Analyzing Execution Time of Card-Based Protocols*

Daiki Miyahara[1], Itaru Ueda[1], Yu-ichi Hayashi[2],
Takaaki Mizuki[3], and Hideaki Sone[3]

[1] Graduate School of Information Sciences, Tohoku University
6–3–09 Aramaki-Aza-Aoba, Aoba, Sendai 980–8579, Japan
daiki.miyahara.q4@dc.tohoku.ac.jp
[2] Graduate School of Information Science, Nara Institute of Science and Technology
8916–5 Takayama, Ikoma, Nara 630–0192, Japan
[3] Cyberscience Center, Tohoku University
6–3 Aramaki-Aza-Aoba, Aoba, Sendai 980–8578, Japan
tm-paper+cardtime@g-mail.tohoku-university.jp

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 multi-party computations

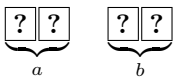
1 Introduction

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, 6, 13]). To illustrate this, let us explain how to manipulate Boolean values based on a two-colored deck of cards. Given a black card $\boxed{\clubsuit}$ and a red card $\boxed{\heartsuit}$, a Boolean value can be expressed as:

$$\boxed{\clubsuit}\boxed{\heartsuit} = 0\,, \quad \boxed{\heartsuit}\boxed{\clubsuit} = 1\,.$$

* This paper appears in Proceedings of UCNC 2018. The final publication is available at Springer via https://doi.org/10.1007/978-3-319-92435-9_11.

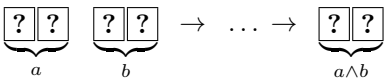
Following this encoding, for example, two players, Alice and Bob, can each put two cards face down on a table representing their private bits a and b, respectively:



(1)

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 (1) 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 (1), 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:



As shown in Table 1, there are many existing AND protocols (in committed format[i]). 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, and then analyze the execution time of each operation. Furthermore, we actually evaluate all the AND protocols[ii] shown in Table 1, based on our new criteria by counting the number of operations thoroughly. We also make a comparison of the AND protocols and discuss which protocol is the most efficient and practical. It should be noted that card-based protocols are outside the Turing model [8, 9].

The rest of this paper is organized as follows. In Section 2, we introduce the AND protocol invented by Stiglic [15] as an example, and then give a formalization of the operations in card-based protocols [8]. In Section 3, we give new metrics of efficiency, which directly indicate the execution time of a protocol. In Section 4, we evaluate the existing AND protocols based on our proposed metrics. We conclude this study in Section 5.

[i] There are also "non-committed-format" AND protocols [1, 7].

[ii] This paper addresses only AND computation because the other important primitive, XOR, can be done with only four cards and one trial [10].

Table 1: The existing AND protocols (in committed format)

	Year	**#Colors**	**#Cards**	**Avg.#Trials**
Crépeau & Kilian [2]	1993	4	10	6
Niemi & Renvall [11]	1998	2	12	2.5
Stiglic [15]	2001	2	8	2
Mizuki & Sone [10]	2009	2	6	1
Five-card KWH [5]	2015	2	5	1
Four-card KWH [5]	2015	2	4	3

2 Preliminaries: A Protocol with Operations

In this section, we introduce Stiglic's AND protocol [15] as an example to demonstrate the possible operations in card-based protocols. As already seen in Table 1, this 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:

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

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:

$$\overset{1}{?}\overset{2}{?}\overset{3}{?}\overset{4}{?}\overset{5}{?}\overset{6}{?}\overset{7}{?}\overset{8}{?},$$

then a random cut results in one of the following eight sequences (with a probability of 1/8):

$$\overset{1}{?}\overset{2}{?}\overset{3}{?}\overset{4}{?}\overset{5}{?}\overset{6}{?}\overset{7}{?}\overset{8}{?},$$

$$\overset{2}{?}\overset{3}{?}\overset{4}{?}\overset{5}{?}\overset{6}{?}\overset{7}{?}\overset{8}{?}\overset{1}{?},$$

$$\vdots$$

$$\overset{8}{?}\overset{1}{?}\overset{2}{?}\overset{3}{?}\overset{4}{?}\overset{5}{?}\overset{6}{?}\overset{7}{?}.$$

Note that a random cut is known to be easily implemented by humans securely via the Hindu cut [16] (as shown in Figure 1).

