Card-Based Protocols with Mixed Encoding: Computable Functions and the Minimum Number of Cards

Saeki Koizumi* Xiao-Nan Lu† Hidetomo Nabeshima‡

Abstract

Card-based cryptography performs secure computation with physical cards, where a bit is usually encoded by two cards but can also be encoded by a single card. We study *mixed encoding*, in which one player gives a single-card commitment and the other a two-card commitment, for perfectly secure protocols with finite runtime and public actions. Mizuki (Theoretical Computer Science, vol. 622, 2016) proposed an AND protocol with mixed encoding as a building block for the conjunction of many variables. To give a single-card commitment, a player chooses one of two cards of different types and must keep the other card hidden. We call this card a *frozen card* and include it in the number of cards. We show that a secure protocol in committed format never turns over a single-card input, and use this to determine the functions that can be computed when all inputs are single-card commitments and to characterize those that can be computed with mixed encoding, which include AND but not XOR. We prove that Mizuki's five-card protocol is card-minimal. No AND protocol with mixed encoding uses four cards, including the frozen card, even with arbitrary shuffles and a non-committed output. For formal verification, we express the existence of a protocol as a quantified Boolean formula (QBF) in which security is a quantifier alternation, and confirm this lower bound with QBF solvers for a class of protocols. We discuss the number and type of shuffles, and analyze Mizuki's protocol under injection attacks that change only the input.

Keywords: card-based cryptography; single-card encoding; mixed encoding; impossibility proof; formal verification; quantified Boolean formula

1 Introduction

1.1 Background

Card-based cryptography performs secure computation with a deck of physical cards. Players compute a function of their private inputs without revealing any information beyond the output value. We use a deck of two types of cards, $\boxed{\clubsuit}$ and $\boxed{\heartsuit}$, whose backs are all identical and written as $\boxed{?}$. A bit $x \in \{0, 1\}$ is encoded by two cards or by one card, according to the following encoding rules:

two-card encoding: $\boxed{\clubsuit}\boxed{\heartsuit} = 0$, $\boxed{\heartsuit}\boxed{\clubsuit} = 1$; single-card encoding: $\boxed{\clubsuit} = 0$, $\boxed{\heartsuit} = 1$.

*Computer Science and Engineering Course, Department of Engineering, Integrated Graduate School of Medicine, Engineering, and Agricultural Sciences, University of Yamanashi, 4-3-11 Takeda, Kofu, Yamanashi 400-8511, Japan.

†Corresponding author. Department of Electrical, Electronics and Computer Engineering, Gifu University, 1-1 Yanagido, Gifu 501-1193, Japan. lu.xiaonan.b3@f.gifu-u.ac.jp

‡Faculty of Engineering, Graduate Faculty of Interdisciplinary Research, University of Yamanashi, 4-3-11 Takeda, Kofu, Yamanashi 400-8511, Japan. nabesima@yamanashi.ac.jp

Face-down cards holding x with the two-card and the single-card encoding are called a *two-card commitment* and a *single-card commitment* to x, respectively, and are expressed as

$$\underbrace{\boxed{?}\boxed{?}}_{x} \quad \text{and} \quad \underbrace{\boxed{?}}_{x}.$$

Cards whose symbols do not depend on the inputs are called *helping cards*. A protocol is in *committed format* if it outputs a commitment to the function value, and in *non-committed format* if this value is obtained by turning over cards. In committed format, the output commitment can be used as an input to another protocol without being revealed, provided that this protocol takes commitments with the same encoding.

Most card-based protocols, including the five-card trick of den Boer [1] in non-committed format and the six-card AND protocol of Mizuki and Sone [16] in committed format, use the two-card encoding for all inputs. For AND protocols with two-card inputs and a two-card output, the minimum number of cards has been determined for various restrictions on the runtime and on the shuffles [9, 5, 7]. In particular, Koch, Walzer, and Härtel [9] showed that four cards do not suffice for finite runtime. Moreover, Iino et al. [3] showed that there is no four-card AND protocol that uses only one closed shuffle, even in non-committed format.

The single-card encoding was introduced by Niemi and Renvall [18] together with a copy protocol that reveals the input with a small probability, called the failure probability. Mizuki and Shizuya [14] showed that no copy protocol for the single-card encoding is perfectly secure, so a positive failure probability cannot be avoided. Recently, Shinagawa [20] gave a protocol with the single-card encoding for every Boolean function, together with AND and XOR protocols with lower failure probabilities. However, none of these protocols is perfectly secure.

Mizuki [12] studied the conjunction of multiple Boolean variables given as two-card commitments. To compute it with one helping card, he splits a two-card commitment into two single-card commitments. As a building block, he uses a protocol that takes a single-card commitment to a and a two-card commitment to b and outputs a single-card commitment to $a \wedge b$ [12, Sect. 3.2]. We call this protocol *Mizuki's mixed AND protocol*, since its inputs mix the two encodings. Note that the left and the right card of a two-card commitment to b can be viewed as single-card commitments to b and $\bar{b}$, respectively. Given Alice's single-card commitment to a and Bob's two-card commitment to b, the protocol proceeds as follows.

1. Place a helping card $\boxed{\clubsuit}$ to the left of the input cards, and turn it over:

$$\boxed{\clubsuit}\ \underbrace{\boxed{?}}_{a}\ \underbrace{\boxed{?}\boxed{?}}_{b} \to \underbrace{\boxed{?}}_{0}\ \underbrace{\boxed{?}}_{a}\ \underbrace{\boxed{?}}_{b}\ \underbrace{\boxed{?}}_{\bar{b}}.$$

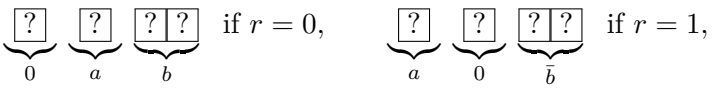
2. Exchange the second and the third card:

$$\underbrace{\boxed{?}}_{0}\ \underbrace{\boxed{?}}_{b}\ \underbrace{\boxed{?}}_{a}\ \underbrace{\boxed{?}}_{\bar{b}}.$$

3. Apply a random bisection cut, i.e., divide the sequence into two halves and swap them with probability $\frac{1}{2}$, so that nobody knows whether they were swapped:

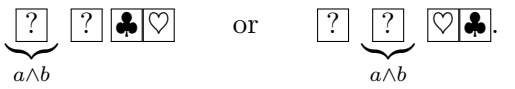
$$\big[\boxed{?}\boxed{?}\ \big|\ \boxed{?}\boxed{?}\big].$$

4. Exchange the second and the third card again. Let $r \in \{0, 1\}$ indicate whether the two halves were swapped in step 3. Then the sequence is



   so the third and the fourth card form a two-card commitment to $b \oplus r$.

5. Turn over the third and the fourth card. If they are ♣♡, the first card encodes $a \wedge b$; if they are ♡♣, the second card encodes $a \wedge b$:



Since r is uniformly random and unknown to the players, step 5 reveals only $b \oplus r$, which gives no information on a and b. If the turned cards are ♣♡, then $b \oplus r = 0$, so either $b = r = 0$ and the first card encodes $0 = a \wedge b$, or $b = r = 1$ and it encodes $a = a \wedge b$. Symmetrically, if they are ♡♣, the second card encodes $a \wedge b$.

1.2 Questions and contributions

In this paper, we call the setting in which Alice gives a single-card commitment to a and Bob gives a two-card commitment to b *mixed encoding*. To choose her input, Alice needs one card of each type, and she keeps the card of $\bar{a}$. This card must be kept hidden, since it would reveal a. We call a card of the deck that is never used by the protocol a *frozen card*, and the other cards *working cards*. With the two-card encoding, all cards are working cards. Rearrangements, turns, and shuffles act only on the working cards. Following Koch [7] and Nakai et al. [17], we include frozen cards in the number of cards and state the number of working cards separately. For example, Mizuki's mixed AND protocol uses five cards, four of which are working cards and one of which is a frozen card.

We address the following three questions for secure protocols with finite runtime in which all actions are public.

(Q1) Which Boolean functions can be computed in committed format when all inputs are given as single-card commitments?

(Q2) Which Boolean functions can be computed in committed format with mixed encoding?

(Q3) What is the minimum number of cards for AND with mixed encoding, and does the type of shuffle matter?

In this paper, we answer (Q1)–(Q3) and give a formal verification method for impossibility results.

1. For (Q1), the impossibility proof of Mizuki and Shizuya [14] concerns copying, and extensions to NOT and AND were suggested in [12, Sect. 3.1] and [6, p. 51] or stated with reference to that result [7, 4]. We give a direct proof that, when all inputs are given as single-card commitments, a secure protocol can output a single-card commitment to the value of a Boolean function f only if f is a constant or a projection. In particular, no perfectly secure AND, OR, XOR, NOT, or copy protocol with single-card outputs exists, for any number of helping cards (Theorem 3.3).

2. For (Q2), we show that a secure protocol with mixed encoding can output a single-card commitment to $f(a, b)$ if and only if the function $a \mapsto f(a, b)$ is constant or the identity for each fixed b (Theorem 3.4). So AND and OR can be computed by Mizuki's mixed AND protocol and its variants (Corollary 2.2), but XOR, NAND, and the negation of a cannot be computed with any number of cards. If Alice may give $\bar{a}$ instead of a, exactly XOR and XNOR remain impossible.

3. For (Q3), we show that five cards (four working cards and one frozen card) are necessary for AND with mixed encoding in both committed and non-committed format, regardless of the type and number of shuffles (Theorem 4.2). Hence Mizuki's mixed AND protocol and its variants of Corollary 2.2 are card-minimal (Corollary 4.3), and their single shuffle is also the minimum. With four working cards and a committed output, a random bisection cut suffices, but random cuts of the entire sequence do not (Proposition 4.5).

4. Impossibility results in card-based cryptography are usually proved by case analyses of KWH-trees [9, 5, 7], and Koch, Schrempp, and Kirsten [8] supported them by software bounded model checking. Fujita, Ikeda, Shinagawa, and Yoneyama [2] applied this model checking to four-card XOR protocols with two-card inputs that use only random cuts, and combined it with a mathematical proof that no such protocol exists for any number of shuffles. The difficulty lies in security. Unlike correctness, security is not a property of a single execution, since it compares executions with different inputs. As a methodological contribution, we state the existence of a protocol with possibilistic security as a quantified Boolean formula (QBF). In this formula, security is expressed by a quantifier alternation. If the formula is false, no secure protocol of the considered class exists (Proposition 4.6). With this formulation and QBF solvers, we confirm the impossibility of AND with four cards for protocols that turn cards only once, after all shuffles (Section 4.2).

In addition, we analyze Mizuki's mixed AND protocol under injection attacks that change only the input. We show that an input of Bob that does not follow the encoding rule, i.e., two cards of the same type, is always detected without revealing Alice's bit. Since Alice's single-card commitment always follows the encoding rule, no additional check against these injections is needed (Section 5).

The rest of this paper is organized as follows. Section 2 formalizes mixed encoding in the computational model, reviews the formal definition of security, and describes Mizuki's mixed AND protocol in this model. Section 3 determines the computable functions. Section 4 proves card-minimality and gives the QBF formulation. Section 5 analyzes injection attacks. Section 6 concludes the paper and lists open problems.

