Evaluating card-based protocols in terms of execution time

Daiki Miyahara^{1,2} · Itaru Ueda¹ · Yu-ichi Hayashi³ · Takaaki Mizuki⁴ · Hideaki Sone⁴

Published online: 27 November 2020
 © The Author(s) 2020

Abstract

Card-based cryptography is an attractive and unconventional computation model; it provides secure computing methods using a deck of physical cards. It is noteworthy that a card-based protocol can be easily executed by non-experts such as high school students without the use of any electric device. One of the main goals in this discipline is to develop efficient protocols. The efficiency has been evaluated by the number of required cards, the number of colors, and the average number of protocol trials. Although these evaluation metrics are simple and reasonable, it is difficult to estimate the total number of operations or execution time of protocols based only on these three metrics. Therefore, in this paper, we consider adding other metrics to estimate the execution time of protocols more precisely. Furthermore, we actually evaluate some of the important existing protocols using our new criteria.

Keywords Cryptography · Card-based protocols · Real-life hands-on cryptography · Secure multiparty computations

1 Introduction

Card-based protocols are unconventional computing methods using a deck of physical cards; their advantage is that they can be executed by humans practically (e.g., [4,8,11,16]). To illustrate this, let us explain how to manipulate Boolean values based on a two-colored deck of cards. Given a black card

and a red card , a Boolean value can be expressed as:

$$\begin{bmatrix} \clubsuit & \heartsuit \end{bmatrix} = 0, \begin{bmatrix} \heartsuit & \clubsuit \end{bmatrix} = 1. \quad (1)$$

Following this encoding, for example, two players, Alice and Bob, can each put two cards face down on a table, representing their private bits a and b , respectively:

$$\underbrace{\begin{bmatrix} ? & ? \end{bmatrix}}_a \quad \underbrace{\begin{bmatrix} ? & ? \end{bmatrix}}_b. \quad (2)$$

Here, we assume that the backs of all cards are indistinguishable and that the fronts or are also indistinguishable if the cards have the same color. We call the left pair of two face-down cards in (2) a *commitment* to a . Similarly, the right pair of two face-down cards are a commitment to b .

Typically, given two input commitments to $a, b \in \{0, 1\}$, as in (2), a card-based protocol should generate a commitment to the value of a predetermined function $f(a, b)$. For instance, we can get a commitment to $a \wedge b$ without leaking any information about a and b , if we execute an AND protocol:

$$\underbrace{\begin{bmatrix} ? & ? \end{bmatrix}}_a \quad \underbrace{\begin{bmatrix} ? & ? \end{bmatrix}}_b \rightarrow \dots \rightarrow \underbrace{\begin{bmatrix} ? & ? \end{bmatrix}}_{a \wedge b}.$$

An earlier version of this paper was presented at the 17th International Conference on Unconventional Computation and Natural Computation, Fontainebleau, France, June 25–29, 2018, and appeared in Proc. UCN 2018, Lecture Notes in Computer Science, vol. 10867, Springer, pp.145–158, 2018 [7].

✉ Daiki Miyahara
 daiki.miyahara.q4@dc.tohoku.ac.jp
 Takaaki Mizuki
 tm-paper+cardtime@g-mail.tohoku-university.jp

- ¹ Graduate School of Information Sciences, Tohoku University, 6–3 Aramaki-Aza-Aoba, Aoba, Sendai 980-8578, Japan
- ² National Institute of Advanced Industrial Science and Technology, 2–3–26, Aomi, Koto-ku, Tokyo 135-0064, Japan
- ³ Graduate School of Science and Technology, Nara Institute of Science and Technology, 8916–5 Takayama, Ikoma, Nara 630–0192, Japan
- ⁴ Cyberscience Center, Tohoku University, 6–3 Aramaki-Aza-Aoba, Aoba, Sendai 980-8578, Japan

Table 1 The existing AND protocols (in committed format)

	Year	#Colors	#Cards	Avg.#Trials	Shuffle
Crépeau & Kilian [3]	1993	4	10	6	Uniform
Niemi & Renvall [15]	1998	2	12	2.5	Uniform
Stiglic [23]	2001	2	8	2	Uniform
Mizuki & Sone [13]	2009	2	6	1	Uniform
Five-card KWH [6]	2015	2	5	1	Non-uniform
Four-card KWH [6]	2015	2	4	3	Non-uniform
Abe & Hayashi & Mizuki & Sone [1]	2018	2	5	5	Uniform

As in Table 1, there are many existing AND protocols (in committed format¹). This table implies that the design of “efficient” protocols is one of the goals of card-based protocols; so far, the efficiency has been evaluated in terms of three metrics: (i) the number of required cards, (ii) the number of colors, and (iii) the average number of required trials. These evaluation metrics are simple and reasonable. However, if we are going to actually execute a card-based protocol, these three metrics are insufficient to accurately estimate the number of operations that need to be done during the protocol and the overall execution time of the protocol.

Therefore, in this paper, we introduce new metrics to evaluate protocol efficiency more precisely. That is, we determine all the operations during a protocol to help analyze the execution time. Furthermore, we actually evaluate all the AND protocols² shown in Table 1, based on our new criteria by counting the number of operations thoroughly. As an application example, we also make a comparison of the AND protocols and discuss which protocol is the most efficient and practical.