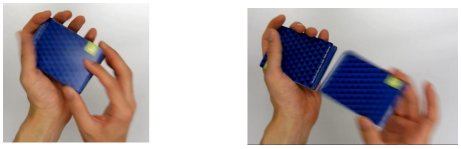


Fig. 1: The Hindu cut

3. Turn over the first two cards (from the left).
   (a) If the revealed cards are $\boxed{\heartsuit}\boxed{\heartsuit}$, we obtain a commitment to $a \wedge b$ as follows:
   $$\boxed{\heartsuit}\boxed{\heartsuit}\boxed{?}\boxed{?}\boxed{?}\underbrace{\boxed{?}\boxed{?}}_{a\wedge b}\boxed{?}.$$
   (b) If the revealed cards are $\boxed{\clubsuit}\boxed{\clubsuit}$, we obtain
   $$\boxed{\clubsuit}\boxed{\clubsuit}\boxed{?}\underbrace{\boxed{?}\boxed{?}}_{a\wedge b}\boxed{?}\boxed{?}\boxed{?}.$$
   (c) If the revealed cards are $\boxed{\clubsuit}\boxed{\heartsuit}$ or $\boxed{\heartsuit}\boxed{\clubsuit}$, turn over the third card.
      i. If the three face-up cards are $\boxed{\heartsuit}\boxed{\clubsuit}\boxed{\clubsuit}$, we have
      $$\boxed{\heartsuit}\boxed{\clubsuit}\boxed{\clubsuit}\boxed{?}\underbrace{\boxed{?}\boxed{?}}_{a\wedge b}\boxed{?}\boxed{?}.$$
      ii. If the three face-up cards are $\boxed{\clubsuit}\boxed{\heartsuit}\boxed{\heartsuit}$, we have
      $$\boxed{\clubsuit}\boxed{\heartsuit}\boxed{\heartsuit}\boxed{?}\boxed{?}\boxed{?}\underbrace{\boxed{?}\boxed{?}}_{a\wedge b}.$$
      iii. If the three face-up cards are $\boxed{\clubsuit}\boxed{\heartsuit}\boxed{\clubsuit}$ or $\boxed{\heartsuit}\boxed{\clubsuit}\boxed{\heartsuit}$, turn them over and go back to Step 2.

This is Stiglic's AND protocol, which we denote by $\mathcal{P}_{\mathrm{Sti}}$ hereinafter. A shuffling operation called a random cut is used in Step 2 of $\mathcal{P}_{\mathrm{Sti}}$. The average number of trials is two, because the probability that Step 3–(c)–iii occurs and we go back to Step 2 is 1/2. As seen partially in the description of $\mathcal{P}_{\mathrm{Sti}}$, 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 [8]. Below, we assume a sequence of d cards $\Gamma = (\alpha_1, \alpha_2, \ldots, \alpha_d)$.

1. **Turning-over operation:** (turn, i).
   A turn operation involves turning over the i-th card α_i, as shown in Figure 2. The resulting sequence is $(\alpha_1, \ldots, \alpha_{i-1}, \beta_i, \alpha_{i+1}, \ldots, \alpha_d)$, where β_i is obtained by turning over α_i.

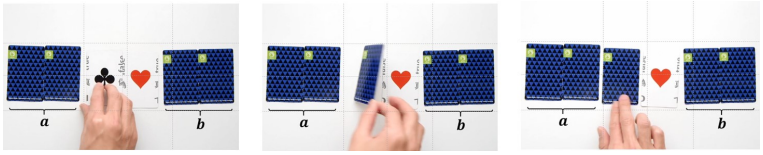


Fig. 2: Turning-over operation

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 Figure 3. The resulting sequence is $(\alpha_{\pi^{-1}(1)}, \alpha_{\pi^{-1}(2)}, \ldots, \alpha_{\pi^{-1}(d)})$.