2 Preliminaries

2.1 Computational model

We use the Mizuki–Shizuya model [14] in the notation of Koch, Walzer, and Härtel [9], without its random flip action. Kastner et al. [5, Cor. 1] showed that this action can be removed from a secure protocol without affecting security or the output and without increasing the runtime. A sequence of d cards is written $(\alpha_1, \ldots, \alpha_d)$, each card being face up or face down. Its *visible sequence* shows the symbols of the face-up cards and ? for the face-down cards. A face-down card with symbol $c \in \{\clubsuit, \heartsuit\}$ is written $\frac{?}{c}$, and a face-up one $\frac{c}{?}$. When all cards are face down, we write only their symbols, and we identify $\clubsuit$ and $\heartsuit$ with 0 and 1. For example, $(\clubsuit, a, b, \bar{b})$ is the sequence of four face-down cards encoding 0, a, b, and $\bar{b}$. Let S_d denote the symmetric group of degree d, i.e., the set of all permutations of $\{1, \ldots, d\}$. A protocol applies the following actions to the current sequence:

- (perm, π) for $\pi \in S_d$ rearranges the sequence to $(\alpha_{\pi^{-1}(1)}, \dots, \alpha_{\pi^{-1}(d)})$, i.e., the card at position i moves to position $\pi(i)$.
- (turn, T) for a nonempty $T \subseteq \{1, \dots, d\}$ turns over the cards at the positions in T.
- $(\mathsf{shuffle}, \Pi, \mathcal{F})$ chooses a permutation π from a nonempty set $\Pi \subseteq S_d$ with probability $\mathcal{F}(\pi)$ and applies (perm, π), where $\mathcal{F}$ is a probability distribution on Π with $\mathcal{F}(\pi) > 0$ for every $\pi \in \Pi$. The chosen permutation π is unknown to the players, unless it can be determined from the face-up cards. The shuffle is *uniform* if $\mathcal{F}$ is the uniform distribution on Π, and in this case we write $(\mathsf{shuffle}, \Pi)$.
- $(\mathsf{result}, i_1, \dots, i_\ell)$ terminates the protocol and outputs the cards at positions $i_1, \dots, i_\ell$.

Formally, a protocol is a quadruple (D, U, Q, A), where D is a deck, U an input set of sequences, Q a set of states with an initial and a final state, and A an action function that determines the next state and the next action from the current state and the current visible sequence. All actions are public. The *visible sequence trace* of an execution is $V = (\boldsymbol{v}_0, \boldsymbol{v}_1, \dots)$, where $\boldsymbol{v}_0$ is the visible sequence of the input sequence and $\boldsymbol{v}_t$ is the visible sequence after the tth step. Given the visible sequence trace so far, each shuffle chooses its permutation independently of the input and of the previous shuffles. A protocol has *finite runtime* if there is a fixed bound on the number of steps. In this paper we consider only finite-runtime protocols.

With mixed encoding, the deck D consists of one $\boxed{\clubsuit}$ and one $\boxed{\heartsuit}$ for each of Alice and Bob, together with the helping cards. Alice places one of her cards face down as a single-card commitment to her bit a, and Bob places a two-card commitment to his bit b. These cards and the helping cards are arranged in an order that does not depend on (a, b), and we write $\Gamma^{a,b}$ for the resulting input sequence, so the input set is $U = \{\Gamma^{a,b} : a, b \in \{0, 1\}\}$. The cards of the input sequence are the *working cards*, and the cards of D that are not in it are the *frozen cards*. The *number of cards* of a protocol is $|D|$, and its *number of working cards* is the length d of its sequences. Here the only frozen card is Alice's other card, the card of $\bar{a}$, so $|D| = d + 1$. In the Mizuki–Shizuya model, every input sequence consists of all cards of D. Our input sequences leave out the frozen card. Since all actions are defined on sequences, they act only on the working cards, and the frozen card is never rearranged, turned over, or shuffled. The rest of the model therefore applies without change. The input sequence $\Gamma^{1,b}$ has one more $\heartsuit$ and one fewer $\clubsuit$ than $\Gamma^{0,b}$, so the number of $\heartsuit$s in the input sequence reveals a. Hence cards that remain face down at the end must not be turned over, as noted by Shinagawa [20].

Mizuki [12, Sect. 3.2] and Shinagawa [20] count a single-card commitment as one card, i.e., they count working cards. In this count, Mizuki's mixed AND protocol uses four cards. For his protocol for more than two inputs, however, Mizuki [12, Sect. 3.3] also counts the card that is discarded when a single-card commitment is created, since two cards of different types are needed to create it. For the same reason, Koch [7, p. 115] argued that the single-card encoding does not save cards. Similarly, Nakai et al. [17, Sect. 2.2.1] note that a player needs two cards for a bit even with the single-card encoding when all operations are public, so that $2m$ cards are necessary for m input bits. Our number of cards follows this second count, and the number of working cards follows the first.

A shuffle $(\mathsf{shuffle}, \Pi, \mathcal{F})$ is *closed* if Π is a permutation group. A *random cut*, introduced by den Boer [1], is the uniform shuffle $(\mathsf{shuffle}, \{\pi^k : 0 \le k < d\})$ with $\pi = (1\,2\,\cdots\,d)$. It acts on the entire sequence of d working cards, including face-up cards. For even d, a *random bisection cut*, introduced by Mizuki and Sone [16], divides the sequence into two halves of $d/2$ cards and swaps them with probability $\frac{1}{2}$. It is the uniform shuffle

$$(\mathsf{shuffle}, \{\mathrm{id},\ (1\ \tfrac{d}{2}{+}1)(2\ \tfrac{d}{2}{+}2)\cdots(\tfrac{d}{2}\ d)\}).$$

Both are uniform closed shuffles.

In this model, Mizuki's mixed AND protocol has the input sequence $\Gamma^{a,b} = (\clubsuit, a, b, \bar{b})$, which is the sequence after step 1 in Section 1.1. It is given in Algorithm 1, whose actions are steps 2–5 in Section 1.1. It uses five cards (four working cards and one frozen card) and one shuffle, the random bisection cut in line 2. In every execution of Algorithm 1, the visible sequences before the turn action in line 4 are $(?, ?, ?, ?)$, so the visible sequence trace is determined by its last visible sequence, $(?, ?, \clubsuit, \heartsuit)$ or $(?, ?, \heartsuit, \clubsuit)$. Lines 1–3 of Algorithm 1 can be combined into a single action $(\mathsf{shuffle}, \{\mathrm{id}, (1\ 2)(3\ 4)\})$ applied to $\Gamma^{a,b}$, which exchanges the helping card and the card of a and, at the same time, the two cards of the commitment to b.

As Mizuki [12, Sect. 3.2] notes, this protocol is closely related to the six-card AND protocol of Mizuki and Sone [16]. The former is obtained from the latter, with the roles of a and b exchanged, by removing the second card of the two-card commitment to a and the second card $\boxed{\heartsuit}$ of the helping commitment to 0. The AND protocol of Shinagawa [20] uses a similar random bisection cut, which yields $(0, y)$ or $(y, 0)$ for a single-card commitment to y, together with a two-card commitment to a random bit. In Mizuki's mixed AND protocol, the two-card commitment to b plays the role of that random bit.

2.2 Correctness, security, and KWH-trees

Let $f\colon \{0,1\}^m \to \{0,1\}$ be the Boolean function to be computed, and let I denote the input, e.g., $I = (a, b)$ for mixed encoding, where $m = 2$. A protocol *computes f in committed format* if, for every input and every execution, it terminates with a result action whose output is a two-card or a single-card commitment to $f(I)$. It *computes f in non-committed format* if, in every execution, $f(I)$ is determined by the visible sequence trace. In either case, the protocol is called *correct*.

Let the input be random with every input having positive probability, let Γ_0 be the corresponding input sequence, and let V be the visible sequence trace. Following Koch, Walzer, and Härtel [9], a protocol in committed format is *secure* if Γ_0 and V are stochastically independent. Equivalently, V has the same distribution for every input, so the definition does not depend on the input distribution. A protocol in non-committed format is secure if, for any two inputs i, i' with $f(i) = f(i')$, the distribution of V is the same under $I = i$ and $I = i'$. A protocol that is secure in either format is also called *perfectly secure*, in contrast to protocols with a positive failure probability.

Kastner et al. [5] introduced *possibilistic security* for committed format. It requires only that every visible sequence trace that occurs with positive probability under some input also occurs with positive probability under every input. Every perfectly secure protocol is possibilistically secure, but not conversely, since possibilistic security ignores the probabilities of the traces. Hence, if no

Algorithm 1 Mizuki's mixed AND protocol [12, Sect. 3.2]

Input set: $\{\Gamma^{a,b} = (\frac{?}{\clubsuit}, \frac{?}{a}, \frac{?}{b}, \frac{?}{b}) : a, b \in \{0,1\}\}$
1: $(\mathsf{perm}, (2\ 3))$
2: $(\mathsf{shuffle}, \{\mathrm{id}, (1\ 3)(2\ 4)\})$
3: $(\mathsf{perm}, (2\ 3))$
4: $(\mathsf{turn}, \{3, 4\})$
5: **if** the visible sequence is $(?, ?, \clubsuit, \heartsuit)$ **then**
6: $\quad(\mathsf{result}, 1)$
7: **else if** the visible sequence is $(?, ?, \heartsuit, \clubsuit)$ **then**
8: $\quad(\mathsf{result}, 2)$
9: **end if**

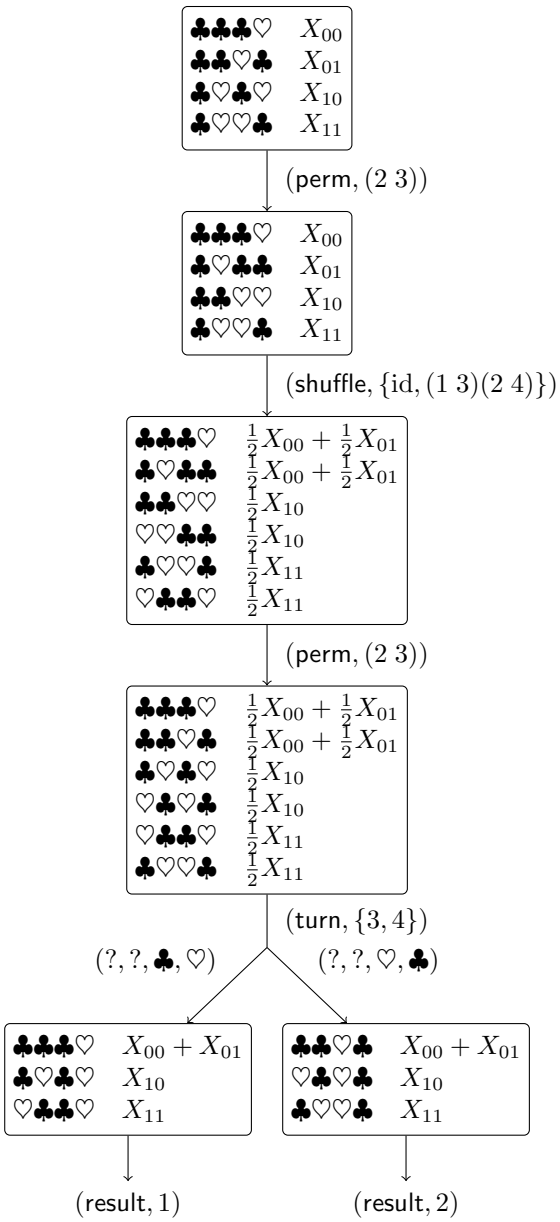


Figure 1: KWH-tree of Mizuki's mixed AND protocol (Algorithm 1). Each state lists the possible card sequences with their polynomials. After the turn action the polynomials are normalized.

possibilistically secure protocol of some class exists, no perfectly secure one exists either. We use this in Section 4.2 for formal verification.