The rest of this paper is organized as follows: In Sect. 2, we introduce the AND protocol invented by Stiglic [23] as an example and then give a formalization of the operations in card-based protocols [10]. In Sect. 3, we give new metrics of efficiency, which directly indicate the execution time of a protocol. In Sect. 4, we evaluate the existing AND protocols based on our proposed metrics. In Sect. 5, we conduct a further investigation into evaluation of the execution time. We conclude this study in Sect. 6.

An earlier version of this paper was presented and appeared as a conference paper [7]. The difference is as follows. This paper takes the AND protocol recently proposed by Abe et al. [1] into account, so that Sect. 3.2.4 along with Fig. 8 has been added and Sect. 4 has been updated. Furthermore, this paper first creates a full version of the KWH-tree [6] of the Crépeau–Kilian AND protocol [3] as Fig. 6 in Sect. 3.2.2 and that of the Niemi–Renvall AND protocol [15]

as Fig. 7 in Sect. 3.2.3. In addition, this paper investigates the execution time of rearrangement operations for cards more precisely; Sect. 5 is devoted to this new treatment.

2 Preliminaries: a protocol with operations

In this section, we introduce Stiglic’s AND protocol [23] as an example to demonstrate the possible operations in card-based protocols. It should be noted that card-based protocols are outside the Turing model [10,12].

As seen in Table 1, Stiglic’s protocol requires a two-colored deck of eight cards and two average trials. Given input commitments to a and b along with four additional cards $\clubsuit, \clubsuit, \heartsuit, \heartsuit$, the protocol proceeds as follows:

1. Arrange the sequence as:

$$\underbrace{? \ ? \ \heartsuit \ \clubsuit}_{a} \underbrace{? \ ? \ \clubsuit \ \heartsuit}_{b} \rightarrow \underbrace{? \ ? \ ? \ ?}_{a} \underbrace{? \ ?}_{1} \underbrace{? \ ?}_{b} \underbrace{? \ ?}_{0}.$$

2. Apply a *random cut* to the sequence of eight cards:

$$\langle ? \ ? \ ? \ ? \ ? \ ? \ ? \ ? \rangle \rightarrow ? \ ? \ ? \ ? \ ? \ ? \ ? \ ?.$$

The term “random cut” means a cyclic shuffle. If we attach numbers to the cards for the sake of convenience:

$$\begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ ? & ? & ? & ? & ? & ? & ? & ? \end{array},$$

then a random cut results in one of the following eight sequences (with a probability of $\frac{1}{8}$):

$$\begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ ? & ? & ? & ? & ? & ? & ? & ? \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 1 \\ ? & ? & ? & ? & ? & ? & ? & ? \\ \vdots & & & & & & & \\ 8 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ ? & ? & ? & ? & ? & ? & ? & ? \end{array}.$$

¹ There are also “non-committed-format” AND protocols [2,9].

² This paper addresses only AND computation because the other important primitive, XOR, can be done with only four cards and one trial [13].

Note that a random cut is known to be easily implemented by humans securely via the Hindu cut [25] as shown in Fig. 3.

3. Turn over the first two cards (from the left).

(a) If the revealed cards are $\heartsuit \heartsuit$, we obtain a commitment to $a \wedge b$ as follows:

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \heartsuit & \heartsuit & ? & ? & ? & ? & ? \\ \hline \end{array} \quad \underbrace{\hspace{1.5cm}}_{a \wedge b}.$$

(b) If the revealed cards are $\clubsuit \clubsuit$, we obtain

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \clubsuit & \clubsuit & ? & ? & ? & ? & ? \\ \hline \end{array} \quad \underbrace{\hspace{1.5cm}}_{a \wedge b}.$$

(c) If the revealed cards are $\clubsuit \heartsuit$ or $\heartsuit \clubsuit$, turn over the third card.

i. If the three face-up cards are $\heartsuit \clubsuit \clubsuit$, we have

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \heartsuit & \clubsuit & \clubsuit & ? & ? & ? & ? \\ \hline \end{array} \quad \underbrace{\hspace{1.5cm}}_{a \wedge b}.$$

ii. If the three face-up cards are $\clubsuit \heartsuit \heartsuit$, we have

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \clubsuit & \heartsuit & \heartsuit & ? & ? & ? & ? \\ \hline \end{array} \quad \underbrace{\hspace{1.5cm}}_{a \wedge b}.$$

iii. If the three face-up cards are $\clubsuit \heartsuit \clubsuit$ or $\heartsuit \clubsuit \heartsuit$, turn them over and go back to Step 2.

This is Stiglic's AND protocol, which we denote by $\mathcal{P}_{\text{Sti}}$ hereinafter. A shuffling operation called a random cut is used in Step 2 of $\mathcal{P}_{\text{Sti}}$. The average number of trials is two, because the probability that Step 3–(c)–iii occurs and we go back to Step 2 is $\frac{1}{2}$.

As seen partially in the description of $\mathcal{P}_{\text{Sti}}$, the possible operations used in card-based protocols (not just Stiglic's but others that have not been described thus far) are turning-over, rearrangement, and shuffling operations, which can be formalized as follows [10]. Below, we assume a sequence of d cards $\Gamma = (\alpha_1, \alpha_2, \dots, \alpha_d)$.

1. **Turning-over operation:** (turn, i). A turn operation involves turning over the i th card α_i , as shown in Fig. 1. The resulting sequence is

$$(\alpha_1, \dots, \alpha_{i-1}, \beta_i, \alpha_{i+1}, \dots, \alpha_d),$$

where β_i is obtained by turning over α_i .