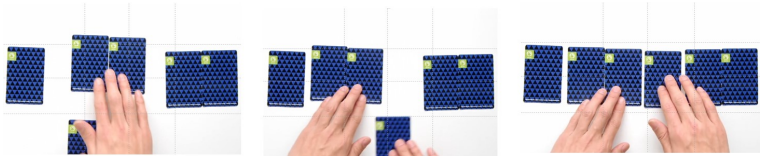


Fig. 3: Rearrangement operation

3. **Shuffling operation:** (shuffle, Π, $\mathcal{F}$).
   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 $\mathcal{F}$, as shown in Figure 4. Note that a set Π along with a distribution $\mathcal{F}$ specifies a shuffle.

Fig. 4: Shuffling operation

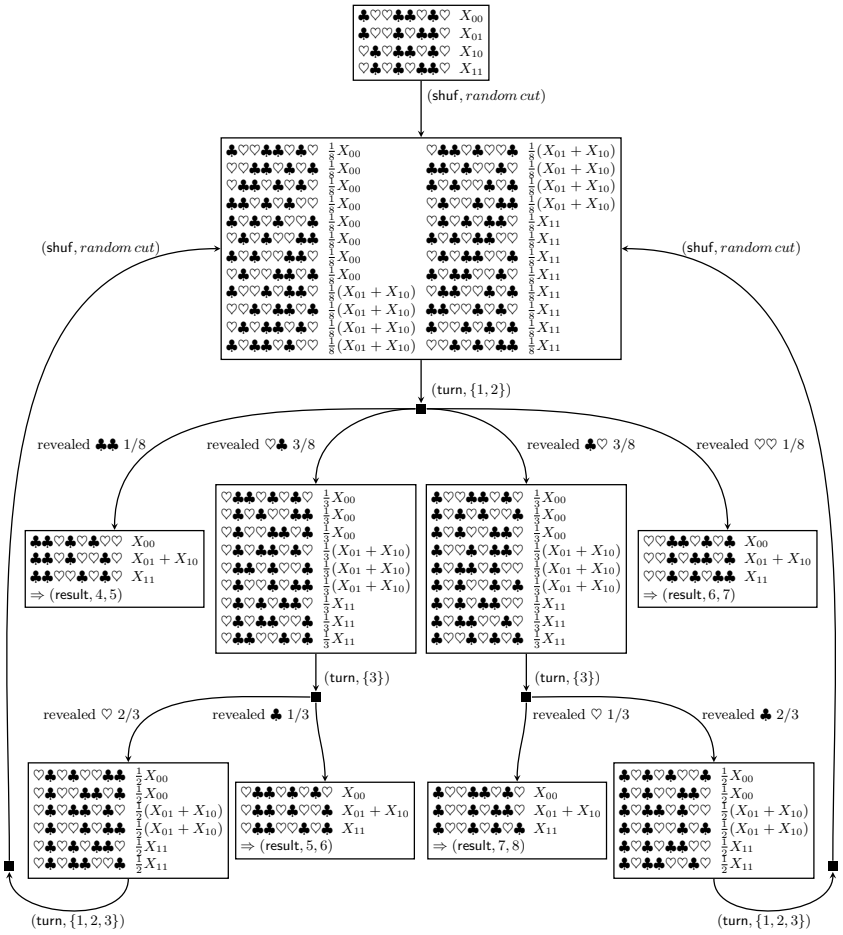


Fig. 5: $\mathcal{P}_{\mathrm{Sti}}$'s KWH-tree

3 New Metrics and Execution Time of Protocols

As mentioned in Section 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 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 Section 3.1, we clarify all the operations that need to be considered. In Section 3.2, we count the number of occurrences of each operation for every AND protocol. In Section 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 Section 2, we define another operation, named place. The place 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 Figure 8. When actually executing a protocol that requires additional cards, this place operation is necessary.

