vec{x}'_b$ revealed in step 5), and the binary value represented by $\vec{x}'_b$ 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., Millionaires' 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, Millionaires' 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 Million-

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

aires' protocol I, whereas Bob is required to announce his result to Alice in Yao's original protocol (see step $\langle 8 \rangle$).

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

5.3.1 Ideas Behind Millionaires' Protocol II

In order to reduce the number of cards to below $2d$, it is natural to represent the wealth of Alice and Bob as binary number with $\lceil \log d \rceil$ bits (i.e., $2\lceil \log d \rceil$ 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 bits.

Let $(a_m, \dots, a_1)$ and $(b_m, \dots, b_1)$ be the binary representation of the positive integers a and b, respectively, where $m := \lceil \log d \rceil$ and $a_i, b_i \in \{0, 1\}$, $i = 1, 2, \dots, m$. For each $i \in [m]$, 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.[*2] 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 $(a_m, \dots, a_1)$ 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

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

by a card as well) called *comparison storage*, denoted by $cs \in \{♣, ♡\}$, 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 ♡ (i.e., 1) at the end of the protocol. Similarly, Alice is shown to be *richer* if the comparison storage is ♣ (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 $(a_m, \ldots, a_1)$ 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 $ds \in \{♣, ♡\}$, 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*. 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.[*3]

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).
- 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).

[*3]This problem is very similar to the well-known logical problem "*The fork in the road.*"

Table 5.2: Synchronization Mechanism in Millionaires' Protocol II

		(cs, ds), $w = l$			(ds, cs), $w = r$		
a_i $(\alpha_{i,l})$	b_i $(\beta_{i,l})$	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 $= ds$

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 m, 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 5.2 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 5.2). 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., cs, 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 logic puzzle "*The fork in the road*," [19, 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 Millionaires' 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 Millionaires' protocol II.

5.3.2 Millionaires' 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 12). For the upper bound $d \in \mathbb{N}$ of the wealth of Alice and Bob, let $m := \lceil \log d \rceil$.

[*4]This card can be arbitrary since it is a dummy card.

Protocol 12 Millionaires' Protocol II

1) Alice prepares a face-down ♣ and a face-down $\heartsuit^{*4}$ 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 $(\alpha_{m,l}\alpha_{m,r}, \ldots, \alpha_{2,l}\alpha_{2,r}, \alpha_{1,l}\alpha_{1,r})$, which is a binary representation of her wealth $a \in [d]$. Bob also prepares the sequence of $2n$ cards $(\beta_{m,l}\beta_{m,r}, \ldots, \beta_{2,l}\beta_{2,r}, \beta_{1,l}\beta_{1,r})$, which is the binary representation of his wealth $b \in [d]$.

2) For $i = 1, 2, \ldots, m$, 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 $\mathcal{R}_2^{\mathsf{bc}}$ which is defined in (2.4) with $v = 1$:

$$(\sigma_l, \sigma_r) := \mathsf{PP}^{[2]}_{\mathcal{R}_2^{\mathsf{bc}}}((cs, ds), \chi^{\mathsf{eq}}(w, r)) \tag{5.4}$$

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 $\alpha_{i,w}$. 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 $\mathcal{R}_3^{\mathsf{ow1}} := \{\pi_0, \pi_1\}$ where $\pi_0 : (1, 2, 3) \mapsto (1, 3, 2)$ and $\pi_1 : (1, 2, 3) \mapsto (3, 2, 1)$:

$$(\sigma_l', \sigma_r', \eta) := \mathsf{PP}^{[3]}_{\mathcal{R}_3^{\mathsf{ow1}}}((\sigma_l, \sigma_r, \beta_{i,l}), \overline{\chi^{\mathsf{eq}}}(\beta_{i,l}, \alpha_{i,w})) \tag{5.5}$$

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., executes the PP such that

$$\mathsf{PP}^{[2]}_{\mathcal{R}_2^{\mathsf{bc}}}((\sigma_l', \sigma_r'), \chi^{\mathsf{eq}}(w, l)), \tag{5.6}$$

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

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

Example of Millionaires' protocol II. We show a simple example for understanding how the Millionaires' 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), (5.4) 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 (5.5) 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 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, (5.4) 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 $\beta_{1,l} = \clubsuit$ 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 (5.5).