Fig. 1 Turning-over operation

Fig. 2 Rearrangement operation

Fig. 3 Shuffling operation: The Hindu cut

2. **Rearrangement operation:** (perm, π). A perm operation involves the application of a permutation $\pi \in S_d$ (where S_d represents the symmetric group of degree d) to the sequence, as illustrated in Fig. 2. The resulting sequence is

$$(\alpha_{\pi^{-1}(1)}, \alpha_{\pi^{-1}(2)}, \dots, \alpha_{\pi^{-1}(d)}).$$

3. **Shuffling operation:** (shuffle, $\Pi, \mathcal{F}$). A shuffle operation involves the application of a permutation $\pi \in \Pi$ chosen from a permutation set $\Pi \subseteq S_d$ according to a probability distribution $\mathcal{F}$. See Fig. 3 again for an example of a shuffle. Note that a set Π along with a distribution $\mathcal{F}$ specifies a shuffle. We simply write (shuffle, Π) if $\mathcal{F}$ is uniform.

We define

$$\text{RC}^{\{i_1, i_2, \dots, i_\ell\}} \stackrel{\text{def}}{=} \{(i_1 i_2 \dots i_\ell)^j \mid 1 \leq j \leq \ell\},$$

for a cyclic permutation $(i_1 i_2 \dots i_\ell)$ such that $i_1 < i_2 < \dots < i_\ell$. Note that the random cut in Stiglic's protocol can be expressed as (shuffle, $\text{RC}^{\{1, 2, 3, 4, 5, 6, 7, 8\}}$).

3 New metrics and execution time of protocols

As mentioned in Sect. 2, turn, perm, and shuffle operations are used in card-based protocols. We need to take these operations into account to analyze the “execution time” of

Fig. 4 Add operation: adding two cards

protocols. In other words, the efficiency evaluation metrics shown in Table 1, i.e., the number of required cards, the number of colors, and the average number of trials, are insufficient to estimate the overall execution time.

In Sect. 3.1, we clarify all the operations that need to be considered. In Sect. 3.2, we count the number of occurrences of each operation for every AND protocol. In Sect. 3.3, we provide new metrics to estimate the execution time of protocols.

3.1 Operations to consider

In addition to the three kinds of operations, i.e., turn, perm, and shuffle, introduced in Sect. 2, we define another operation, named add. The add operation involves the addition of a card to the sequence with its face up (in order for players to be able to confirm the color), as shown in Fig. 4. When actually executing a protocol that requires additional cards, this add operation is necessary.

Therefore, altogether, the actual execution of a card-based protocol invokes four kinds of operations: add, turn, perm, and shuffle.

3.2 Analysis of the number of operations in each protocol

In this subsection, we analyze the number of operations in each of the seven existing AND protocols shown in Table 1. To this end, we use the *KWH-tree* [6] developed by Koch, Walzer, and Härtel, which is a diagram showing the state transition.

Let us take the KWH-tree of $\mathcal{P}_{\text{Sti}}$ shown in Fig. 5 as an example to explain the notation of the KWH-tree. Each box is associated with the “visible sequence trace” that means transitions of a sequence of cards visible from the table during the execution of the protocol. In each box, there are sequences of colors and the associated polynomials next to them. A polynomial represents the conditional probability that the current sequence is compatible with the associated sequence of colors (given the visible sequence trace), where X_{ij} denotes the probability that the input is (i, j) .

3.2.1 Stiglic’s protocol

We first analyze $\mathcal{P}_{\text{Sti}}$ in detail. The KWH-tree of $\mathcal{P}_{\text{Sti}}$ is shown in Fig. 5. This figure enables us to count all the operations appearing in $\mathcal{P}_{\text{Sti}}$, as follows:

1. **The number of add (adding a card) operations**

The number of add operations in $\mathcal{P}_{\text{Sti}}$ is four, because we add four cards to execute the protocol.

2. **The number of turn(turning over a card) operations**

Firstly, we execute the turn operation four times, because we need to turn over the four added cards after checking their colors. Secondly, we require the turn operation twice because of (turn, {1, 2}) after applying the first random cut. At this time, the probability that $\clubsuit\clubsuit$ or $\heartsuit\heartsuit$ appears and the protocol terminates is $\frac{1}{8} \times 2$. On the other hand, the probability that the protocol terminates by (turn, {3}) is $\frac{3}{8} \times \frac{1}{3} \times 2$. If the protocol does not terminate by (turn, {3}), we have to turn over the three face-up cards and execute (turn, {1, 2}) again after applying a random cut. Consequently, the expected number of turn operations in $\mathcal{P}_{\text{Sti}}$ is

$$4 + \sum_{n=1}^{\infty} \left((12n - 7) \times \frac{1}{4} \times (1/2)^{n-1} \right) = 12.5.$$

3. **The number of perm (rearranging a sequence of cards) operations**

We use no perm operation in $\mathcal{P}_{\text{Sti}}$, and hence, the number of utilizations of the perm operation is 0.

4. **The number of shuffle (shuffling a sequence of cards) operations**

As seen in the calculation for turn, the probability that $\mathcal{P}_{\text{Sti}}$ terminates by (turn, {1, 2}) is $\frac{1}{4}$. The probability that $\mathcal{P}_{\text{Sti}}$ terminates by (turn, {3}) is $\frac{1}{4}$, and the probability that $\mathcal{P}_{\text{Sti}}$ does not terminate and gets into a loop is $\frac{1}{2}$. Therefore, the expected number of shuffle operations is 2.