Fig. 8: Place operation: Adding two cards

Therefore, altogether, the actual execution of a card-based protocol invokes four kinds of operations: place, 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 six existing AND protocols shown in Table 1. To this end, we use the *KWH-tree* [5] developed by Koch, Walzer, and Härtel, which is a diagram showing the state transition.

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

1. **The number of place (adding a card) operations in $\mathcal{P}_{\mathrm{Sti}}$.**
   The number of place operations in $\mathcal{P}_{\mathrm{Sti}}$ is four, because we add four cards to execute the protocol.
2. **The number of turn (turning over a card) operations in $\mathcal{P}_{\mathrm{Sti}}$.**
   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 $(\mathsf{turn}, \{1, 2\})$ after applying the first random cut. At this time, the probability that ♣♣ or ♡♡ appears and the protocol terminates is $\frac{1}{8} \times 2$. On the other hand, the probability that the protocol terminates by $(\mathsf{turn}, \{3\})$ is $\frac{3}{8} \times \frac{1}{3} \times 2$. If the protocol does not terminate by $(\mathsf{turn}, \{3\})$, we have to turn over the three face-up cards and execute $(\mathsf{turn}, \{1, 2\})$ again after applying a random cut. Consequently, the expected number of turn operations in $\mathcal{P}_{\mathrm{Sti}}$ is

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

3. **The number of** perm **(rearranging a sequence of cards) operations in** $\mathcal{P}_{\mathrm{Sti}}$**.**
   We use no perm operation in $\mathcal{P}_{\mathrm{Sti}}$, and hence the number of utilizations of the perm operation is 0.

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

$$\sum_{n=1}^{\infty} \left\{ n \times \frac{1}{2} \times \left(\frac{1}{2}\right)^{n-1} \right\} = 2.$$

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

Similarly, we also create the KWH-trees of $\mathcal{P}_{\mathrm{CK}}$ (Crépeau and Kilian's protocol [2]) and $\mathcal{P}_{\mathrm{NR}}$ (Niemi and Renvall's protocol [11]), as shown in Figures 6 and 7, respectively; the KWH-tree of $\mathcal{P}_{\mathrm{MS}}$ (Mizuki and Sone's protocol [10]) has been given in some existing literatures (e.g. [9]). Utilizing these KWH-trees, we are able to count each operation in $\mathcal{P}_{\mathrm{CK}}$, $\mathcal{P}_{\mathrm{NR}}$, and $\mathcal{P}_{\mathrm{MS}}$. Table 2 summarizes the results.

In addition, we conducted the same calculation for the two KWH protocols [5]. Table 3 shows the number of operations in the protocols. These protocols need shuffles which have non-uniform probability distributions, and hence, they need special indistinguishable boxes or envelopes [12] to be implemented. Therefore, we have judged that these two protocols are more time-consuming than the other four protocols. Therefore, in the sequel, we focus on the four protocols in Table 2, which we call "practical" AND protocols.

Table 2: The number of operations in the practical AND protocols

	#place	#turn	#perm	#shuffle
$\mathcal{P}_{\mathrm{CK}}$ [2]	6	21	1	8
$\mathcal{P}_{\mathrm{NR}}$ [11]	8	28	4.5	7.5
$\mathcal{P}_{\mathrm{Sti}}$ [15]	4	12.5	0	2
$\mathcal{P}_{\mathrm{MS}}$ [10]	2	4	2	1

Table 3: The number of operations in the KWH protocols [5]

	#place	#turn	#perm	#shuffle
Five-card KWH [5]	1	11/3	7/6	14/3
Four-card KWH [5]	0	7	2	8

3.3 Execution Time of Protocols

Here, we present an expression for the execution time of each protocol based on four metrics. First, we denote the execution time of place, turn, perm, and shuffle by t_{place}, t_{turn}, t_{perm}, and t_{shuf}, respectively. In addition, 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 easily expressed as follows.