Koch, Walzer, and Härtel [9] describe all executions of a protocol by a tree of *states*, known as a KWH-tree. For mixed encoding, a state maps every card sequence s to a polynomial $\sum_{a,b} \beta_{ab} X_{ab}$. Here X_{ab} stands for the probability of the input (a, b), and the polynomial is the probability that s is the current sequence given the visible sequence trace so far. The start state maps each input sequence $\Gamma^{a,b}$ to X_{ab}. A perm action moves the polynomials with the cards. A shuffle maps the state S to $S'(s) = \sum_{\pi \in \Pi} \mathcal{F}(\pi)\, S(\pi^{-1}(s))$. After each action, the state branches into one state for each visible sequence $\boldsymbol{v}$ that can be observed, and this state contains the sequences compatible with $\boldsymbol{v}$. If all cards are face down, only a turn action creates more than one branch.

Koch, Walzer, and Härtel showed that a protocol in committed format is secure if and only if, for each such branch, the sum of the polynomials of the sequences compatible with $\boldsymbol{v}$ has the form $\beta_{\boldsymbol{v}}(X_{00} + X_{01} + X_{10} + X_{11})$ for a constant $\beta_{\boldsymbol{v}}$. The polynomials in that branch are then normalized, i.e., divided by $\beta_{\boldsymbol{v}}$. A protocol in committed format is correct if and only if, at every result action, each sequence whose polynomial contains X_{ab} with a non-zero coefficient has output cards encoding $f(a, b)$.

In Mizuki's mixed AND protocol, turning over the third and the fourth card leaks no information, as Mizuki [12, Sect. 3.2] argued informally. For completeness, we give a formal proof with the criteria above.

Proposition 2.1. *Mizuki's mixed AND protocol (Algorithm 1) is a secure finite-runtime AND protocol in committed format for mixed encoding that outputs a single-card commitment to $a \wedge b$.*

Proof. Figure 1 shows the KWH-tree of the protocol. Every execution consists of two rearrangements, one shuffle, one turn action, and one result action, so the protocol has finite runtime. The only action that turns cards is the turn action in line 4. It yields the visible sequence $(?, ?, \clubsuit, \heartsuit)$ or $(?, ?, \heartsuit, \clubsuit)$, and in both cases the polynomials of the compatible sequences sum to $\frac{1}{2}(X_{00} + X_{01} + X_{10} + X_{11})$, so the protocol is secure by the criterion above. In the left final state, card 1 is $\heartsuit$ exactly in the sequence $\heartsuit\clubsuit\clubsuit\heartsuit$, whose polynomial is X_{11}. In the right final state, card 2 is $\heartsuit$ exactly in the sequence $\clubsuit\heartsuit\heartsuit\clubsuit$, whose polynomial is X_{11}. So in every execution the output card encodes $a \wedge b$. □

As remarked in Section 2.1, the face-down card that is not output must not be turned over. By Figure 1, it encodes a if $b = 0$ and 0 if $b = 1$, i.e., it encodes $a \wedge \bar{b}$, so turning it over would reveal a when $b = 0$.

Changing the helping card or the order of the two cards of the commitment to b gives protocols for three other functions.

Corollary 2.2. *Let $c \in \{0, 1\}$. Algorithm 1 with the input sequence $(c, a, b, \bar{b})$ is secure and outputs a single-card commitment to a if $b = 1$ and to c if $b = 0$. With the input sequence $(c, a, \bar{b}, b)$, it outputs a if $b = 0$ and c if $b = 1$. In particular, the input sequences $(\heartsuit, a, \bar{b}, b)$, $(\clubsuit, a, \bar{b}, b)$, and $(\heartsuit, a, b, \bar{b})$ give protocols for $a \vee b$ (OR), $a \wedge \bar{b}$, and $a \vee \bar{b}$, respectively. Each of them uses five cards (four working cards and one frozen card) and one random bisection cut.*

Proof. After line 3, the card sequence is $(c, a, b, \bar{b})$ or $(a, c, \bar{b}, b)$, each with probability $\frac{1}{2}$. So the turn action shows $(b, \bar{b})$ or $(\bar{b}, b)$, i.e., it reveals $b \oplus r$ for a uniformly random bit r, independently of the input. If it shows $(\clubsuit, \heartsuit)$, card 1 is c when $b = 0$ and a when $b = 1$. If it shows $(\heartsuit, \clubsuit)$, card 2 is a when $b = 1$ and c when $b = 0$. Exchanging the two cards of the commitment to b replaces b by $\bar{b}$. □

3 Computable functions with single-card commitments

This section determines which functions can be computed securely in committed format in two settings: when all inputs are given as single-card commitments, and when the inputs use mixed encoding. Both results follow from one observation. If some input is given as a single-card commitment, then a secure protocol in committed format never turns the card of that input (Lemma 3.1). From this we derive that only constants and projections can be computed when all inputs are given as single-card commitments (Theorem 3.3), and we characterize the functions that can be computed with mixed encoding (Theorem 3.4).

Throughout this section, we consider finite-runtime protocols in committed format that are secure, i.e., the visible sequence trace V is independent of the input. Let a be an input given as a

single-card commitment. We fix the values $\boldsymbol{x}$ of all other inputs, each given as a single-card or a two-card commitment, and compare the two values of a. Changing a changes only the symbol of the card of a.

Recall that the visible sequence trace of an execution is $V = (\boldsymbol{v}_0, \boldsymbol{v}_1, \dots)$, where $\boldsymbol{v}_t$ is the visible sequence after the tth step. We write $V_{\le t} = (\boldsymbol{v}_0, \dots, \boldsymbol{v}_t)$ for its prefix up to step t.

Lemma 3.1 also states that the conditional distribution of the positions of the cards, given the visible sequence trace, does not depend on a.

Lemma 3.1. *Let P be a secure finite-runtime protocol in committed format with the input described above. For each step t, let $\sigma_t \in S_d$ be the random permutation such that the card originally at position k is at position $\sigma_t(k)$ after t steps. Then*

(i) with probability 1*, no* turn *action turns the card of a;*

(ii) for every step t and every visible sequence trace $V_{\le t}$ up to step t with positive probability, the conditional distribution of σ_t given $V_{\le t}$ and the input $(a, \boldsymbol{x})$ does not depend on a.

Proof. We prove (i) and (ii) together by induction on t. Assume that the card of a is not turned in the first t steps and that (ii) holds for step t. We show that the card of a is not turned in step $t+1$ and that (ii) holds for step $t+1$. In this step, (ii) for step t is used to show (i) for step $t+1$, and both are used to show (ii) for step $t+1$. For $t = 0$, no step has been executed, and (ii) holds since $\sigma_0 = \mathrm{id}$. Since V is independent of the input, the probability $\Pr[\boldsymbol{v}_{t+1} \mid V_{\le t}, (a, \boldsymbol{x})]$ of the next visible sequence does not depend on $(a, \boldsymbol{x})$.

We first show (i) for step $t+1$. Only a turn action can turn the card of a, so suppose that step $t+1$ is a turn action. A card that it turns from face up to face down is not the card of a, since the card of a has not been turned face up in the first t steps. So let j be a position that this action turns from face down to face up. The card at position j is the card originally at position $\sigma_t^{-1}(j)$. Its symbol is a if this is the card of a, and otherwise it does not depend on a. By (ii) for step t, $q_j := \Pr[\text{card at } j \text{ is the card of } a \mid V_{\le t}, (a, \boldsymbol{x})]$ does not depend on a, and

$$\Pr[\text{card at } j \text{ is } \heartsuit \mid V_{\le t}, (a, \boldsymbol{x})] = g_j(V_{\le t}, \boldsymbol{x}) + q_j\, a$$

for some g_j that does not depend on a. The left-hand side is determined by the distribution of $\boldsymbol{v}_{t+1}$, so it does not depend on a. Hence $q_j = 0$, i.e., the card of a is not at position j.

We now show (ii) for step $t+1$. If step $t+1$ is a perm action, σ_{t+1} is obtained from σ_t by a fixed permutation, and $\boldsymbol{v}_{t+1}$ is determined by $V_{\le t}$. If it is a shuffle action, we have $\sigma_{t+1} = \pi \circ \sigma_t$ for the permutation π chosen by this shuffle, whose distribution depends only on $V_{\le t}$. The next visible sequence $\boldsymbol{v}_{t+1}$ depends only on π and on the face-up cards, whose symbols are given by $V_{\le t}$. If it is a turn action, we have $\sigma_{t+1} = \sigma_t$, and by (i) for step $t+1$ the turned cards are not the card of a, so $\boldsymbol{v}_{t+1}$ is determined by σ_t, $V_{\le t}$, and $\boldsymbol{x}$. In each case, the conditional joint distribution of σ_{t+1} and $\boldsymbol{v}_{t+1}$ given $V_{\le t}$ and $(a, \boldsymbol{x})$ is obtained from that of σ_t without using a, so by (ii) for step t it does not depend on a. Conditioning on $\boldsymbol{v}_{t+1}$, which has the same probability for both values of a, gives (ii) for step $t+1$. □

At the end of an execution, the result action is determined by the visible sequence trace. So Lemma 3.1(ii) gives the following corollary.

Corollary 3.2. *Conditioned on the visible sequence trace, each output card is either the card of a or a card whose value does not depend on a. The conditional probabilities of these cases and the conditional distribution of the value in the second case do not depend on a.*

When all inputs are given as single-card commitments, Lemma 3.1 applies to each of them, and only constants and projections can be computed.

Theorem 3.3. *Let P be a secure finite-runtime protocol in committed format whose inputs $x_1, \ldots, x_m$ are given as single-card commitments, together with any number of helping cards. If P outputs a single-card commitment to $f(x_1, \ldots, x_m)$, then f is a constant or a projection $(x_1, \ldots, x_m) \mapsto x_i$. In particular, there is no such protocol for AND, OR, XOR, or NOT, and no copy protocol for the single-card encoding.*

Proof. Let j be the position of the output card, σ the final permutation, and $k = \sigma^{-1}(j)$, so that the output card is the card originally at position k. Applying Lemma 3.1(ii) to each x_i in turn, the conditional distribution of σ given V does not depend on the input. Since j is determined by V, neither does that of k. Fix a visible sequence trace V with positive probability and a position k with $\Pr[\sigma^{-1}(j) = k \mid V] > 0$. By security, V has positive probability under every input, so this k has positive probability given V under every input. By correctness, the card originally at position k encodes $f(x_1, \ldots, x_m)$ for every input. If it is the card of x_i, then $f(x_1, \ldots, x_m) = x_i$, and if it is a helping card encoding c, then $f(x_1, \ldots, x_m) = c$. Hence f is a constant or a projection. A copy protocol would output two cards at positions $j \neq j'$ encoding x_1, so $\sigma^{-1}(j)$ and $\sigma^{-1}(j')$ would both be the original position of the card of x_1, which is impossible. □

Earlier works stated the impossibility of NOT and AND for the single-card encoding by reference to the copy impossibility of Mizuki and Shizuya [14], which treats copy protocols only, although its proof [14, Sect. 4] already uses the fact that turning over the input card would reveal its value. Theorem 3.3 proves these statements directly for finite-runtime protocols with a single-card output and any number of helping cards. Hence a positive failure probability cannot be avoided for any function other than constants and projections, as in the AND and XOR protocols of Shinagawa [20].