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 (5.6) 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 (5.4), 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 (5.5). Bob sends $(\sigma'_l, \sigma'_r) = (\heartsuit, \clubsuit)$ to Alice, and she obtains $(cs, ds) := (\heartsuit, \clubsuit)$ due to (5.6).

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 (5.4), 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 (5.5). Bob sends $(\sigma'_l, \sigma'_r) = (\clubsuit, \heartsuit)$ to Alice, and she obtains $(cs, ds) := (\heartsuit, \clubsuit)$ due to (5.6).

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

Efficiency of Millionaires' protocol II. This protocol requires two communications for every bit therefore it requires $2\lceil \log d \rceil$ 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 $2\lceil \log d \rceil + 1$ PPs. The number of cards is $4\lceil \log d \rceil + 2$.

Theorem 2 *Millionaires' protocol II is perfectly secure; it satisfies* (2.7) *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 $R_A = (W_1, W_2, \ldots, W_m)$ where W_i is the random variable corresponding to w in the i-th loop in step 2). Each random variable W_i, $i = 1, 2, \ldots, m$ 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 Millionaires' 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 m values represented by the face-up cards in step (2-iii), i.e., $\Lambda =$

$(\alpha_{1,W_1}, \alpha_{2,W_2}, \dots, \alpha_{m,W_m})$. Hence, Λ is easily simulated by S_A by a which is represented by her $2m$ cards, and the random variable $R_A = (W_1, W_2, \dots, W_m)$. On the other hand, for Bob, $\Lambda = (\alpha_{1,W_1}, \alpha_{2,W_2}, \dots, \alpha_{m,W_m})$ seems to be uniformly distributed over $\{\heartsuit, \clubsuit\}^m$ 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. $\square$

5.3.3 Millionaires' Problem Can Be Solved with Only Six Cards

Although Millionaires' protocol II was improved in terms of the numbers of PPs and communications, as is shown in Table 5.1, it still requires the same number of cards with the previous work based on logic gates. However, we show that Millionaires' 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 12.[*5] 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 12, 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 $w = l$ 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

[*5]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 [47].

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 12. Concretely, if η is in two-card representation, Bob returns randomized η to Alice instead of discarding it by Bob in Protocol 12. 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 Protocols 13. 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 Millionaires' protocol II. This protocol requires two communications for every bit therefore it requires $2\lceil \log d \rceil$ 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 $2\lceil \log d \rceil + 1$ PPs.

5.4 Results and Discussion

This chapter proposed the following three efficient card-based protocols for the millionaires' problem by utilizing PP.

- Section 5.2: Millionaires' protocol I (based on Yao's solution [61])

Protocol 13 Improvement of Millionaires' 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 $i = 1, 2, \ldots, m$, repeat the following operations.

   (2-i) Alice chooses $\alpha_i \in \{0, 1\}$ uniformly at random privately. Then, execute the following permutation:

   $$(\sigma_l, \sigma_r) := \mathsf{PP}^{[4]}_{\mathcal{R}^{\mathsf{bc}}_4}((cs, ds), \overline{\chi^{\mathsf{eq}}}(\alpha_i, a_i)) \tag{5.7}$$

   (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 $\mathcal{R}^{\mathsf{ow2}}_6 := \{\pi_0, \pi_1\}$ where $\pi_0 : (1, 2, 3, 4, 5, 6) \mapsto (1, 2, 5, 6, 3, 4)$ and $\pi_1 : (1, 2, 3, 4, 5, 6) \mapsto (5, 6, 3, 4, 1, 2)$:

   $$(\sigma'_l, \sigma'_r, \eta) := \mathsf{PP}^{[6]}_{\mathcal{R}^{\mathsf{ow2}}_6}((\sigma_l, \sigma_r, b_i), \overline{\chi^{\mathsf{eq}}}(b_i, \alpha_i)) \tag{5.8}$$

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

   $$\eta' := \mathsf{PP}^{[2]}_{\mathcal{R}^{\mathsf{bc}}_2}(\eta, r_b) \tag{5.9}$$

   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

   $$\mathsf{PP}^{[4]}_{\mathcal{R}^{\mathsf{bc}}_4}((\sigma'_l, \sigma'_r), \overline{\chi^{\mathsf{eq}}}(\alpha_i, a_i)), \tag{5.10}$$

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

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

   $$\eta'' := \mathsf{PP}^{[2]}_{\mathcal{R}^{\mathsf{bc}}_2}(\eta', r_a) \tag{5.11}$$

   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.