1. Crépeau & Kilian's protocol ($\mathcal{P}_{\text{CK}}$).
   $\text{Time}(\mathcal{P}_{\text{CK}}) = 6t_{\text{place}} + 21t_{\text{turn}} + t_{\text{perm}} + 8t_{\text{shuf}}$.
2. Niemi & Renvall's protocol ($\mathcal{P}_{\text{NR}}$).
   $\text{Time}(\mathcal{P}_{\text{NR}}) = 8t_{\text{place}} + 28t_{\text{turn}} + 4.5t_{\text{perm}} + 7.5t_{\text{shuf}}$.
3. Stiglic's protocol ($\mathcal{P}_{\text{Sti}}$).
   $\text{Time}(\mathcal{P}_{\text{Sti}}) = 4t_{\text{place}} + 12.5t_{\text{turn}} + 2t_{\text{shuf}}$.
4. Mizuki & Sone's protocol ($\mathcal{P}_{\text{MS}}$).
   $\text{Time}(\mathcal{P}_{\text{MS}}) = 2t_{\text{place}} + 4t_{\text{turn}} + 2t_{\text{perm}} + t_{\text{shuf}}$.

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 four 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 Section 3.3 or Table 2. Obviously, we obtain the following inequalities:

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

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

Therefore, $\mathcal{P}_{\text{Sti}}$ is superior to $\mathcal{P}_{\text{CK}}$ and $\mathcal{P}_{\text{NR}}$. Hence, it suffices to compare $\mathcal{P}_{\text{Sti}}$ with $\mathcal{P}_{\text{MS}}$.

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

Time($\mathcal{P}_{\mathrm{MS}}$) has $2t_{\mathrm{perm}}$ while Time($\mathcal{P}_{\mathrm{Sti}}$) has no t_{perm}. Therefore, we actually measured the duration of each operation by manipulating real cards. As a result, our measurement provides us the following relationship:

$$t_{\mathrm{place}} = t_{\mathrm{turn}} \text{ and } 0.1t_{\mathrm{perm}} < t_{\mathrm{turn}}.$$

Moreover, it is reasonable to assume that

$$t_{\mathrm{perm}} < t_{\mathrm{shuf}}$$

because the shuffling operation generally takes more time than the rearrangement operation. From these findings, we have

$$\begin{aligned}\mathrm{Time}(\mathcal{P}_{\mathrm{MS}}) &= 2t_{\mathrm{place}} + 4t_{\mathrm{turn}} + 2t_{\mathrm{perm}} + t_{\mathrm{shuf}} \\ &< 2t_{\mathrm{place}} + 14t_{\mathrm{turn}} + t_{\mathrm{perm}} + t_{\mathrm{shuf}} \\ &< 4t_{\mathrm{place}} + 12.5t_{\mathrm{turn}} + 2t_{\mathrm{shuf}} = \mathrm{Time}(\mathcal{P}_{\mathrm{Sti}}).\end{aligned}$$

Therefore, we have Time($\mathcal{P}_{\mathrm{MS}}$) < Time($\mathcal{P}_{\mathrm{Sti}}$). This implies that $\mathcal{P}_{\mathrm{MS}}$ is the protocol with the least execution time.