Thus, the numbers of add, turn, perm, and shuffle operations are 4, 12.5, 0, and 2, respectively. See the line of $\mathcal{P}_{\text{Sti}}$ in Table 2.

3.2.2 Crépeau and Kilian’s protocol

We also create the KWH-tree of $\mathcal{P}_{\text{CK}}$ (Crépeau and Kilian’s protocol [3]) as shown in Fig. 6, which tells us that the numbers of add, turn, perm, and shuffle operations are 6, 21, 1, and 8, respectively. See the line of $\mathcal{P}_{\text{CK}}$ in Table 2.

3.2.3 Niemi and Renvall’s protocol

The KWH-tree of $\mathcal{P}_{\text{NR}}$ is shown in Fig. 7, from which we can count the numbers of operations as shown in Table 2.

Fig. 5 $\mathcal{P}_{\text{Sti}}$'s KWH-tree

Table 2 The number of operations in the practical AND protocols

	#add	#turn	#perm	#shuffle
$\mathcal{P}_{\text{CK}}$ [3]	6	21	1	8
$\mathcal{P}_{\text{NR}}$ [15]	8	28	2.5	7.5
$\mathcal{P}_{\text{Sti}}$ [23]	4	12.5	0	2
$\mathcal{P}_{\text{MS}}$ [13]	2	4	2	1
$\mathcal{P}_{\text{AHMS}}$ [1]	1	17	6	7

3.2.4 Abe, Hayashi, Mizuki, and Sone's protocol

The KWH-tree of $\mathcal{P}_{\text{AHMS}}$ is shown in Fig. 8, from which we can count the numbers of operations as shown in Table 2.

3.2.5 The others

The KWH-tree of $\mathcal{P}_{\text{MS}}$ (Mizuki and Sone's protocol [13]) has been given in some existing literatures (e.g., [6,12]). Utilizing this KWH-tree, we are able to count each operation in $\mathcal{P}_{\text{MS}}$. Table 2 summarizes the result.

In addition, we conducted the same calculation for the two KWH protocols [6]. Table 3 shows the number of operations in the protocols. These protocols need shuffles which

have non-uniform probability distributions,³ and hence, they need special indistinguishable boxes or envelopes [16] to be implemented. Therefore, we have judged that these two protocols are more time-consuming than the other five protocols. Therefore, in the sequel, we focus on the five protocols in Table 2, which we call “practical” AND protocols.

3.3 Execution time of protocols

In this subsection, we present an expression for the execution time of each protocol based on four metrics. First, we denote the execution time of add, turn, perm, and shuffle by t_{add} , t_{turn} , t_{perm} , and t_{shuf} , respectively. In addition, $\text{Time}(\mathcal{P})$ denotes the overall execution time of a protocol $\mathcal{P}$. Then, the execution time of the protocols in Table 2 can be simply expressed as follows:

1. Crépeau & Kilian's protocol ($\mathcal{P}_{\text{CK}}$):
 $\text{Time}(\mathcal{P}_{\text{CK}}) = 6t_{\text{add}} + 21t_{\text{turn}} + t_{\text{perm}} + 8t_{\text{shuf}}$.
2. Niemi & Renvall's protocol ($\mathcal{P}_{\text{NR}}$):
 $\text{Time}(\mathcal{P}_{\text{NR}}) = 8t_{\text{add}} + 28t_{\text{turn}} + 2.5t_{\text{perm}} + 7.5t_{\text{shuf}}$.

³ Recently, improved protocols having uniform distributions were proposed [5,20], but they require “non-closed” shuffles, still implying implementation issues.

Fig. 6 $\mathcal{P}_{CK}$'s KWH-tree; this figure is rotated clockwise by 90° . Note that we use (six) perm* in this figure for avoiding the KWH-tree to be complicated; the original $\mathcal{P}_{CK}$ in [3] does not use them, and we do not count the number of them for Table 2

Fig. 7 $\mathcal{P}_{NR}$'s KWH-tree, where $X_0 = X_{00} + X_{01} + X_{10}$ and $X_1 = X_{11}$. The above left part makes two copied commitments to a , the above right part makes two copied commitments to b , and the bottom part produces a commitment to $a \wedge b$ from these four commitments

3. Stiglic's protocol ($\mathcal{P}_{Sti}$):
Time ($\mathcal{P}_{Sti}$) = $4t_{add} + 12.5t_{turn} + 2t_{shuf}$.
4. Mizuki & Sone's protocol ($\mathcal{P}_{MS}$):
Time ($\mathcal{P}_{MS}$) = $2t_{add} + 4t_{turn} + 2t_{perm} + t_{shuf}$.
5. Abe & Hayashi & Mizuki & Sone's protocol ($\mathcal{P}_{AHMS}$):
Time ($\mathcal{P}_{AHMS}$) = $t_{add} + 17t_{turn} + 6t_{perm} + 7t_{shuf}$.

In the next section, we make a comparison to determine the most efficient and practical protocol.

4 Comparison of the protocols

In this section, we evaluate the efficiency of the five practical AND protocols in Table 2 and discuss which protocol is the most efficient.

4.1 Efficiency comparison based on the execution time

In this subsection, we compare the execution times of the protocols.

First, we compare each coefficient of equation shown in Sect. 3.3 or Table 2. Obviously, we obtain the following inequalities:

$$\text{Time}(\mathcal{P}_{Sti}) < \text{Time}(\mathcal{P}_{CK}),$$

$$\text{Time}(\mathcal{P}_{Sti}) < \text{Time}(\mathcal{P}_{NR}).$$

Therefore, $\mathcal{P}_{Sti}$ is superior to $\mathcal{P}_{CK}$ and $\mathcal{P}_{NR}$.