- Section 5.3.2: Millionaires' protocol II
- Section 5.3.3: Improvement of Millionaires' protocol II

Millionaires' protocol I is constructed by directly converting Yao's solution into card-based cryptography. This result is due to bridge the gap between algebraic MPC and card-based cryptography by introducing PP. This is the first achievement of constructing a new card-based protocol by converting algebraic MPC into card-based cryptography.[*6] Millionaires' protocol succeeded in improving the number of PPs and communications. However, the number of cards exponentially increases from the existing protocol. It is worth mentioning that millionaires' 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.

Millionaires' 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 logic 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 millionaires' protocol II and the well-known logic puzzle known as "The fork in the road" are deeply related. However, this protocol is not made efficient from the viewpoint of the number of cards (see Table 5.1). Hence we proposed a method to reduce the number of cards. The improved protocol works only six cards by memorizing the inputs in players' minds without representing them using cards.[*7] This protocol is the

[*6]After the conference version of this work was published, the protocol based on Yao's solution was proposed in public model [30].

[*7]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 [48].

most potent result to claim that PP is effective in improving the efficiency of card-based cryptography.

Chapter 6

Conclusion

Traditional card-based cryptography is based on the assumption that all operations are performed in public. Although this assumption has the advantage of preventing cheats, it causes the lower bound of the number of cards because it limits the input representation to use face-down cards. Also, traditional card-based cryptography utilizes “shuffle” to achieve confidentiality in a situation where all operations are published. However, shuffle is based on too card-oriented assumption, and thus it results that card-based cryptography framework is different from algebraic MPC, which is based on “private randomness.”

From the awareness of the above problem, we proposed a new card-based cryptography model that is introduced a new operation, “private permutation (PP),” which is naturally derived from private randomness. The results obtained by introducing PP are as follows:

Chapter 3: Card-based protocols for logic gates

PP allowed input values to be expressed without using face-down cards. Chapter 3 focused on logic gates, which is the mainstream in previous work. At least four cards were required to construct a logic gate protocol, but it can be realized with a smaller number of cards by using PP as the input representation. The protocols proposed in Chapter 3 are as follows.

- Section 3.3.1: three-card OR protocol
- Section 3.3.2: two-card XOR protocol

Chapter 4: Card-based threshold function protocol

Chapter 3 succeeded in reducing the number of cards in the basic protocols. Thus, Chapter focused on the more advanced function, the threshold function. Then, it was a mainstream that advanced functions were realized by a combination of logic gates. However, this general method has a problem that the number of shuffles increases. Thus, we did not adopt this method but a specific construction to improve efficiency. The protocols proposed in Chapter 3 are as follows.

- Section 4.2: Simultaneous AND/OR protocol with four cards
- Section 4.2: 3-input majority voting protocol with four cards
- Section 4.3: (t, n)-threshold function protocol with $n-1$ cards

Chapter 5: Card-based protocols for millionaires' problem: PP removed the card-oriented assumption of shuffle. As a result, card-based cryptography framework became closer to algebraic MPC, and it became easier to use techniques between them. We demonstrated that Yao's algebraic millionaires' protocol could be converted into a card-based protocol. Although this protocol is improved in the viewpoint of the number of PPs and communications, the number of cards exponentially increases. Thus, we proposed another millionaires' protocol. This protocol was based on the bitwise comparison. Finally, we showed millionaires' protocol could be realized with only six cards. The interesting point of this protocol was to utilize the famous logic puzzle "The fork in the road" in order to avoid information leaking in the bitwise comparison. This protocol was the most potent result of showing the power of PP. The protocols proposed in Chapter 5 are as follows.