With mixed encoding, Lemma 3.1 restricts only the card of Alice's input. As in Mizuki's mixed AND protocol, Bob's two-card commitment can be turned over after a random bisection cut, which reveals only $b \oplus r$ for a uniformly random bit r. The revealed value can then decide which card is output. This leads to the following characterization.

Theorem 3.4. *Let $f\colon \{0,1\}^2 \to \{0,1\}$. There is a secure finite-runtime protocol in committed format for f with mixed encoding that outputs a single-card commitment to $f(a,b)$ if and only if, for each $b \in \{0,1\}$, the function $a \mapsto f(a,b)$ is constant or the identity. Moreover, no f that depends on a can be output as a two-card commitment.*

Proof. We first show that the condition is necessary. Suppose that, for some b_0, the function $a \mapsto f(a, b_0)$ is the negation, i.e., $f(a, b_0) = \bar{a}$ for both a. Let Y be the output, and fix a visible sequence trace V with positive probability. By security, V has the same probability for $(0, b_0)$ and $(1, b_0)$. By Corollary 3.2, given V, the output card is the card of a with a probability q that does not depend on a. In that case $Y = a \neq f(a, b_0)$, so $q = 0$. Hence the conditional distribution of Y does not depend on a, and

$$\Pr[Y = 1 \mid V,\, I = (0, b_0)] + \Pr[Y = 0 \mid V,\, I = (1, b_0)] = 1,$$

whereas correctness requires both terms to be 1.

Conversely, suppose that the condition is satisfied. If $a \mapsto f(a,b)$ is constant for both b, then $f \in \{0, 1, b, \bar{b}\}$, which is computed by outputting a helping card or one card of the commitment to b. If $f(a,b) = a$ for both b, output the card of a. Otherwise, $f(a,b) = a$ for one value of b and $f(a,b) = c$ for the other, where c is a constant. By Corollary 2.2, this f is computed by Algorithm 1 with the input sequence $(c, a, b, \bar{b})$ if $f(a,1) = a$, and with $(c, a, \bar{b}, b)$ if $f(a,0) = a$.

Finally, consider an output given as a two-card commitment. A two-card commitment to $f(a,b)$ consists of cards encoding $f(a,b)$ and $\overline{f(a,b)}$. By the first part of the proof, both $a \mapsto f(a,b)$ and $a \mapsto \overline{f(a,b)}$ must be constant or the identity. If one of them were the identity, the other would be the negation, so both are constant and f does not depend on a. □

The condition permits $3^2 = 9$ of the 16 Boolean functions of a and b. Besides 0, 1, a, b, and $\bar{b}$, the functions $a \wedge b$ (AND), $a \vee b$ (OR), $a \wedge \bar{b}$, and $a \vee \bar{b}$ can be computed with mixed encoding. The functions $a \oplus b$ (XOR), $\overline{a \oplus b}$ (XNOR), $\overline{a \wedge b}$ (NAND), $\overline{a \vee b}$ (NOR), $\bar{a} \wedge b$, $\bar{a} \vee b$, and $\bar{a}$ (NOT) cannot, whatever the number of cards.

Remark 3.5. In Theorem 3.4, the input a is the value of the card that Alice gives to the protocol. Mizuki [12, Sect. 3.2] noted that a NAND protocol is obtained by starting from a helping card $\boxed{\heartsuit}$ and the card of $\bar{a}$. Since $\bar{a} \vee \bar{b} = \overline{a \wedge b}$, this is the protocol for $a' \vee \bar{b}$ with the input $a' = \bar{a}$. That is, Alice sets $a' = \bar{a}$ and gives a' as her input. A player who knows the bit can negate it in this way before the protocol starts. This is not possible for a card whose value nobody knows, e.g., a card output by another protocol as in the m-input AND protocol [12, Sect. 3.3]. Such a card cannot be negated securely in committed format, since the negation of a cannot be computed with mixed encoding.

Suppose now that Alice gives either the card of a or the card of $\bar{a}$, as specified by the protocol. If she gives the card of $\bar{a}$, a protocol for g with mixed encoding outputs $g(\bar{a}, b)$, so it computes f if and only if $g(a,b) = f(\bar{a}, b)$ for all a, b. Hence f can be computed in this way if and only if f or $(a,b) \mapsto f(\bar{a}, b)$ satisfies the condition of Theorem 3.4. Both violate the condition if and only if $a \mapsto f(a,b)$ is the identity for one value of b and the negation for the other, i.e., if and only if f is XOR or XNOR. So, among all functions of a and b, exactly XOR and XNOR cannot be computed even if Alice may negate her input.

Finally, if Alice also gives her frozen card to the protocol, her input becomes a two-card commitment to a. Then every Boolean function can be computed with finite runtime [16]. So the frozen card marks the boundary of the results of this section.

4 Card-minimality of Mizuki's mixed AND protocol

4.1 Mathematical proofs

We show that Mizuki's mixed AND protocol uses the minimum number of cards, i.e., that no finite-runtime protocol with four cards (three working cards and one frozen card) exists, for any shuffles and in committed or non-committed format. We consider finite-runtime protocols in the Mizuki–Shizuya model with mixed encoding. A protocol may use any finite number of shuffles. Shuffles are arbitrary, i.e., they need not be uniform or closed. All actions are public. So Theorem 4.2 does not apply to the private model, in which a player may permute cards out of the other players' sight. A three-card AND protocol of Marcedone, Wen, and Shi [11] exists in the private model (see also Nakai et al. [17, Sect. 2.2.2]).

Every protocol with mixed encoding has exactly one frozen card, the card of $\bar{a}$, and at least three working cards, namely the card of a and the two cards of the commitment to b. With four cards, there are therefore three working cards and no helping card. Up to the order of the cards, the input sequence for an input $I = (a,b)$ is $\Gamma^{a,b} = (a, b, \bar{b})$:

$$\Gamma^{00} = (\clubsuit, \clubsuit, \heartsuit), \quad \Gamma^{01} = (\clubsuit, \heartsuit, \clubsuit), \quad \Gamma^{10} = (\heartsuit, \clubsuit, \heartsuit), \quad \Gamma^{11} = (\heartsuit, \heartsuit, \clubsuit).$$

As in Section 2.1, a permutation σ moves the card at position k to position $\sigma(k)$, and V denotes the visible sequence trace and $V_{\le t}$ its prefix up to step t.

We prove the impossibility for every protocol computing $a \wedge b$ with the following two properties, where (S) follows from security and (C) is correctness.

(S) The visible sequence trace V has the same distribution for the inputs $(0,0)$, $(0,1)$, and $(1,0)$.

(C) The output equals $a \wedge b$ with probability 1. Here the output is the value encoded by the output cards (committed format), or a value computed from V (non-committed format).

Property (S) holds for every protocol in committed format, since then V is independent of I. It also holds for every protocol in non-committed format, since then V may depend on I only through $a \wedge b$.

Our argument looks at the first turn action. Before it, all cards are face down, so the visible sequence trace does not change. Since the next action is a function of the visible sequence trace, the actions up to the first turn action are fixed, and so is the step at which it is executed, if any. At that step, the card sequence is obtained from Γ^I by a random permutation in S_3, which is the composition of the fixed perm actions and of the permutations drawn by the shuffles. The shuffles draw from fixed distributions, so the distribution of this permutation does not depend on I.

Lemma 4.1. *Suppose that the first* turn *action turns the set T of positions, and let $\sigma \in S_3$ be the random permutation such that the card sequence at that step is $\sigma(\Gamma^I)$. Fix $j \in T$, and let E_j be the event that the card at position j shows ♣. If (S) holds, then*

$$\Pr[E_j \mid I = (1,0)] = \Pr[E_j \mid I = (1,1)] = \tfrac{1}{2},$$

and on E_j the card sequence for both $I = (1,0)$ and $I = (1,1)$ has ♣ at position j and ♡ at the other positions. In particular, on E_j both inputs produce the same visible sequence trace up to that step.

Proof. Let $p_k = \Pr[\sigma(k) = j]$ for $k = 1, 2, 3$. Then $p_1 + p_2 + p_3 = 1$, and the p_k do not depend on I, since the distribution of σ does not. The card at position j is the card originally at position $\sigma^{-1}(j)$. The ♣s of $\Gamma^{00}, \Gamma^{01}, \Gamma^{10}, \Gamma^{11}$ are at positions $\{1,2\}, \{1,3\}, \{2\}, \{3\}$, respectively. Hence

$$\begin{aligned} &\Pr[E_j \mid I = (0,0)] = p_1 + p_2, && \Pr[E_j \mid I = (0,1)] = p_1 + p_3, \\ &\Pr[E_j \mid I = (1,0)] = p_2, && \Pr[E_j \mid I = (1,1)] = p_3. \end{aligned}$$

The event E_j is determined by the visible sequence right after the turn, and hence by V. So, by (S), it has the same probability for $I = (0,0)$, $(0,1)$, and $(1,0)$, which gives $p_1 + p_2 = p_1 + p_3 = p_2$. Thus $p_1 = 0$ and $p_2 = p_3 = \frac{1}{2}$, and therefore $\Pr[E_j \mid I = (1,0)] = \Pr[E_j \mid I = (1,1)] = \frac{1}{2}$. Finally, Γ^{10} and Γ^{11} contain a single ♣, so on E_j both card sequences have ♣ at position j and ♡ at the other positions. Consequently, the visible sequence after the turn is the same for both inputs. □

Theorem 4.2. *No finite-runtime protocol with mixed encoding and four cards (three working cards and one frozen card) satisfies both (S) and (C). In particular, no AND protocol with mixed encoding uses four cards, in committed or in non-committed format.*

Proof. Suppose, for a contradiction, that a finite-runtime protocol with mixed encoding and four cards satisfies (S) and (C). Let Y be its output, and write $\Gamma^{a,b}_k$ for the kth card of $\Gamma^{a,b} = (a, b, \bar{b})$.

Case 1: no card is ever turned. Then V is constant. In non-committed format, Y is a function of V, hence constant, which contradicts (C). In committed format, the result action is fixed. Let j be the position of the output card that encodes Y, and let σ be the final permutation. This card is $\Gamma^{a,b}_k$ for $k = \sigma^{-1}(j)$, and the distribution of k does not depend on the input. For every k with positive probability, (C) requires $\Gamma^{a,b}_k$ to encode $a \wedge b$ for all inputs, but $\Gamma^{10}_1 = \Gamma^{01}_2 = \Gamma^{00}_3 = ♡$, whereas $a \wedge b = 0$ for these three inputs.