Next, we compare $\mathcal{P}_{Sti}$ with $\mathcal{P}_{MS}$. At first glance, the coefficients might give us an impression that $\mathcal{P}_{MS}$ would be better than $\mathcal{P}_{Sti}$. However, we cannot immediately come to a con-

Fig. 8 $\mathcal{P}_{\text{AHMS}}$'s KWH-tree [1], where $X_0 = X_{00} + X_{01} + X_{10}$ and $X_1 = X_{11}$

Table 3 The number of operations in the KWH protocols [6]

	#add	#turn	#perm	#shuffle
Five-card KWH [6]	1	$\frac{11}{3}$	$\frac{7}{6}$	$\frac{14}{3}$
Four-card KWH [6]	0	7	2	8

clusion because $\text{Time}(\mathcal{P}_{\text{MS}})$ has $2t_{\text{perm}}$, while $\text{Time}(\mathcal{P}_{\text{Sti}})$ has no t_{perm} . Therefore, we actually measured the duration of each operation by manipulating real cards. For its experimental method and detailed result, see Appendix A.

As a result, our measurement provides us the following relationship:

$$t_{\text{add}} = t_{\text{turn}} \text{ and } 0.1t_{\text{perm}} < t_{\text{turn}}. \quad (3)$$

Moreover, it is reasonable to assume that

$$t_{\text{perm}} < t_{\text{shuf}},$$

because the shuffling operation generally takes more time than the rearrangement operation. From these assumptions, we have $\text{Time}(\mathcal{P}_{\text{MS}}) < \text{Time}(\mathcal{P}_{\text{Sti}})$.

Finally, we compare $\mathcal{P}_{\text{MS}}$ with $\mathcal{P}_{\text{AHMS}}$. Because we have assumed that $t_{\text{add}} = t_{\text{turn}}$ as the above, we have $\text{Time}(\mathcal{P}_{\text{MS}}) < \text{Time}(\mathcal{P}_{\text{AHMS}})$.

Fig. 9 The total execution time of each protocol for different shuffle times

As a result, we may conclude that $\mathcal{P}_{\text{MS}}$ is the protocol with the least execution time (as long as we admit the above assumptions).

One might think that the execution time of the shuffle used in $\mathcal{P}_{\text{Sti}}$ (i.e., a random cut) and the execution time of the shuffle used in $\mathcal{P}_{\text{MS}}$ and $\mathcal{P}_{\text{AHMS}}$ (i.e., a random bisection cut) are different. We note that the authors in [24] showed that a random bisection cut can be reduced to a random cut if the backsides of all cards are vertically asymmetric such as [?]. Therefore, we may assume that they are the same. Moreover, we measured the execution time of the secure implementation for a random bisection cut proposed in [24], which uses additional tools such as a ball and a rubber band; if we use this implementation, $\mathcal{P}_{\text{Sti}}$ is faster (see Sect. 4.3 for the details).

4.2 Impact of execution time of shuffling

In the previous subsection, we assumed that $t_{\text{perm}} < t_{\text{shuf}}$ holds. In this subsection, we further investigate how the difference between t_{perm} and t_{shuf} affects the overall execution time of a protocol. To this end, we regard t_{shuf} as a variable and other metrics t_{add} , t_{turn} , and t_{perm} as constants. Specifically, based on our measurement of the actual execution time, we fix⁴

$$t_{\text{add}} = t_{\text{turn}} = 0.8 \text{ (sec.)}, \quad t_{\text{perm}} = 7t_{\text{turn}}. \quad (4)$$

Then, we vary the value t_{shuf} from three seconds to sixty seconds; Fig. 9 shows the result. According to this figure, $\mathcal{P}_{\text{Sti}}$ and $\mathcal{P}_{\text{MS}}$ are considered to be more efficient.

⁴ This is an example assumption; we note that the time varies from person to person, of course.

4.3 Execution time of the secure implementation proposed in [24]

As shown in Table 2, we counted the numbers of the shuffles used in the practical AND protocols in the same way even if they are different. This is because a random bisection cut used in $\mathcal{P}_{\text{MS}}$ and $\mathcal{P}_{\text{AHMS}}$ can be reduced to the application of the Hindu cut to a sequence of cards if the backsides are vertically asymmetric, as mentioned before. On the other hand, the authors in [24] proposed a secure implementation of a random bisection cut using additional tools such as a ball and rubber band, which does not depend on a pattern of the backsides. Therefore, it is reasonable to consider the case where one uses such an implementation of a random bisection cut in $\mathcal{P}_{\text{MS}}$, which might affect our conclusion that $\mathcal{P}_{\text{MS}}$ is the fastest one. For this, we measured the execution time of it (denoted by t_{RBC}) by manipulating real tools that are exactly the same ones used in [24]. Its experimental method and result are shown in Appendix A.2.

As a result, we found that $t_{\text{RBC}} > 120$ sec. Because the authors in [25] showed that it suffices to apply the Hindu cut for 30 s, we derive the following relationship:

$$t_{\text{RBC}} > 4t_{\text{HC}},$$

where t_{HC} denotes the execution time of applying the Hindu cut. From this, we have $\text{Time}(\mathcal{P}_{\text{Sti}}) < \text{Time}(\mathcal{P}_{\text{MS}})$, and hence, we may conclude that $\mathcal{P}_{\text{Sti}}$ is the fastest one (on average) if one uses the above implementation [24].