- Section 5.2: Millionaires' protocol I (based on Yao's solution [61])
- Section 5.3.2: Millionaires' protocol II
- Section 5.3.3: Improvement of Millionaires' protocol II

Finally, we describe the future work in the following.

- *Generic construction in our model:* We explained two directions for the protocol construction, a generic one and a specialized one, in Section 1.1. All of our proposed protocols include in the generic construction. Namely, it is not evident whether PP is also useful in generic construction, such as the combination of logic gates. Thus, it is required to verify the usefulness by constructing generic protocols using PP. Then, it is necessary to discuss the conditions for composing protocols securely.

- *New algebraic protocol based on card-based protocol:* In this thesis, we succeeded in obtaining a new card-based protocol by converting an algebraic MPC protocol into a card-based protocol. However, there is no example of creating a new algebraic MPC protocol from card-based cryptography. Thus, it is a challenge to verify whether card-based cryptography effectively contributes to the discovery of new algebraic MPC protocols.

- *Introduction of other private operations:* In this thesis, we introduced the only permutation as the private operation. On the other hand, we can see other private operations by observing general card-games. For instance, poker allows each player to look at the cards' surface without showing it to other players. It is an interesting problem whether the introduction of other private operations improves efficiency.

Bibliography

[1] Y. Abe, Y. Hayashi, T. Mizuki, and H. Sone. Five-card and protocol in committed format using only practical shuffles. In *Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop*, APKC '18, page 3–8, New York, NY, USA, 2018. Association for Computing Machinery.

[2] Y. Abe, M. Iwamoto, and K. Ohata. How to detect malicious behaviors in a card-based majority voting protocol with three inputs. In *2020 International Symposium on Information Theory and Its Applications (ISITA)*, pages 377–381, 2020.

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

[4] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, page 1–10, New York, NY, USA, 1988. Association for Computing Machinery.

[5] J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *Proceedings of the Twenty-Sixth Annual ACM Sym-*

posium on Theory of Computing, STOC ’94, page 544–553, New York, NY, USA, 1994. Association for Computing Machinery.

[6] C. Boyd. A new multiple key cipher and an improved voting scheme. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology — EUROCRYPT ’89*, pages 617–625, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg.

[7] R. Canetti. Security and composition of multiparty cryptographic protocols. *J. Cryptol.*, 13(1):143–202, Jan. 2000.

[8] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, FOCS ’01, page 136, USA, 2001. IEEE Computer Society.

[9] R. Canetti and M. Fischlin. Universally composable commitments. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 2001.

[10] D. Chaum, C. Crépeau, and I. Damgard. Multiparty unconditionally secure protocols. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC ’88, page 11–19, New York, NY, USA, 1988. Association for Computing Machinery.

[11] R. Cramer, I. B. Damgrd, and J. B. Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, USA, 1st edition, 2015.

[12] C. Crépeau and J. Kilian. Discreet solitary games. In *Advances in Cryptology - CRYPTO ’93, 13th Annual International Cryptology Con-*

ference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings, pages 319–330, 1993.

[13] P. D'Arco and R. De Prisco. Secure two-party computation: A visual way. pages 18–38, 01 2014.

[14] P. D'Arco and R. De Prisco. Secure computation without computers. *Theor. Comput. Sci.*, 651(C):11–36, Oct. 2016.

[15] B. den Boer. More efficient match-making and satisfiability: *The Five Card Trick*. In *Advances in Cryptology - EUROCRYPT '89, Workshop on the Theory and Application of of Cryptographic Techniques, Houthalen, Belgium, April 10-13, 1989, Proceedings*, pages 208–217, 1989.

[16] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology*, pages 10–18, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

[17] R. Fagin, M. Naor, and P. Winkler. Comparing information without leaking it. *Commun. ACM*, 39(5):77–85, May 1996.

[18] M. K. Franklin and M. K. Reiter. The design and implementation of a secure auction service. *IEEE Trans. Softw. Eng.*, 22(5):302–312, May 1996.