4.2 Impact of The Execution Time of Shuffling

In the previous subsection, we assumed that $t_{\mathrm{perm}} < t_{\mathrm{shuf}}$ 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_{place}, t_{turn}, and t_{perm} as constants. Specifically, based on our measurement of the actual execution time, we fix

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

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

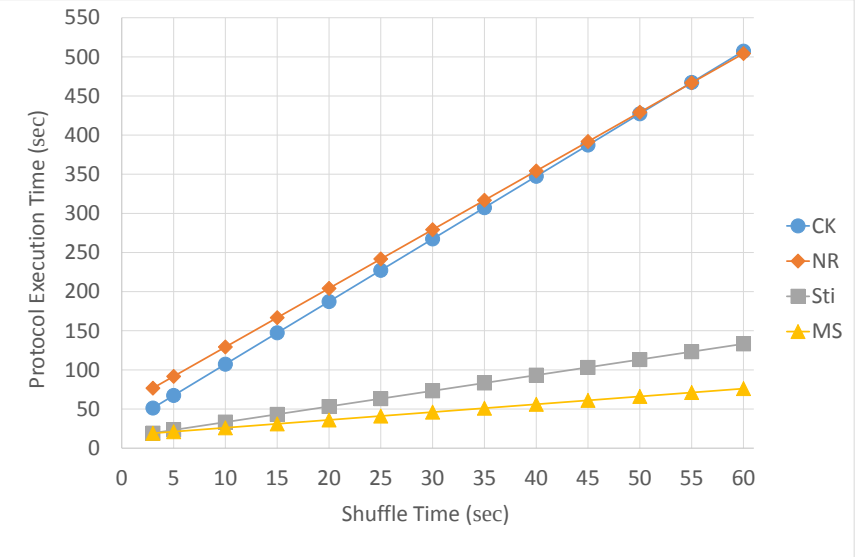


Fig. 9: The total execution time of each protocol for different shuffle times

5 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 allows us to evaluate the efficiency of protocols. Thus, we were able to compare the execution time of the protocols. We concluded that the Mizuki–Sone AND protocol [10] 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 $\mathcal{P}_{\mathrm{CK}}$, $\mathcal{P}_{\mathrm{NR}}$, and $\mathcal{P}_{\mathrm{Sti}}$, as shown in Figures 7, 6, and 5, respectively. This is the first attempt to describe KWH-trees for these previous protocols, and we believe that Figures 7, 6, and 5 themselves form one of the major contributions of this paper.

Our future work involves (i) applying our new criteria to the other existing protocols (e.g. [3, 14]) and (ii) clarifying the variables that affect the execution time of a shuffle (e.g., the number of cards) and other operations.

Acknowledgments

We thank the anonymous referees, whose comments have helped us to improve the presentation of the paper. This work was supported by JSPS KAKENHI Grant Number JP17K00001.

References

1. 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, Heidelberg (1990)
2. 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, Heidelberg (1994)
3. Hashimoto, Y., Shinagawa, K., Nuida, K., Inamura, M., Hanaoka, G.: Secure grouping protocol using a deck of cards. In: Shikata, J. (ed.) Information Theoretic Security. Lecture Notes in Computer Science, vol. 10681, pp. 135–152. Springer, Cham (2017)
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., 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, Heidelberg (2015)
6. 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. Lecture Notes in Computer Science, vol. 7956, pp. 162–173. Springer, Berlin, Heidelberg (2013)
7. 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, Heidelberg (2012)
8. Mizuki, T., Shizuya, H.: A formalization of card-based cryptographic protocols via abstract machine. International Journal of Information Security 13(1), 15–23 (2014)
9. 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)
10. 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, Heidelberg (2009)
11. Niemi, V., Renvall, A.: Secure multiparty computations without computers. Theoretical Computer Science 191(1–2), 173–183 (1998)
12. Nishimura, A., Hayashi, Y.i., Mizuki, T., Sone, H.: An implementation of non-uniform shuffle for secure multi-party computation. In: Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography. pp. 49–55. AsiaPKC '16, ACM, New York, NY, USA (2016), http://doi.acm.org/10.1145/2898420.2898425