5 Further investigation of rearrangement operations

In the previous section, we varied the value t_{shuf} , while t_{perm} is fixed. In this section, we discuss possible necessity of varying the value t_{perm} .

We now enumerate in Table 4 the perm operations used in the protocols shown in Table 2. Let us take a look at (perm, (2 4 3)) used in $\mathcal{P}_{\text{MS}}$ as an example. This rearrangement operation can be done by exchanging the second card and the portion consisting the third and fourth cards:

$$\begin{array}{|c|} \hline 2 \\ \hline \boxed{?} \\ \hline \end{array} \Leftrightarrow \begin{array}{|c|c|} \hline 3 & 4 \\ \hline \boxed{?} & \boxed{?} \\ \hline \end{array}.$$

As in this example, any cyclic permutation of the form $(i_1 i_2 \cdots i_\ell)^j$ for some j , $1 \leq j \leq \ell - 1$ such that $i_1 < i_2 < \cdots < i_\ell$ can be done by exchanging two “portions” of cards. Therefore, we can assume that the execution times of the permutations shown in Table 4 except for the ones used in $\mathcal{P}_{\text{CK}}$ and $\mathcal{P}_{\text{NR}}$ are the same.

Table 4 Permutations used in the practical AND protocols, where π_1 and π_2 are given in Eq. 5

	Permutations
$\mathcal{P}_{\text{CK}}$ [3]	(2 3 6 4 7 5)
$\mathcal{P}_{\text{NR}}$ [15]	(9 10), π_1 , π_2 , and similar ones
$\mathcal{P}_{\text{Sti}}$ [23]	no permutation
$\mathcal{P}_{\text{MS}}$ [13]	(2 4 3), (2 3 4)
$\mathcal{P}_{\text{AHMS}}$ [1]	(1 2), (2 3), (2 3 4 5)

In contrast, (perm, (2 3 6 4 7 5)) used in $\mathcal{P}_{\text{CK}}$ and

$$\begin{aligned} \pi_1 &= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 8 & 7 & 9 & 10 & 2 & 1 & 4 & 3 & 5 & 6 \end{pmatrix}, \\ \pi_2 &= \begin{pmatrix} 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 6 & 7 & 3 & 11 & 8 & 12 & 5 & 4 & 10 & 9 \end{pmatrix} \end{aligned} \quad (5)$$

(and similar ones) used in $\mathcal{P}_{\text{NR}}$ are not such a form. First, (perm, (2 3 6 4 7 5)) can be illustrated as follows:

To perform this, we move the second card to the third position and the fifth card to the second position and then exchange the portions consisting the third and fourth cards with the portion consisting the sixth and seventh cards. We actually measured the execution time of this move by manipulating real cards. As a result, we found that (perm, (2 3 6 4 7 5)) takes approximately four times as long as just exchanging two portions.

Remember that π_1 is one of the six permutations appeared in the left part of $\mathcal{P}_{\text{NR}}$'s KWH-tree in Fig. 7. These six permutations are similar: They move the first to fourth cards to the seventh to tenth, the fifth to eighth cards to the first to fourth, and the ninth and tenth cards to the fifth and sixth, respectively. Among the six permutations, (perm, π_1) should take the longest time. We also actually measured the execution time of this move. As a result, we found that it takes approximately nine to ten times as long as just exchanging two portions.

Finally, π_2 is one of the six permutations appeared in the right part of $\mathcal{P}_{\text{NR}}$'s KWH-tree. These six permutations are also similar: They move the third and fourth cards to the sixth and seventh, the ninth to tenth cards to the fourth and fifth, and the eleventh to twelfth cards to the ninth to tenth, respectively. They also move the revealed black cards to the third and eighth and the revealed red cards to the eleventh and the twelfth. We also actually measured the execution time of (perm, π_2), which should take the longest time among the

six permutations. As a result, we found that it takes approximately eleven to twelve times as long as just exchanging two portions.

Therefore, Time ($\mathcal{P}_{CK}$) and Time ($\mathcal{P}_{NR}$) could be larger than those in Sect. 3.3. However, even if so, the statement that $\mathcal{P}_{MS}$ is the protocol with the least execution time (which we concluded in Sect. 4.1) does hold.

The detailed results for measuring the execution time of applying the four permutations can be seen in Appendix A.3.

6 Conclusion

The widely-used efficiency evaluation metrics of card-based protocols do not capture the number of operations fully, and hence, it is difficult to estimate their execution time accurately. Therefore, we considered all kinds of possible operations so that we have four metrics, and focused on counting the number of operations comprehensively to estimate the execution time of protocols. Our new criteria allow us to evaluate the efficiency of protocols. Thus, we were able to compare the execution time of the protocols. Some reasonable assumptions concluded that the Mizuki–Sone AND protocol [13] is the most efficient and practical as an AND protocol in terms of the execution time.

To count the number of operations, we created KWH-trees for $\mathcal{P}_{CK}$ and $\mathcal{P}_{Sti}$ as shown in Figs. 5 and 6, respectively. This is the first attempt to describe KWH-trees for these previous protocols, and we believe that Figs. 5 and 6 themselves form one of the major contributions of this paper.

Intriguing future work involves applying our new criteria to other existing protocols using different types of cards (e.g., [21,22]) or those using private operations (e.g., [14,17–19]).