Case 2: some card is turned. Let t be the step of the first turn action, T the set of positions it turns, and $j \in T$. By Lemma 4.1,

$$\Pr[E_j \mid I = (1,0)] = \Pr[E_j \mid I = (1,1)] = \tfrac{1}{2},$$

and on E_j both inputs give the same $V_{\le t}$ and the same card sequence after step t. From step t on, each action is a function of the visible sequence trace so far, perm and turn act deterministically on the card sequence, and each shuffle draws its permutation from a fixed distribution. Hence, conditioned on this $V_{\le t}$, the visible sequence trace and the final card sequence have the same joint distribution under $(1,0)$ and $(1,1)$. In either format, Y is a function of them, so

$$\Pr[Y = 0 \mid V_{\le t},\, I = (1,0)] = \Pr[Y = 0 \mid V_{\le t},\, I = (1,1)].$$

By (C), the left-hand side is 1 and the right-hand side is 0. This is a contradiction. □

With a fourth working card, the proof of Theorem 4.2 fails, because the input sequences $(\clubsuit, a, b, \bar{b})$ for $(1,0)$ and $(1,1)$ contain two ♣s each, so revealing a ♣ no longer determines the card sequence. Mizuki's mixed AND protocol turns a two-card commitment to $b \oplus r$ for a uniformly random bit r. So each of the two possible visible sequences after the turn has probability $\frac{1}{2}$ for every input, and no two inputs with different outputs end up with the same card sequence.

Corollary 4.3. *Mizuki's mixed AND protocol is card-minimal, and so are the protocols of Corollary 2.2. For finite-runtime protocols with mixed encoding, five cards (four working cards and one frozen card) are necessary and sufficient for AND, OR, $a \wedge \bar{b}$, and $a \vee \bar{b}$, in committed and in non-committed format.*

Proof. Exchanging ♣ and ♡ on all cards turns a protocol for f into a protocol for $(a,b) \mapsto \overline{f(\bar{a},\bar{b})}$ with the same number of cards and the same shuffles, and it preserves security. For $f(a,b) = a \wedge b$, this function is $a \vee b$, and vice versa. Exchanging the two cards of the commitment to b turns a protocol for f into a protocol for $(a,b) \mapsto f(a,\bar{b})$, so $a \wedge b$ becomes $a \wedge \bar{b}$, and $a \vee b$ becomes $a \vee \bar{b}$. Hence a protocol with four cards for $a \vee b$, $a \wedge \bar{b}$, or $a \vee \bar{b}$ would give one for AND, which Theorem 4.2 excludes. Five cards (four working cards and one frozen card) suffice by Proposition 2.1 and Corollary 2.2. Turning the selected output card gives a secure protocol in non-committed format. □

Remark 4.4. Every protocol with mixed encoding has exactly one frozen card, so its number of cards and its number of working cards differ by one, and Mizuki's mixed AND protocol is card-minimal in both counts (Section 2.1). By the argument of Nakai et al. recalled in Section 2.1, the players need $2m = 4$ cards. Theorem 4.2 also shows that these cards do not suffice for AND with mixed encoding, i.e., a helping card is necessary.

Table 1 summarizes finite-runtime AND protocols and impossibility results for each type of input and output, including our results for the single-card encoding and for mixed encoding. With five cards, Mizuki's mixed AND protocol uses one card fewer than the six-card protocol of Mizuki and Sone [16], which also uses one random bisection cut. With two-card inputs and a two-card output, at least five cards are needed [9], and six with closed shuffles [5]. These lower bounds concern a two-card output and do not apply to a single-card output. A two-card commitment is negated by exchanging its two cards, whereas a single-card commitment cannot be negated or copied with perfect security (Theorems 3.3 and 3.4). The output of Mizuki's mixed AND protocol can serve as the single-card input of a further protocol with mixed encoding, as in the m-input AND protocol [12, Sect. 3.3], but not where a circuit needs its negation (Remark 3.5) or uses it twice.

Table 1: Selected finite-runtime AND protocols and impossibility results. The numbers of cards include frozen cards.

Inputs	Output	Cards and shuffles	Protocol or result
two-card	non-committed	5 cards, one random cut	Five-card trick [1]
two-card	non-committed	4 cards, two closed shuffles	Mizuki–Kumamoto–Sone [13]
two-card	non-committed	4 cards, one closed shuffle: impossible	Iino et al. [3]
two-card	two-card	6 cards, one random bisection cut	Mizuki–Sone [16]
two-card	two-card	minimum 5 cards	Koch–Walzer–Härtel [9]
two-card	two-card	minimum 6 cards with closed shuffles	Kastner et al. [5]
single-card	single-card	perfectly secure: impossible	Theorem 3.3
single-card	single-card	$2^{s+2}+3$ cards ($2^{s+2}+1$ working, 2 frozen), input revealed with probability $2^{-(s+1)}$	Shinagawa [20]
mixed	single-card	5 cards (4 working, 1 frozen), one random bisection cut	Mizuki's mixed AND [12]
mixed	single-card	4 cards (3 working, 1 frozen): impossible	Theorem 4.2

Mizuki's mixed AND protocol also uses the minimum number of shuffles. Without a shuffle, the position of every card is known throughout, so a secure protocol can turn only helping cards and cannot compute AND. With two-card inputs, four cards do not suffice for AND with a single closed shuffle, even in non-committed format [3]. With mixed encoding, in contrast, four working cards and a single closed shuffle suffice for a single-card output. With four working cards and a committed output, however, the type of shuffle matters. A random bisection cut suffices, but random cuts do not.

Proposition 4.5. *No secure finite-runtime AND protocol in committed format with mixed encoding and four working cards uses only random cuts as shuffles.*

Proof. By Theorem 3.4, the output is a single-card commitment. With four working cards, the input sequence is $\Gamma^{a,b} = (c, a, b, \bar{b})$ up to the order of the cards, where c is the value of the helping card. As in Lemma 3.1, let $\sigma_t \in S_4$ be the random permutation such that the card originally at position k is at position $\sigma_t(k)$ after t steps. So the card of a is at position $\sigma_t(2)$, and by Lemma 3.1(i) this position is never turned. Every random cut is $(\mathsf{shuffle}, \{\rho^h : 0 \le h < 4\})$ with $\rho = (1\ 2\ 3\ 4)$. We call it *hiding* if it is applied while all four cards are face down.

Before the first hiding random cut. We show by induction on t that σ_t is determined by $V_{\le t}$ and that only the helping card, at position $\sigma_t(1)$, can be face up.

- A rearrangement (perm, π) gives $\sigma_{t+1} = \pi \circ \sigma_t$.
- A turn action (turn, T) has $T \subseteq \{\sigma_t(1)\}$, since position $\sigma_t(2)$ is never turned and turning $\sigma_t(3)$ or $\sigma_t(4)$ would reveal b.
- A random cut that is not hiding gives $\sigma_{t+1} = \rho^h \circ \sigma_t$ for a uniformly random h. Since some card is face up, the helping card is face up, at position $\rho^h(\sigma_t(1))$, which is different for the four values of h. So the visible sequence $\boldsymbol{v}_{t+1}$ determines h.

In particular, $V_{\le t}$ does not depend on the input up to the first hiding random cut.

Case 1: a hiding random cut is applied. Let it be step s. Then $\sigma_s = \rho^h \circ \sigma_{s-1}$ for a uniformly random h that $\boldsymbol{v}_s$ does not reveal, so

$$\Pr[\sigma_s(2) = j \mid V_{\le s}] = \tfrac{1}{4} \quad \text{for } j = 1, 2, 3, 4.$$

Rearrangements and further random cuts, all hiding while no card is face up, keep this distribution. So a turn action at a step $t > s$ would turn position $\sigma_t(2)$ with positive probability, which is excluded, and no card is turned after step s. The output position is therefore determined by $V_{\le s}$, and the output card is the card of a with probability $\frac{1}{4}$. For $b = 1$, however, the output must be a for both values of a, which by Corollary 3.2 requires this probability to be 1.

Case 2: no hiding random cut is applied. Then the final permutation is determined by V, so the output card is the card originally at a known position, which encodes c, a, b, or $\bar{b}$. None of these equals $a \wedge b$ for all inputs. □

4.2 Formal verification by QBF

We now state the existence of a protocol as a quantified Boolean formula (QBF), in which possibilistic security is expressed by a quantifier alternation. We use this formula to confirm Theorem 4.2 and Propositions 2.1 and 4.5 for protocols that turn cards only once, after all shuffles.

Let f be a Boolean function of a and b, let n be the number of working cards, and let $k \ge 1$ be the number of shuffles. For an input (a, b), the input sequence $\Gamma^{a,b}$ consists of the card of a, the two cards of the commitment to b, and $n - 3$ helping cards, e.g., $(a, b, \bar{b})$ for $n = 3$ and $(\clubsuit, a, b, \bar{b})$ for $n = 4$. All cards are initially face down. We consider protocols that apply a rearrangement $(\mathsf{perm}, \pi_\ell)$ followed by a shuffle $(\mathsf{shuffle}, \Pi_\ell, \mathcal{F}_\ell)$ for $\ell = 1, \dots, k$. After the last shuffle, such a protocol turns the cards in a fixed set T with $|T| \le n - 1$. Here $T = \emptyset$ means that no card is turned. The protocol then outputs the card at a position $R(T, \boldsymbol{v}) \notin T$ that depends on the visible sequence after the turn. We identify this visible sequence with the symbols $\boldsymbol{v} \in \{0, 1\}^T$ of the turned cards. Mizuki's mixed AND protocol belongs to this class with $k = 1$, $\pi_1 = \mathrm{id}$, $\Pi_1 = \{\mathrm{id}, (1\ 2)(3\ 4)\}$, and $T = \{3, 4\}$, since lines 1–3 of Algorithm 1 combine into this shuffle.

The sets Π_ℓ are either arbitrary, i.e., any nonempty subsets of S_n, or restricted to a class $\mathcal{C}$ of subsets of S_n. Table 2 uses the following restricted classes $\mathcal{C}$.

- The sets $\{\mathrm{id}, \tau\}$ for a transposition τ.
- The sets $\{\mathrm{id}, \tau_1 \cdots \tau_{n/2}\}$ for pairwise disjoint transpositions $\tau_1, \dots, \tau_{n/2}$, which give random bisection cuts up to conjugation by rearrangements.
- All subgroups of S_n, which give closed shuffles.
- The cyclic groups generated by an n-cycle, which give random cuts up to conjugation by rearrangements.

We express the existence of such a protocol that computes f in committed format by a formula over Boolean variables. It states that there is a protocol of this class with two properties. First, for every input and all outcomes of the shuffles, the execution is correct. Second, for every other input, some execution with that input shows the same symbols. The bounded model checking of Koch, Schrempp, and Kirsten [8] follows the KWH-tree of a protocol, whose states contain all card sequences that are possible for some input, and checks security on each state. It can therefore also express perfect security, and it permits turns between shuffles within a given step bound. Our formula needs no representation of these states. It follows single executions and relates executions with different inputs by a quantifier alternation. We identify true with 1, identify ♣ and ♡ with 0 and 1, and number the positions $1, \dots, n$. As in Section 2.1, a permutation π moves the card at position j to position $\pi(j)$. Throughout, ℓ ranges over $1, \dots, k$, π over S_n, j over the positions, and T over the sets of positions with $|T| \le n - 1$. We introduce the variables in the order of the quantifier prefix.