13. Nishimura, A., Nishida, T., Hayashi, Y.i., Mizuki, T., Sone, H.: Card-based protocols using unequal division shuffles. Soft Computing (Oct 2017), https://doi.org/10.1007/s00500-017-2858-2
14. Shinagawa, K., Mizuki, T., Schuldt, J.C.N., Nuida, K., Kanayama, N., Nishide, T., Hanaoka, G., Okamoto, E.: Multi-party computation with small shuffle complexity using regular polygon cards. In: Au, M.H., Miyaji, A. (eds.) Provable Security. Lecture Notes in Computer Science, vol. 9451, pp. 127–146. Springer, Cham (2015)
15. Stiglic, A.: Computations with a deck of cards. Theoretical Computer Science 259(1–2), 671–678 (2001)
16. Ueda, I., Nishimura, A., Hayashi, Y.i., 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)

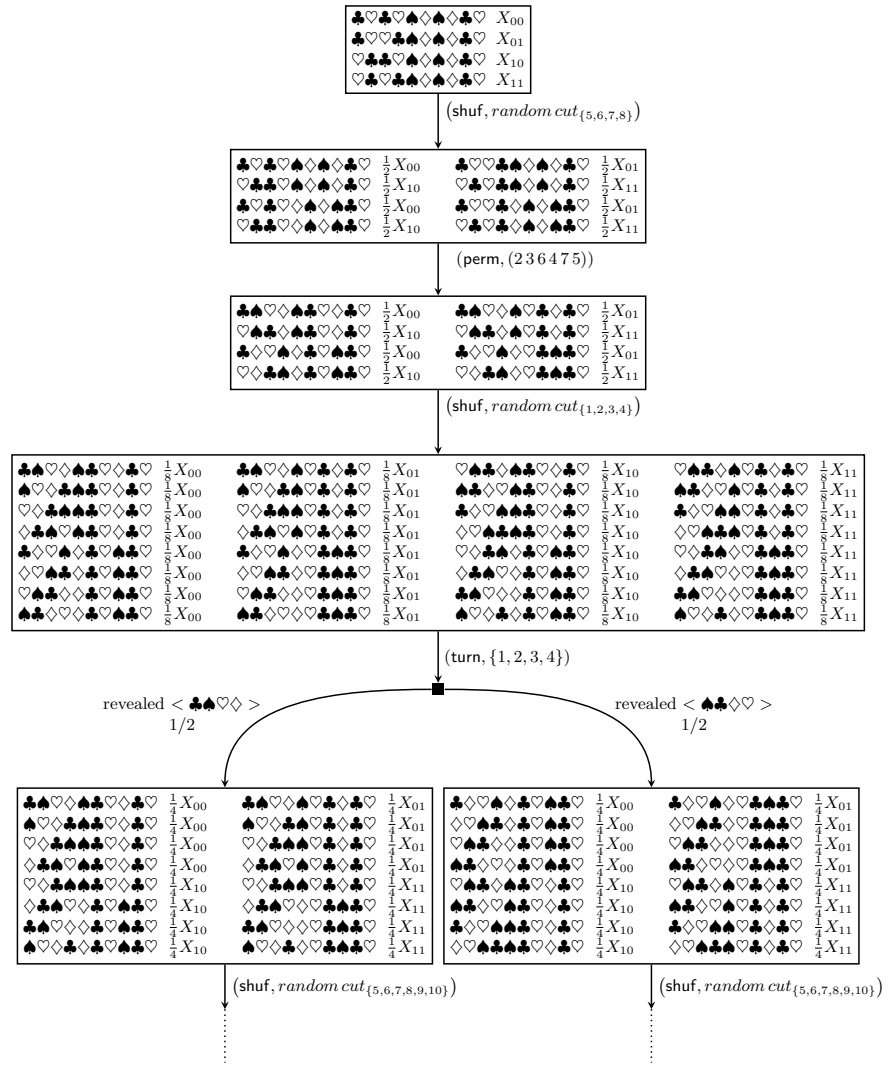


Fig. 6: The first part of $\mathcal{P}_{\mathrm{CK}}$'s KWH-tree. The expression $< \clubsuit\spadesuit\heartsuit\diamondsuit >$ means $\clubsuit\spadesuit\heartsuit\diamondsuit$, $\spadesuit\heartsuit\diamondsuit\clubsuit$, $\heartsuit\diamondsuit\clubsuit\spadesuit$, or $\diamondsuit\clubsuit\spadesuit\heartsuit$.