Acknowledgements We thank the anonymous referees, whose comments have helped us to improve the presentation of the paper.

Funding This work was supported in part by JSPS KAKENHI, Grant Number JP17K00001 and JP19J21153.

Compliance with ethical standards

Conflicts of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any new studies with human participants or animals performed by any of the authors.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material

Fig. 10 The tools used in our experiment for measuring the execution time of the secure implementation proposed in [24]; the sequence of six cards, separator, rubber band, curving polystyrene foam ball, tape, and scissors

is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

A Experimental results

In this appendix, we show our experimental methods and detailed results. A deck of cards used in our experiments is the one shown in Figs. 1 to 4, which was made by the fourth author.

A.1 Add, turn, and perm

We show how to derive the relationship (3) and (4) shown in Sects. 4.1 and 4.2. This measurement was mainly conducted by the second author, who is familiar with playing cards.

Method. To measure t_{add} and t_{turn} , we measured the execution time of adding and turning over ten cards five times, respectively. To measure t_{perm} , we measured the execution time of applying all the permutations appeared in $\mathcal{P}_{CK}$, $\mathcal{P}_{NR}$, $\mathcal{P}_{Sti}$, and $\mathcal{P}_{MS}$ (namely six permutations) five times.

Result. Table 5 summarizes the experimental result; $t_{\text{add}} = 0.655$, $t_{\text{turn}} = 0.803$, and $t_{\text{perm}} = 5.889$. Since the difference between t_{add} and t_{turn} is relatively small (about 0.15 seconds), we have fixed $t_{\text{add}} = t_{\text{turn}} = 0.8$ as in (4).

A.2 Secure implementation using a ball and rubber Band

We show how to derive $t_{\text{RBC}} > 120$ sec shown in Sect. 4.3. This evaluation was mainly conducted by the first author, who is an inventor of this implementation.

Table 5 The experimental result of measuring the execution time of adding, turning over, and permuting cards, which was mainly conducted by the second author, who is familiar with playing cards. Since we measured the execution time of adding and turning over ten cards for each

	1st (sec)	2nd (sec)	3rd (sec)	4th (sec)	5th (sec)	avg. (sec)	Avg. (sec per card)
Add	6.8	6.4	6.1	5.8	7.7	6.6	0.66
Turn	8.7	8.2	8.1	7.1	8.1	8.0	0.80
Perm	37.0	38.3	35.7	34.2	31.5	35.3	5.89

time, the values in the rightmost column are the ones in the left of them divided by ten, respectively. For perm, since we measured the execution time of applying six (different) permutations, the value in the rightmost column is the one in the left of it divided by six

Table 6 The experimental result of measuring the execution time of the secure implementation proposed in [24]

	1st (sec)	2nd (sec)	3rd (sec)	4th (sec)	5th (sec)	Avg. (sec)
t_{RBC}	159.2	138.1	138.2	124.8	130.8	138.2

Table 7 The experimental result of measuring the four permutations

	1st (sec)	2nd (sec)	3rd (sec)	4th (sec)	5th (sec)	Avg. (sec)
(perm, (2 4 3))	2.2	2.4	2.3	2.6	2.1	2.3
(perm, (2 3 6 4 7 5))	10.2	9.1	10.7	9.5	9.1	9.7
(perm, π_1)	23.2	19.9	23.3	21.6	21.6	21.9
(perm, π_2)	32.7	25.9	24.4	26.6	26.8	27.3

Method. To measure t_{RBC} , we prepared exactly the same tools in [24] as shown in Fig. 10. We measured the execution time of the following procedure five times:

1. Band the two halves of the sequence of six cards using the rubber band and separator.
2. Place the banded pile of cards into one half of the curving ball.
3. On the half, put the other half and tape them together using the tape and scissors.
4. Throw the ball in the air in a spinning manner.
5. Remove the tape and rubber band and place the resulting sequence in its original position.

Result. Table 6 summarizes the experimental result. Because it took more than 120 seconds at every time, we have $t_{RBC} > 120$ sec.

A.3 Several kinds of permutations

We show experimental method and result for measuring the execution time of the four permutations shown in Sect. 5. This evaluation was mainly conducted by the first author. We measured the execution time of applying each permutation five times. Table 7 summarizes the experimental result.

References

1. Abe, Y., Hayashi, Y., Mizuki, T., Sone, H.: Five-card AND protocol in committed format using only practical shuffles. In: Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop. pp. 3–8. APKC '18, ACM, New York, NY, USA (2018)
2. den Boer, B.: More efficient match-making and satisfiability the five card trick. In: Quisquater, J.J., Vandewalle, J. (eds.) Advances in Cryptology – EUROCRYPT '89. Lecture Notes in Computer Science, vol. 434, pp. 208–217. Springer, Berlin (1990)
3. Crépeau, C., Kilian, J.: Discreet solitary games. In: Stinson, D.R. (ed.) Advances in Cryptology – CRYPTO' 93. Lecture Notes in Computer Science, vol. 773, pp. 319–330. Springer, Berlin (1994)
4. Ishikawa, R., Chida, E., Mizuki, T.: Efficient card-based protocols for generating a hidden random permutation without fixed points. In: Calude, C.S., Dinneen, M.J. (eds.) Unconventional Computation and Natural Computation. Lecture Notes in Computer Science, vol. 9252, pp. 215–226. Springer, Cham (2015)
5. Koch, A.: The landscape of optimal card-based protocols. Cryptology ePrint Archive, Report 2018/951 (2018). <https://eprint.iacr.org/2018/951>
6. Koch, A., Walzer, S., Härtel, K.: Card-based cryptographic protocols using a minimal number of cards. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology—ASIACRYPT 2015. Lecture Notes in Computer Science, vol. 9452, pp. 783–807. Springer, Berlin (2015)
7. Miyahara, D., Ueda, I., Hayashi, Y., Mizuki, T., Sone, H.: Analyzing execution time of card-based protocols. In: Stepney, S., Verlan, S. (eds.) Unconventional Computation and Natural Computation. Lecture Notes in Computer Science, vol. 10867, pp. 145–158. Springer, Cham (2018)
8. Mizuki, T., Asiedu, I.K., Sone, H.: Voting with a logarithmic number of cards. In: Mauri, G., Dennunzio, A., Manzoni, L., Porreca, A.E. (eds.) Unconventional Computation and Natural Computation