[19] M. Gardner. Hexaflexagons and other mathematical diversions: The first scientific american book of mathematical puzzles and games. Univ of Chicago Press, 1956.

[20] C. Gentry. *A Fully Homomorphic Encryption Scheme*. PhD thesis, Stanford, CA, USA, 2009.

[21] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, page 218–229, New York, NY, USA, 1987. Association for Computing Machinery.

[22] R. Gradwohl, M. Naor, B. Pinkas, and G. N. Rothblum. Cryptographic and physical zero-knowledge proof systems for solutions of sudoku puzzles. In *Proceedings of the 4th International Conference on Fun with Algorithms*, FUN'07, page 166–182, Berlin, Heidelberg, 2007. Springer-Verlag.

[23] J. Kastner, A. Koch, S. Walzer, D. Miyahara, Y.-i. Hayashi, T. Mizuki, and H. Sone. The minimum number of cards in practical card-based protocols. In T. Takagi and T. Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 126–155, Cham, 2017. Springer International Publishing.

[24] A. Koch. *Cryptographic Protocols from Physical Assumptions.* PhD thesis, Karlsruher Institut für Technologie (KIT), 2019.

[25] A. Koch, M. Schrempp, and M. Kirsten. *Card-Based Cryptography Meets Formal Verification*, pages 488–517. 11 2019.

[26] A. Koch and S. Walzer. Foundations for actively secure card-based cryptography. In *IACR Cryptol. ePrint Arch.*, 2017.

[27] A. Koch, S. Walzer, and K. Härtel. Card-based cryptographic protocols using a minimal number of cards. In *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part I*, pages 783–807, 2015.

[28] Y. Komano and T. Mizuki. *Multi-party Computation Based on Physical Coins: 7th International Conference, TPNC 2018, Dublin, Ireland, December 12–14, 2018, Proceedings*, pages 87–98. 01 2018.

[29] A. Marcedone, Z. Wen, and E. Shi. Secure dating with four or fewer cards. Cryptology ePrint Archive, Report 2015/1031, 2015. `https://eprint.iacr.org/2015/1031`.

[30] D. Miyahara, Y.-i. Hayashi, T. Mizuki, and H. Sone. Practical card-based implementations of yao's millionaire protocol. *Theoretical Computer Science*, 803, 11 2019.

[31] T. Mizuki. Efficient and secure multiparty computations using a standard deck of playing cards. In *Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings*, pages 484–499, 2016.

[32] T. Mizuki, I. K. Asiedu, and H. Sone. Voting with a logarithmic number of cards. In *Unconventional Computation and Natural Computation - 12th International Conference, UCNC 2013, Milan, Italy, July 1-5, 2013. Proceedings*, pages 162–173, 2013.

[33] T. Mizuki, Y. Kugimoto, and H. Sone. Secure multiparty computations using a dial lock. pages 499–510, 01 2007.

[34] T. Mizuki, Y. Kugimoto, and H. Sone. Secure multiparty computations using the 15 puzzle. pages 255–266, 08 2007.

[35] T. Mizuki, M. Kumamoto, and H. Sone. The five-card trick can be done with four cards. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, pages 598–606, 2012.

[36] T. Mizuki and H. Shizuya. A formalization of card-based cryptographic protocols via abstract machine. *Int. J. Inf. Sec.*, 13(1):15–23, 2014.

[37] T. Mizuki and H. Shizuya. Practical card-based cryptography. In *Fun with Algorithms - 7th International Conference, FUN 2014, Lipari Island, Sicily, Italy, July 1-3, 2014. Proceedings*, pages 313–324, 2014.

[38] T. Mizuki and H. Sone. Six-card secure AND and four-card secure XOR. In *Frontiers in Algorithmics, Third International Workshop, FAW 2009, Hefei, China, June 20-23, 2009. Proceedings*, pages 358–369, 2009.

[39] T. Moran and M. Naor. Basing cryptographic protocols on tamper-evident seals. In *Proceedings of the 32nd International Conference on Automata, Languages and Programming*, ICALP'05, page 285–297, Berlin, Heidelberg, 2005. Springer-Verlag.