Encoding the protocol. The protocol is described by the following existentially quantified variables, and we write $\mathcal{P}$ for the tuple of all of them.

- $s_{\ell,\pi}$ is true if $\pi_\ell = \pi$.
- $m_{\ell,\pi}$ is true if $\pi \in \Pi_\ell$.
- u_T is true if the protocol turns the cards in T.
- $r_{T,\boldsymbol{v},i}$, for $\boldsymbol{v} \in \{0,1\}^T$ and $i \notin T$, is true if $R(T,\boldsymbol{v}) = i$.
- $c_{\ell,\Pi}$, for $\Pi \in \mathcal{C}$, is true if $\Pi_\ell = \Pi$. These variables occur only for a restricted class $\mathcal{C}$.

For Boolean variables $t_1, \ldots, t_q$, we write $\mathrm{ExactlyOne}(t_1, \ldots, t_q)$ for the formula

$$(t_1 \vee \cdots \vee t_q) \wedge \bigwedge_{1 \le h < h' \le q} (\neg t_h \vee \neg t_{h'}),$$

which is in conjunctive normal form and states that exactly one of $t_1, \ldots, t_q$ is true. We use the same notation for a family of variables indexed by a finite set, e.g., $\mathrm{ExactlyOne}(s_{\ell,\pi} : \pi \in S_n)$. The constraint $\mathrm{Protocol}(\mathcal{P})$ is defined by

$$\begin{aligned}\mathrm{Protocol}(\mathcal{P}) := \bigwedge_{\ell=1}^{k} \mathrm{ExactlyOne}(s_{\ell,\pi} : \pi \in S_n) \wedge \mathrm{ExactlyOne}(u_T : |T| \le n-1) \\ \wedge \bigwedge_{|T| \le n-1} \bigwedge_{\boldsymbol{v} \in \{0,1\}^T} \mathrm{ExactlyOne}(r_{T,\boldsymbol{v},i} : i \notin T) \wedge \mathrm{Sets}(\mathcal{P}),\end{aligned}$$

where $\mathrm{Sets}(\mathcal{P})$ constrains the sets of permutations. For arbitrary sets of permutations, it states that every Π_ℓ is nonempty,

$$\mathrm{Sets}(\mathcal{P}) := \bigwedge_{\ell=1}^{k} \bigvee_{\pi \in S_n} m_{\ell,\pi},$$

and for a restricted class $\mathcal{C}$, it states that every Π_ℓ belongs to $\mathcal{C}$,

$$\mathrm{Sets}(\mathcal{P}) := \bigwedge_{\ell=1}^{k} \Big(\mathrm{ExactlyOne}(c_{\ell,\Pi} : \Pi \in \mathcal{C}) \wedge \bigwedge_{\Pi \in \mathcal{C}} \Big(\bigwedge_{\pi \in \Pi} (\neg c_{\ell,\Pi} \vee m_{\ell,\pi}) \wedge \bigwedge_{\pi \notin \Pi} (\neg c_{\ell,\Pi} \vee \neg m_{\ell,\pi}) \Big) \Big).$$

Encoding the input and outcomes. The universally quantified variables are the input bits a and b and, for each ℓ, $\lceil \log_2 n! \rceil$ variables $\mathcal{O}_\ell$ that encode the permutation drawn by the ℓth shuffle in binary. For $\pi \in S_n$, we write $e_{\ell,\pi}$ for the conjunction of literals of $\mathcal{O}_\ell$ stating that $\mathcal{O}_\ell$ encodes π. Let $\mathcal{O} = (\mathcal{O}_1, \ldots, \mathcal{O}_k)$. Since $\mathcal{O}_\ell$ can take any value, the constraint $\mathrm{Valid}(\mathcal{O})$, which states that every $\mathcal{O}_\ell$ encodes a permutation in Π_ℓ, is defined by

$$\mathrm{Valid}(\mathcal{O}) := \bigwedge_{\ell=1}^{k} \bigvee_{\pi \in S_n} \big(e_{\ell,\pi} \wedge m_{\ell,\pi} \big).$$

This constraint and the constraints below also contain protocol variables from $\mathcal{P}$, which we do not write as arguments.

Encoding the execution. The execution with input (a, b) and outcomes $\mathcal{O}$ is described by the following existentially quantified variables. We write $\mathcal{X}$ for the tuple of all of them.

- $x_j^{(0)}$ is true if the card at position j is $\heartsuit$ at the start.
- $y_j^{(\ell)}$ is true if the card at position j is $\heartsuit$ after the ℓth rearrangement.
- $x_j^{(\ell)}$ is true if the card at position j is $\heartsuit$ after the ℓth shuffle.
- The auxiliary variable z is true if $f(a, b) = 1$.

Let $\gamma_j^{a,b}$ be the value of the jth card of $\Gamma^{a,b}$, i.e., a, b, $\bar{b}$, or a constant. The constraint $\mathrm{Trace}(a, b, \mathcal{O}, \mathcal{X})$, which states that $\mathcal{X}$ describes the execution with input (a, b) and outcomes $\mathcal{O}$, is defined by

$$\begin{aligned}\mathrm{Trace}(a, b, \mathcal{O}, \mathcal{X}) := {} & \bigwedge_{j=1}^{n} \big(x_j^{(0)} \leftrightarrow \gamma_j^{a,b}\big) \\ & \wedge \bigwedge_{\ell=1}^{k} \bigwedge_{\pi \in S_n} \bigwedge_{j=1}^{n} \big(s_{\ell,\pi} \rightarrow (y_{\pi(j)}^{(\ell)} \leftrightarrow x_j^{(\ell-1)})\big) \\ & \wedge \bigwedge_{\ell=1}^{k} \bigwedge_{\pi \in S_n} \bigwedge_{j=1}^{n} \big(e_{\ell,\pi} \rightarrow (x_{\pi(j)}^{(\ell)} \leftrightarrow y_j^{(\ell)})\big).\end{aligned}$$

In the next formula, $f(a, b)$ stands for a propositional formula in a and b, e.g., $a \wedge b$. The constraint $\mathrm{Correct}(a, b, \mathcal{X})$, which states that the output card encodes $f(a, b)$, is defined by

$$\mathrm{Correct}(a, b, \mathcal{X}) := \big(z \leftrightarrow f(a, b)\big) \wedge \bigwedge_{|T| \le n-1} \bigwedge_{\boldsymbol{v} \in \{0,1\}^T} \bigwedge_{i \notin T} \Big(u_T \wedge r_{T,\boldsymbol{v},i} \wedge \bigwedge_{j \in T} \big(x_j^{(k)} \leftrightarrow v_j\big) \rightarrow \big(x_i^{(k)} \leftrightarrow z\big)\Big).$$

Encoding security. A second execution is described by the universally quantified other input a' and b' and the existentially quantified outcomes $\mathcal{O}'$ and card variables $\mathcal{X}'$, which are defined as $\mathcal{O}$ and the card variables of $\mathcal{X}$. The constraints $\mathrm{Valid}(\mathcal{O}')$ and $\mathrm{Trace}(a', b', \mathcal{O}', \mathcal{X}')$ describe this execution. The constraint $\mathrm{Same}(\mathcal{X}, \mathcal{X}')$, which states that the turned cards show the same symbols in both executions, is defined by

$$\mathrm{Same}(\mathcal{X}, \mathcal{X}') := \bigwedge_{|T| \le n-1} \bigwedge_{j \in T} \big(u_T \rightarrow (x_j'^{(k)} \leftrightarrow x_j^{(k)})\big).$$

Encoding the existence of a protocol. Combining the constraints above gives the formula

$$\begin{aligned}\exists \mathcal{P}\ \forall a, b, \mathcal{O}\ \exists \mathcal{X}\ \forall a', b'\ \exists \mathcal{O}', \mathcal{X}' :\ & \\ \mathrm{Protocol}(\mathcal{P}) \wedge \Big[\mathrm{Valid}(\mathcal{O}) \rightarrow \big(& \mathrm{Trace}(a, b, \mathcal{O}, \mathcal{X}) \wedge \mathrm{Correct}(a, b, \mathcal{X}) \\ & \wedge \mathrm{Valid}(\mathcal{O}') \wedge \mathrm{Trace}(a', b', \mathcal{O}', \mathcal{X}') \wedge \mathrm{Same}(\mathcal{X}, \mathcal{X}')\big)\Big]. \quad (1)\end{aligned}$$

The last three conjuncts express possibilistic security, and the alternation $\forall a', b'\ \exists \mathcal{O}', \mathcal{X}'$ relates executions with different inputs. The constraint $\mathrm{Protocol}(\mathcal{P})$ is imposed outside the implication, since otherwise empty sets of permutations could make the execution requirements vacuous. Dropping the part after $\mathrm{Correct}(a, b, \mathcal{X})$ yields the formula

$$\exists \mathcal{P}\ \forall a, b, \mathcal{O}\ \exists \mathcal{X} :\ \mathrm{Protocol}(\mathcal{P}) \wedge \Big[\mathrm{Valid}(\mathcal{O}) \rightarrow \big(\mathrm{Trace}(a, b, \mathcal{O}, \mathcal{X}) \wedge \mathrm{Correct}(a, b, \mathcal{X})\big)\Big], \quad (2)$$

which only requires correctness.

Proposition 4.6. *Formula* (1) *is true if and only if some protocol of the class computes f with possibilistic security.*

Proof. Since all cards are face down before the turn, the visible sequence trace of a protocol of the class is determined by the symbols of the turned cards. Let a protocol of the class compute f with possibilistic security. Assign to $\mathcal{P}$ its rearrangements, the sets of permutations of its shuffles, its set T, and its output rule R. These satisfy Protocol($\mathcal{P}$). By the definition of a shuffle, every permutation in Π_ℓ is drawn with positive probability. For all inputs and all outcomes $\mathcal{O}$ that encode permutations in these sets, Trace$(a, b, \mathcal{O}, \mathcal{X})$ determines the card variables of $\mathcal{X}$, and with $z = f(a, b)$, correctness implies Correct$(a, b, \mathcal{X})$. For every other input (a', b'), possibilistic security implies that the symbols of the turned cards occur with positive probability under (a', b'), so suitable outcomes $\mathcal{O}'$ that encode permutations in these sets exist. For all other outcomes $\mathcal{O}$, Valid($\mathcal{O}$) is false. Hence formula (1) is true.

Conversely, let formula (1) be true, and fix values of $\mathcal{P}$ for which the rest of formula (1) is true. By Protocol($\mathcal{P}$), they describe a protocol of the class, in which we let each shuffle draw a permutation of Π_ℓ uniformly at random. Then every outcome of the shuffles has positive probability and is encoded by some $\mathcal{O}$ with Valid($\mathcal{O}$) true. So, by Correct$(a, b, \mathcal{X})$, the protocol computes f, and by Same$(\mathcal{X}, \mathcal{X}')$, the symbols of the turned cards under every input and outcome also occur with positive probability under every other input. □

Since every secure protocol is possibilistically secure, no secure protocol of the class computes f if formula (1) is false. If formula (1) is true, the values of $\mathcal{P}$ found by a solver give a protocol of the class with possibilistic security, and its perfect security still requires a separate verification, e.g., by a KWH-tree.