- tion. *Lecture Notes in Computer Science*, vol. 7956, pp. 162–173. Springer, Berlin (2013)
9. Mizuki, T., Kumamoto, M., Sone, H.: The five-card trick can be done with four cards. In: Wang, X., Sako, K. (eds.) *Advances in Cryptology—ASIACRYPT 2012. Lecture Notes in Computer Science*, vol. 7658, pp. 598–606. Springer, Berlin (2012)
10. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. *Int. J. Inf. Secur.* **13**(1), 15–23 (2014)
11. Mizuki, T., Shizuya, H.: Practical card-based cryptography. In: Ferro, A., Luccio, F., Widmayer, P. (eds.) *Fun with Algorithms. Lecture Notes in Computer Science*, vol. 8496, pp. 313–324. Springer, Cham (2014)
12. Mizuki, T., Shizuya, H.: Computational model of card-based cryptographic protocols and its applications. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* E100.A(1), 3–11 (2017)
13. Mizuki, T., Sone, H.: Six-card secure AND and four-card secure XOR. In: Deng, X., Hopcroft, J.E., Xue, J. (eds.) *Frontiers in Algorithmics. Lecture Notes in Computer Science*, vol. 5598, pp. 358–369. Springer, Berlin (2009)
14. Nakai, T., Shirouchi, S., Iwamoto, M., Ohta, K.: Four cards are sufficient for a card-based three-input voting protocol utilizing private permutations. In: Shikata, J. (ed.) *Information Theoretic Security. Lecture Notes in Computer Science*, vol. 10681, pp. 153–165. Springer, Cham (2017)
15. Niemi, V., Renvall, A.: Secure multiparty computations without computers. *Theor. Comput. Sci.* **191**(1–2), 173–183 (1998)
16. Nishimura, A., Nishida, T., Hayashi, Y., Mizuki, T., Sone, H.: Card-based protocols using unequal division shuffles. *Soft Comput.* (2017). <https://doi.org/10.1007/s00500-017-2858-2>
17. Ono, H., Manabe, Y.: Efficient card-based cryptographic protocols for the millionaires’ problem using private input operations. In: 2018 13th Asia Joint Conference on Information Security (AsiaJ-CIS), pp. 23–28 (2018)
18. Ono, H., Manabe, Y.: Card-based cryptographic protocols with the minimum number of cards using private operations. In: Zircir-Heywood, N., Bonfante, G., Debbabi, M., Garcia-Alfaro, J. (eds.) *Foundations and Practice of Security. Lecture Notes in Computer Science*, vol. 11358, pp. 193–207. Springer, Cham (2019)
19. Ono, H., Manabe, Y.: Card-based cryptographic protocols with the minimum number of rounds using private operations. In: Pérez-Solà, C., Navarro-Arribas, G., Biryukov, A., Garcia-Alfaro, J. (eds.) *Data Privacy Management, Cryptocurrencies and Blockchain Technology. Lecture Notes in Computer Science*, vol. 11737, pp. 156–173. Springer, Cham (2019)
20. Ruangwises, S., Itoh, T.: And protocols using only uniform shuffles. In: van Bevern, R., Kucherov, G. (eds.) *Computer Science—Theory and Applications. Lecture Notes in Computer Science*, vol. 11532, pp. 349–358. Springer, Cham (2019)
21. Shinagawa, K., Mizuki, T., Schuldt, J.C.N., Nuida, K., Kanayama, N., Nishide, T., Hanaoka, G., Okamoto, E.: Secure computation protocols using polarizing cards. *IEICE Trans.* **99–A**(6), 1122–1131 (2016)
22. Shinagawa, K., Mizuki, T., Schuldt, J.C., Nuida, K., Kanayama, N., Nishide, T., Hanaoka, G., Okamoto, E.: Card-based protocols using regular polygon cards. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **E100.A**(9), 1900–1909 (2017)
23. Stiglic, A.: Computations with a deck of cards. *Theor. Comput. Sci.* **259**(1–2), 671–678 (2001)
24. Ueda, I., Miyahara, D., Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: Secure implementations of a random bisection cut. *Int. J. Inf. Secur.* **19**(4), 445–452 (2020). <https://doi.org/10.1007/s10207-019-00463-w>
25. Ueda, I., Nishimura, A., Hayashi, Y., Mizuki, T., Sone, H.: How to implement a random bisection cut. In: Martín-Vide, C., Mizuki, T., Vega-Rodríguez, M.A. (eds.) *Theory and Practice of Natural Computing. Lecture Notes in Computer Science*, vol. 10071, pp. 58–69. Springer, Cham (2016)

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