[40] T. Moran and M. Naor. Polling with physical envelopes: A rigorous analysis of a human-centric protocol. In S. Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 88–108, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[41] S. Murata, D. Miyahara, T. Mizuki, and H. Sone. Public-pez cryptography. In W. Susilo, R. H. Deng, F. Guo, Y. Li, and R. Intan, editors, *Information Security*, pages 59–74, Cham, 2020. Springer International Publishing.

[42] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, EC '99, page 129–139, New York, NY, USA, 1999. Association for Computing Machinery.

[43] V. Niemi and A. Renvall. Secure multiparty computations without computers. *Theor. Comput. Sci.*, 191(1-2):173–183, 1998.

[44] T. Nishida, Y. Hayashi, T. Mizuki, and H. Sone. Card-based protocols for any boolean function. In *Theory and Applications of Models of Computation - 12th Annual Conference, TAMC 2015, Singapore, May 18-20, 2015, Proceedings*, pages 110–121, 2015.

[45] T. Nishida, T. Mizuki, and H. Sone. Securely computing the three-input majority function with eight cards. In *Theory and Practice of Natural Computing - Second International Conference, TPNC 2013, Cáceres, Spain, December 3-5, 2013, Proceedings*, pages 193–204, 2013.

[46] A. Nishimura, Y. Hayashi, T. Mizuki, and H. Sone. An implementation of non-uniform shuffle for secure multi-party computation. In *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography, AsiaPKC@AsiaCCS, Xi'an, China, May 30 - June 03, 2016*, pages 49–55, 2016.

[47] H. Ono and Y. Manabe. Efficient card-based cryptographic protocols for the millionaires' problem using private input operations. In *2018 13th Asia Joint Conference on Information Security (AsiaJCIS)*, pages 23–28, 2018.

[48] H. Ono and Y. Manabe. Card-based cryptographic protocols with the minimum number of cards using private operations. In N. Zincir-Heywood, G. Bonfante, M. Debbabi, and J. Garcia-Alfaro, editors, *Foundations and Practice of Security*, pages 193–207, Cham, 2019. Springer International Publishing.

[49] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.

[50] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, page 73–85, New York, NY, USA, 1989. Association for Computing Machinery.

[51] T. Sasao. Switching theory for logic synthesis. Kluwer Academic Publishers, 1999. Norwell, MA, USA, 1st edn.

[52] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[53] A. Shamir, R. L. Rivest, and L. M. Adleman. Mental poker. Mental Poker. Technical Memo LCS/TM-125, Massachusetts Institute of Technology, 1979.

[54] K. Shinagawa, T. Mizuki, J. Schuldt, K. Nuida, N. Kanayama, T. Nishide, G. Hanaoka, and E. Okamoto. Multi-party computation with small shuffle complexity using regular polygon cards. volume 9451, pages 127–146, 11 2015.

[55] K. Shinagawa, T. Mizuki, J. C. N. Schuldt, K. Nuida, N. Kanayama, T. Nishide, G. Hanaoka, and E. Okamoto. Secure computation protocols using polarizing cards. *IEICE Transactions*, 99-A(6):1122–1131, 2016.

[56] K. Shinagawa and K. Nuida. A single shuffle is enough for secure card-based computation of any boolean circuit. *Discrete Applied Mathematics*, 289:248 – 261, 2021.

[57] A. Stiglic. Computations with a deck of cards. *Theor. Comput. Sci.*, 259(1-2):671–678, 2001.

[58] K. Toyoda, D. Miyahara, T. Mizuki, and H. Sone. Secure computation of three-input majority function using six cards. In *Computer Security Symposium (CSS)*, pages 4D1–4, 2020.

[59] I. Ueda, D. Miyahara, A. Nishimura, Y. Hayashi, T. Mizuki, and H. Sone. Secure implementations of a random bisection cut. *Int. J. Inf. Sec.*, 19(4):445–452, 2020.

[60] I. Ueda, A. Nishimura, Y. Hayashi, T. Mizuki, and H. Sone. How to implement a random bisection cut. In *Theory and Practice of Natural Computing - 5th International Conference, TPNC 2016, Sendai, Japan, December 12-13, 2016, Proceedings*, pages 58–69, 2016.