Encoding in conjunctive normal form. Formulas (1) and (2) are in prenex form, i.e., a sequence of quantifiers followed by a quantifier-free part, which is called the *matrix*. For a QBF solver, the matrix is written in conjunctive normal form. Except for the implication with premise Valid($\mathcal{O}$) and for Valid($\mathcal{O}'$), the constraints translate into clauses directly. For this implication, we add the following existentially quantified auxiliary variables, all in the quantifier block of $\mathcal{X}$, i.e., after a, b, $\mathcal{O}$ and before a', b'.

- For each ℓ and π, a variable, again written $e_{\ell,\pi}$, with clauses stating that it is equivalent to the conjunction $e_{\ell,\pi}$.
- Variables $w_1, \ldots, w_k$ and g, with the clauses

$$e_{\ell,\pi} \wedge m_{\ell,\pi} \to w_\ell, \qquad w_1 \wedge \cdots \wedge w_k \to g.$$

If Valid($\mathcal{O}$) is true, the clauses for w_ℓ and g force every w_ℓ and then g to be true. Otherwise, some w_ℓ and g can be set to false. Adding the literal $\neg g$ to every clause of the constraints inside the implication of (1) and (2) therefore expresses this implication, and Protocol($\mathcal{P}$) stays outside it. In the second execution, $\mathcal{O}'$ is existentially quantified, so we encode it directly by variables $e'_{\ell,\pi}$. The constraints ExactlyOne$(e'_{\ell,\pi} : \pi \in S_n)$ and $e'_{\ell,\pi} \to m_{\ell,\pi}$ express Valid($\mathcal{O}'$), and Trace$(a', b', \mathcal{O}', \mathcal{X}')$ uses $e'_{\ell,\pi}$ in place of $e_{\ell,\pi}$. These translations preserve the truth value.

Results. The class contains no rearrangement after the last shuffle, since such a rearrangement can be omitted by renaming the positions in T and R accordingly. For arbitrary sets of permutations, it suffices to consider $k = 1$, since consecutive rearrangements and shuffles combine into a single

Table 2: Results of the QBF formulation for AND with one shuffle (DepQBF).

Input sequence	Shuffle	Security	Variables	Clauses	Truth value	Time (s)
$(a, b, \bar{b})$	$\{\mathrm{id}, \tau\}$	yes	89	372	false	< 0.1
$(a, b, \bar{b})$	closed	yes	92	402	false	< 0.1
$(a, b, \bar{b})$	arbitrary	yes	86	351	false	< 0.1
$(a, b, \bar{b})$	arbitrary	no	69	233	true	< 0.1
$(\clubsuit, a, b, \bar{b})$	random bisection cuts	yes	258	2105	true	< 0.1
$(\clubsuit, a, b, \bar{b})$	closed	yes	285	3185	true	0.2
$(\clubsuit, a, b, \bar{b})$	random cuts	yes	258	2105	false	1.3

shuffle whose set of permutations is again a subset of S_n. For a restricted class, this is not the case. For example, two random cuts with a rearrangement between them combine into a shuffle that is in general not a random cut. For restricted classes, the results in Table 2 therefore cover only protocols with one shuffle. For random cuts with four working cards, Proposition 4.5 covers any number of shuffles.

Table 2 shows the results for AND with one shuffle ($k = 1$). The numbers of variables and clauses grow with $n!$, e.g., ExactlyOne$(s_{\ell,\pi} : \pi \in S_n)$ has $n!(n! - 1)/2 + 1$ clauses. We computed the truth values with the QBF solver DepQBF 6.03 [10] and checked them with CAQE 4.0.1 [19], which gives the same truth values. By Proposition 4.6, "false" means that no protocol of the class computes AND with possibilistic security. Each row gives the input sequence, the class of the shuffle, whether possibilistic security is required, the numbers of variables and clauses, the truth value, and the wall-clock time taken by DepQBF. The truth value is that of formula (1), or of formula (2) if security is not required. Times below 0.1 s are shown as < 0.1. The times were measured on a MacBook Pro with an Apple M1 processor (8 cores) and 16 GB of memory under macOS 15.7.

For three working cards, formula (1) is false even for arbitrary sets of permutations, and hence for every restricted class. Since $k = 1$ suffices for arbitrary sets of permutations, no secure AND protocol of the class with three working cards and any number of shuffles exists, in agreement with Theorem 4.2. Formula (2), which omits security, is true. For example, it is satisfied by the protocol that turns the card at position 3, i.e., $\bar{b}$, and outputs the card at position 2 if this card shows $\boxed{\heartsuit}$, and the card at position 1 otherwise. This protocol reveals b. So, as in the proof of Theorem 4.2, the impossibility for three working cards is due to the security requirement, not to correctness.

For four working cards, with the input sequence $(\clubsuit, a, b, \bar{b})$, formula (1) is true for random bisection cuts, in agreement with Proposition 2.1. For example, it is satisfied by the shuffle $(\mathsf{shuffle}, \{\mathrm{id}, (1\ 2)(3\ 4)\})$ followed by turning the card at position 3 and outputting the card at position 1 if this card shows $\boxed{\clubsuit}$, and the card at position 2 otherwise. This is Mizuki's mixed AND protocol, except that it turns only the card at position 3, not both cards at positions 3 and 4. Formula (1) is also true for closed shuffles, which include this random bisection cut. With random cuts only, formula (1) is false, in agreement with Proposition 4.5. With two random cuts ($k = 2$), formula (1) is also false, but DepQBF needed 489 s, compared with 1.3 s for $k = 1$.

5 Injection attacks

The security notion of Section 2.2 assumes that the players follow the protocol. The inputs are placed face down, so a player may give an input that does not follow the encoding rule. Mizuki and Shizuya [15, Sect. 3] call this an *injection attack*, in which the attacker places two cards of the same type, $\boxed{\clubsuit}\boxed{\clubsuit}$ or $\boxed{\heartsuit}\boxed{\heartsuit}$, instead of a two-card commitment. They showed that in the five-card trick,

Table 3: AND protocols under injection attacks.

Protocol	Attacker	Injected	Detected	Output 1 for $h = 0$	Invalid output	Distance
Five-card trick [1]	Alice or Bob	♣♣	1	0	–	1
	Alice or Bob	♡♡	1	0	–	0
Mizuki–Sone [16]	Alice	either	1	0	–	0
	Bob	either	0	0	if $h = 1$	0
Mizuki–Kumamoto–Sone [13]	Alice or Bob	either	0	1/4	–	1/2
Mizuki's mixed AND [12, Sect. 3.2]	Alice	not possible (single card)				
	Bob	either	1	0	–	0

Alice's input ♣♣ reveals Bob's bit when the cards are turned over. To detect such inputs, they check each two-card commitment before the protocol. The check reveals whether the commitment follows the encoding rule and nothing about its value. With the single-card encoding, every card is a valid input, ♣ for 0 and ♡ for 1, so a player who places one card cannot violate the encoding rule. In this section, we analyze Mizuki's mixed AND protocol under injection attacks and compare it with three AND protocols that use the two-card encoding. The first is the five-card trick, for which the attack was found. The other two use a random bisection cut, one in committed format [16] and one in non-committed format [13].

One player is the attacker, and the other player gives the input $h \in \{0, 1\}$ correctly. After placing the input, both players follow the protocol, including the prescribed shuffle distributions. Thus the attacker changes only the input, not the actions of the protocol. If a visible sequence occurs that cannot occur when both inputs follow the encoding rule, we say that the injection is *detected*, and the protocol is aborted.

An injection needs a card from outside the deck, since each player is given one ♣ and one ♡. As Mizuki and Shizuya note [15, Sect. 3.1], this does not prevent an injection, since the attacker may covertly bring another card with the same back. Koch [6, Sects. 12.1 and 12.9] prevents it by keeping the cards ♣♡ on the table and letting the player choose whether they are exchanged, by a procedure that hides this choice and uses helping cards with a different back. Our model, in which all actions are public, has no such action. We therefore assume that the attacker can bring the extra card, and analyze whether the protocol detects the injection.

Table 3 summarizes the analysis of four AND protocols. For each protocol, attacker, and pair of injected cards, it gives

- the probability that the injection is detected ("Detected"),
- the probability that the output is 1 although $h = 0$ ("Output 1 for $h = 0$"),
- for an output in committed format, whether it fails to be a valid commitment ("Invalid output"),
- how much the visible sequence trace reveals about h, measured by the total variation distance ("Distance")

$$\mathrm{TV}(P_0, P_1) = \frac{1}{2} \sum_V |P_0(V) - P_1(V)|,$$

where $P_h(V)$ is the probability of the visible sequence trace V when the honest input is h.

Since $a \wedge b = 0$ whenever one input is 0, an output 1 for $h = 0$ is caused by the injection. The entry "–" means that the output is not in committed format or that the protocol always aborts, and "either" means that $\boxed{\clubsuit}\boxed{\clubsuit}$ and $\boxed{\heartsuit}\boxed{\heartsuit}$ give the same values. The distance is 0 if the visible sequence trace reveals nothing about h, and 1 if it determines h. The visible sequence trace ends when the protocol terminates or is aborted. It does not include a later turn of a committed output or of unused face-down cards. In non-committed format, it includes the output.

All shuffles in these protocols are uniform and have at most five outcomes, so each entry follows from a short case analysis, and we give the main cases below. We write the sequences with the symbols ♣ and ♡ identified with 0 and 1.

In the five-card trick [1], Alice places a two-card commitment to $\bar{a}$, a helping card $\boxed{\heartsuit}$ is placed next to it, and Bob places a two-card commitment to b. After a random cut, all five cards are turned, and $a \wedge b = 1$ if and only if the three ♡s are consecutive up to cyclic rotation. If Alice places $\boxed{\clubsuit}\boxed{\clubsuit}$, the sequence is $(\clubsuit, \clubsuit, \heartsuit, b, \bar{b})$, which contains only two ♡s, so every execution is detected. Up to rotation, the turned cards are $\boxed{\clubsuit}\boxed{\clubsuit}\boxed{\heartsuit}\boxed{\clubsuit}\boxed{\heartsuit}$ if $b = 0$ and $\boxed{\clubsuit}\boxed{\clubsuit}\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\clubsuit}$ if $b = 1$, so they determine b, as observed by Mizuki and Shizuya [15]. If Alice places $\boxed{\heartsuit}\boxed{\heartsuit}$, the sequence contains four ♡s, so every execution is again detected. Now the turned cards $\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\clubsuit}\boxed{\heartsuit}$ and $\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\clubsuit}$ are equal up to rotation, so they reveal nothing about b. An injection by Bob is analyzed in the same way.