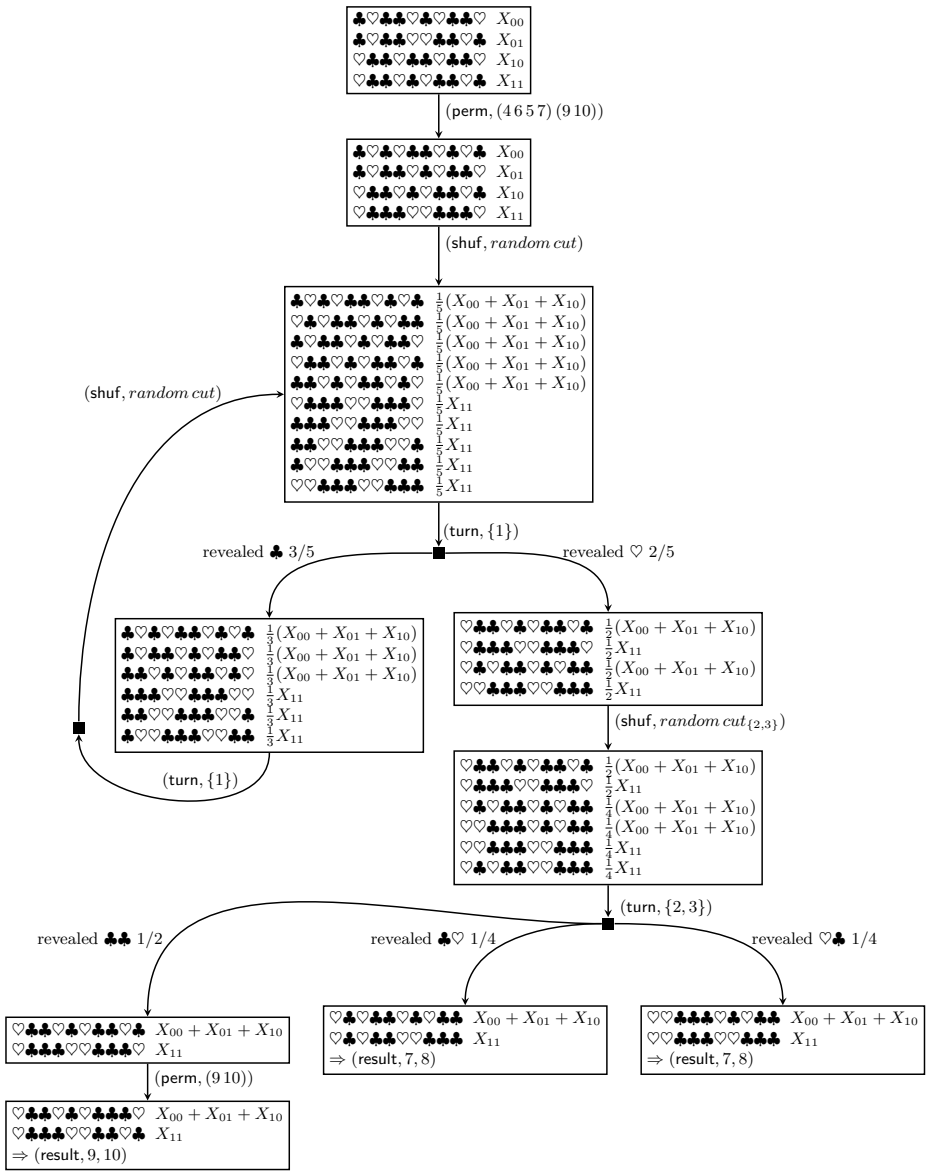


Fig. 7: $\mathcal{P}_{\mathrm{NR}}$'s KWH-tree