[61] A. C.-C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pages 160–164, Washington, DC, USA, 1982. IEEE Computer Society.

[62] A. C.-C. Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167. IEEE Computer Society, 1986.

List of Figures and Tables

List of Protocols

Chapter 5

- Protocol 10: Comparing Protocol Constructed by Logic Gates
- Protocol 11: Millionaires' Protocol I: Card-Based Yao's Solution
- Protocol 12: Millionaires' Protocol II: for Millionaires' Problem with Storage
- Protocol 13: Improvement of Millionaires' Protocol II

List of Notations

Notation for Private Permutaion

- $[t] := 1, 2, \ldots, t$ for a positive integer t
- $\vec{c} \in \{\clubsuit, \heartsuit\}^t$: a vector consisting of t face-down cards
- $\mathcal{P}_t$: a set of all permutations over $[t]$
- $\mathcal{R}_t$: a subset of $\mathcal{P}_t$
- $\mathsf{PP}^{[t]}_{\mathcal{R}_t}(\vec{c}, s) := \pi_s(\vec{c})$: the private permutation over $\mathcal{R}_t = \{\pi_0, \pi_1, \ldots, \pi_{|\mathcal{R}_t|-1}\}$

List of Publications

Journal Paper

1. Takeshi Nakai, Yuto Misawa, Yuuki Tokushige, Mitsugu Iwamoto, Kazuo Ohta: "How to Solve Millionaires' Problem with Two Kinds of Cards," New Generation Computing, online first, 2021.

Refered Conference Papers and Posters

2. Takeshi Nakai, Yuto Misawa, Yuuki Tokushige, Mitsugu Iwamoto, Kazuo Ohta: "Toward Reducing Shuffling in Card-Based Cryptographic Protocol for Millionaire Problem," In 10th International Workshop on Security, IWSEC2015, Poster Session, 2015.

3. Takeshi Nakai, Yuto Misawa, Yuuki Tokushige, Mitsugu Iwamoto, Kazuo Ohta: "Efficient Card-based Cryptographic Protocols for Millionaires Problem Utilizing Private Permutations," In 15th International Conference on Cryptography and Netework Security, CANS2016, LNCS, vol. 10052, pp. 500–517, 2016.

4. Takeshi Nakai, Satoshi Shirouchi, Mitsugu Iwamoto, Kazuo Ohta: "Four Cards Are Sufficient for a Card-based Three-Input Voting Protocol Utilizing Private Permutations," In 10th International Conference on

Information Theoretic Security, ICITS2017, LNCS, vol. 10681, pp. 153–165, 2017.

Non-Refered Conference Papers

5. 中井雄士, 徳重佑樹, 岩本貢, 太田和夫: “カードを用いた効率的な金持ち比べプロトコル,” 暗号と情報セキュリティシンポジウム 2015, SCIS2015, 3F4-2, 2015.

6. 徳重佑樹, 中井雄士, 岩本貢, 太田和夫: “カードベース暗号プロトコルにおける安全な選択処理,” 暗号と情報セキュリティシンポジウム 2015, SCIS2015, 3F4-3, 2015.

7. 中井雄士, 三澤裕人, 徳重佑樹, 岩本貢, 太田和夫: “カード操作の分類とカードベース暗号プロトコル,” 暗号と情報セキュリティシンポジウム 2016, SCIS2016, 4A2-2, 2016.

8. 徳重佑樹, 中井雄士, 岩本貢, 太田和夫: “カードを用いた複数人での金持ち比べプロトコル,” 暗号と情報セキュリティシンポジウム 2017, SCIS2017, 1A2-1, 2017.

9. 城内聡志, 中井雄士, 岩本貢, 太田和夫: “秘匿操作を用いた効率的なカードベース論理演算プロトコル,” 暗号と情報セキュリティシンポジウム 2017, SCIS2017, 1A2-2, 2017.

10. 中井雄士, 野島拓也, 岩本 貢, 太田 和夫: “検索可能暗号における最小漏洩情報に関する考察,” IT・ISEC・WBS 合同研究会, pp. 187–192, 2017.