In the protocol of Mizuki and Sone [16], the input consists of Alice's commitment to a, a helping commitment to 0, and Bob's commitment to b. After a random bisection cut and two rearrangements, cards 1 and 2 are Alice's two cards in an order given by a uniformly random bit r. These two cards are turned. If they are $\boxed{\clubsuit}\boxed{\heartsuit}$, i.e., $a \oplus r = 0$, the output is cards 3 and 4, and otherwise cards 5 and 6. In both cases the output consists of Bob's two cards if $a = 1$, and of the two cards of the commitment to 0 if $a = 0$. If Alice places $\boxed{\clubsuit}\boxed{\clubsuit}$ or $\boxed{\heartsuit}\boxed{\heartsuit}$, the turned cards are $\boxed{\clubsuit}\boxed{\clubsuit}$ or $\boxed{\heartsuit}\boxed{\heartsuit}$, which never happens in an honest execution. Hence every execution is detected, and Bob's cards are never turned. If Bob places $\boxed{\clubsuit}\boxed{\clubsuit}$ or $\boxed{\heartsuit}\boxed{\heartsuit}$, the turned cards are $\boxed{\clubsuit}\boxed{\heartsuit}$ or $\boxed{\heartsuit}\boxed{\clubsuit}$ with probability $\frac{1}{2}$ each, for both values of a. Hence the injection is never detected, and the visible sequence trace reveals nothing about a. The output is a valid commitment to 0 if $a = 0$, but Bob's two cards of the same type if $a = 1$. Turning the output cards over later therefore reveals a.

In the protocol of Mizuki, Kumamoto, and Sone [13], a random bisection cut is applied to $(a, \bar{a}, b, \bar{b})$, and then cards 2 and 3 are exchanged with probability $\frac{1}{2}$. Card 2 is turned, then card 4 if card 2 is $\boxed{\clubsuit}$ and card 1 otherwise, and the output is 1 if and only if the two turned cards show the same symbol. If Alice places $\boxed{\clubsuit}\boxed{\clubsuit}$, the four equally likely sequences before the turns are

$$(\clubsuit, \clubsuit, b, \bar{b}), \qquad (\clubsuit, b, \clubsuit, \bar{b}), \qquad (b, \bar{b}, \clubsuit, \clubsuit), \qquad (b, \clubsuit, \bar{b}, \clubsuit).$$

For $b = 0$, the turned cards are $\boxed{\clubsuit}\boxed{\heartsuit}$, $\boxed{\clubsuit}\boxed{\heartsuit}$, $\boxed{\heartsuit}\boxed{\clubsuit}$, and $\boxed{\clubsuit}\boxed{\clubsuit}$, and the outputs are 0, 0, 0, and 1. For $b = 1$, the turned cards are $\boxed{\clubsuit}\boxed{\clubsuit}$, $\boxed{\heartsuit}\boxed{\clubsuit}$, $\boxed{\clubsuit}\boxed{\clubsuit}$, and $\boxed{\clubsuit}\boxed{\clubsuit}$, and the outputs are 1, 0, 1, and 1. Each of these visible sequences also occurs in an honest execution, so no execution is detected. For $h = b = 0$, the output is 1 with probability $\frac{1}{4}$, and the distributions of the turned cards for $b = 0$ and $b = 1$ have total variation distance $\frac{1}{2}$. The analysis for $\boxed{\heartsuit}\boxed{\heartsuit}$ and for an injection by Bob is similar and gives the same values.

In Mizuki's mixed AND protocol, only Bob can make an injection, since Alice places a single card. If Bob places $\boxed{\clubsuit}\boxed{\clubsuit}$ or $\boxed{\heartsuit}\boxed{\heartsuit}$, cards 3 and 4 after step 4 are his two cards (Section 1.1), so step 5 shows $\boxed{\clubsuit}\boxed{\clubsuit}$ or $\boxed{\heartsuit}\boxed{\heartsuit}$, which never happens in an honest execution. So every injection is detected, and since the turned cards do not depend on a, the visible sequence trace reveals nothing about a. The protocol thus needs no additional check against these injections, whereas the check of Mizuki and Shizuya [15] costs two helping cards and one random bisection cut for each two-card commitment. Turning both cards is essential here. Turning card 3 alone would still give a correct and secure protocol, since this card encodes $b \oplus r$. However, card 3 alone shows $\boxed{\clubsuit}$ or $\boxed{\heartsuit}$, as in an

honest execution, so an injection by Bob would not be detected. Among the protocols in Table 3, Mizuki's mixed AND protocol is the only one in which every injection is detected and the visible sequence trace reveals nothing about the honest input.

6 Conclusion and open problems

We studied card-based protocols with mixed encoding, in which one input is given as a single-card commitment and the other as a two-card commitment. The key observation is that a secure protocol in committed format never turns over a single-card commitment given as input (Lemma 3.1). It implies that, when all inputs are single-card commitments, a single-card output can only be a constant or a projection (Theorem 3.3). With mixed encoding, it yields a characterization of the computable functions (Theorem 3.4), which permits nine of the sixteen Boolean functions of two variables, including AND and OR but not XOR and NAND. For AND and OR, five cards (four working cards and one frozen card) are necessary and sufficient (Theorem 4.2 and Corollary 4.3), so Mizuki's mixed AND protocol is card-minimal. It uses a single random bisection cut, and one shuffle is the minimum, whereas random cuts of the entire sequence alone do not suffice with four working cards in committed format (Proposition 4.5). As a method, we expressed the existence of a protocol with possibilistic security as a quantified Boolean formula, in which security is a quantifier alternation (Proposition 4.6). With this formula and QBF solvers, we confirmed Theorem 4.2 and Propositions 2.1 and 4.5 for protocols that turn cards only once, after all shuffles. Finally, suppose that Bob changes only his input in Mizuki's mixed AND protocol. Then turning both cards of his commitment detects every injection of two cards of the same type without revealing Alice's bit, so no additional check against these injections is needed (Section 5).

Several questions remain open. Our characterizations assume committed format, and a corresponding classification for non-committed format is not established here. With random cuts of the entire sequence only, the minimum number of cards for AND with mixed encoding in committed format is more than five (Proposition 4.5) and is not determined here. The QBF formulation of Section 4.2 applies to any number k of shuffles. However, each further shuffle adds universally quantified variables for its outcome, and the solving time grows quickly with k, as the case of random cuts with four working cards shows. For arbitrary sets of permutations, $k = 1$ suffices, whereas restricted classes such as random cuts may need larger k, for which a more efficient formulation is needed. Extending the formulation to protocols that turn cards between shuffles is also open. Finally, protocols with more inputs can combine the two encodings in more ways. For example, it is open whether the $(2m + 1)$-card m-input AND protocol [12, Sect. 3.3], which uses Mizuki's mixed AND protocol as a building block, is card-minimal.

Declarations

- Funding: This work was partly supported by JSPS KAKENHI Grant Numbers JP22K13949 and JP24K14819 (X.-N. Lu), and JP26K02983, JP26K02897, JP25K03097, and JP25K03190 (H. Nabeshima).

- Competing interests: The authors have no competing interests to declare that are relevant to the content of this article.

- Data and code availability: The generator of formulas (1) and (2), the instances of Table 2 and of the run with two random cuts in Section 4.2, and the solver outputs are available at `https://github.com/xnlu-math/card-based-cryptography-mixed-encoding-qbf`.

References

[1] Bert den Boer. More efficient match-making and satisfiability: The five card trick. In *Advances in Cryptology – EUROCRYPT '89*, volume 434 of *LNCS*, pages 208–217. Springer, 1990.

[2] Kazuhiro Fujita, Shota Ikeda, Kazumasa Shinagawa, and Kazuki Yoneyama. Formal verification and proof of impossibility for four-card XOR protocols using only random cuts. In *Proceedings of the 12th ACM ASIA Public-Key Cryptography Workshop (APKC 2025)*, pages 9–16. ACM, 2025.

[3] Shizuru Iino, Shota Ikeda, Kazumasa Shinagawa, Yang Li, Kazuo Sakiyama, and Daiki Miyahara. Impossibility of four-card AND protocols with a single closed shuffle. In *Cryptology and Network Security (CANS 2025)*, volume 16351 of *LNCS*, pages 213–229. Springer, 2026.

[4] Shunnosuke Ikeda and Kazumasa Shinagawa. Impossibility results of card-based protocols via mathematical optimization. In *Information Security and Cryptology – ICISC 2025*, volume 16487 of *LNCS*, pages 368–382. Springer, 2026.

[5] Julia Kastner, Alexander Koch, Stefan Walzer, Daiki Miyahara, Yu-ichi Hayashi, Takaaki Mizuki, and Hideaki Sone. The minimum number of cards in practical card-based protocols. In *Advances in Cryptology – ASIACRYPT 2017*, volume 10626 of *LNCS*, pages 126–155. Springer, 2017.

[6] Alexander Koch. *Cryptographic Protocols from Physical Assumptions*. PhD thesis, Karlsruhe Institute of Technology, 2019.

[7] Alexander Koch. The landscape of optimal card-based protocols. *Mathematical Cryptology*, 1(2):115–131, 2022.

[8] Alexander Koch, Michael Schrempp, and Michael Kirsten. Card-based cryptography meets formal verification. *New Generation Computing*, 39(1):115–158, 2021.

[9] Alexander Koch, Stefan Walzer, and Kevin Härtel. Card-based cryptographic protocols using a minimal number of cards. In *Advances in Cryptology – ASIACRYPT 2015*, volume 9452 of *LNCS*, pages 783–807. Springer, 2015.

[10] Florian Lonsing and Uwe Egly. DepQBF 6.0: A search-based QBF solver beyond traditional QCDCL. In *Automated Deduction – CADE 26*, volume 10395 of *LNCS*, pages 371–384. Springer, 2017.

[11] Antonio Marcedone, Zikai Wen, and Elaine Shi. Secure dating with four or fewer cards. Cryptology ePrint Archive, Paper 2015/1031, 2015.

[12] Takaaki Mizuki. Card-based protocols for securely computing the conjunction of multiple variables. *Theoretical Computer Science*, 622:34–44, 2016.

[13] Takaaki Mizuki, Michihito Kumamoto, and Hideaki Sone. The five-card trick can be done with four cards. In *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 598–606. Springer, 2012.

[14] Takaaki Mizuki and Hiroki Shizuya. A formalization of card-based cryptographic protocols via abstract machine. *International Journal of Information Security*, 13(1):15–23, 2014.

[15] Takaaki Mizuki and Hiroki Shizuya. Practical card-based cryptography. In *Fun with Algorithms (FUN 2014)*, volume 8496 of *LNCS*, pages 313–324. Springer, 2014.

[16] Takaaki Mizuki and Hideaki Sone. Six-card secure AND and four-card secure XOR. In *Frontiers in Algorithmics (FAW 2009)*, volume 5598 of *LNCS*, pages 358–369. Springer, 2009.

[17] Takeshi Nakai, Keita Iwanari, Tomoki Ono, Yoshiki Abe, Yohei Watanabe, and Mitsugu Iwamoto. Card-based cryptography with a standard deck of cards, revisited: Efficient protocols in the private model. *New Generation Computing*, 42:345–358, 2024.

[18] Valtteri Niemi and Ari Renvall. Secure multiparty computations without computers. *Theoretical Computer Science*, 191:173–183, 1998.

[19] Markus N. Rabe and Leander Tentrup. CAQE: A certifying QBF solver. In *Formal Methods in Computer-Aided Design (FMCAD 2015)*, pages 136–143. IEEE, 2015.

[20] Kazumasa Shinagawa. Card-based protocols with single-card encoding. In *Theoretical Aspects of Computing – ICTAC 2024*, volume 15373 of *LNCS*, pages 182–194. Springer, 2025.