Cooking Cryptographers: Secure Multiparty Computation Based on Balls and Bags

Daiki Miyahara
Cyberscience Center,
Tohoku University
Sendai, Japan
National Institute of Advanced
Industrial Science and Technology
Tokyo, Japan

Yuichi Komano
Corporate R&D Center,
Toshiba Corporation
Kawasaki, Japan

Takaaki Mizuki
Cyberscience Center,
Tohoku University
Sendai, Japan
National Institute of Advanced
Industrial Science and Technology
Tokyo, Japan

Hideaki Sone
Cyberscience Center,
Tohoku University
Sendai, Japan

***Abstract*—Imagine two cryptographers wishing to securely compute the AND value of their secret input bits. They are in the kitchen, where all they have are three saucepans with Borscht soup and some kinds of ingredients. Interestingly, by secretly putting ingredients in the saucepans depending on their inputs, they can find only the AND value from the taste of the cooked Borscht. Because cooking for secure computations is not so convenient, let us regard ingredients and saucepans (with Borscht soup) as balls and bags, respectively, which are easy to handle and also familiar tools for learning Probability in high school. Then, our problem is generalized as: *Can we realize secure multiparty computations (MPCs) with balls and bags*?**

There are techniques to realize MPCs with everyday objects, such as physical cards, coins, and a PEZ dispenser. We encode the input bits with such objects and securely compute some predetermined function using them. In this paper, we present a novel technique based on the physical properties of balls and bags. That is, our challenges are how to utilize an interesting feature that the balls become disordered immediately after they are put into a bag, namely they are "*automatically shuffled*." We give the first framework of MPCs using balls and bags (namely, *ball-based cryptography*), and propose secure AND computation and general MPCs. Our protocols are realizations of usable security which helps people with understanding the principles of MPCs as well as solving social problems in daily life.

***Index Terms*—Secure multiparty computation, Real-life hands-on cryptography, Urn problem**

I. Introduction

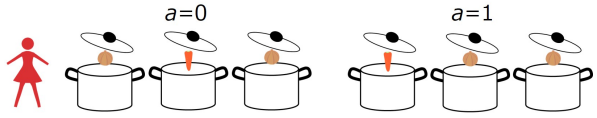
Three cryptographers are just cooking Borscht soup at the kitchen. Each of them has brought typical ingredients for Borscht soup such as carrots and onions. They might be paying for the ingredients, or some of them might be funded by NFSA (National Fictional Security Agency). The three cryptographers respect each other's ideology to have a relation to NFSA, but they wonder if they eat food funded by NFSA. All they have in the kitchen are the ingredients (namely, carrots and onions) and saucepans with Borscht soup. Then, they decide to resolve their uncertainty by conducting a secure AND computation with the ingredients and saucepans, to make sure whether they all paid or not. We call this the *Cooking Cryptographers Problem*, which is named in honor of the *Dining Cryptographers Problem* [1].

The accepted version of the article is available online at https://doi.ieeecomputersociety.org/10.1109/CSF51468.2021.00034

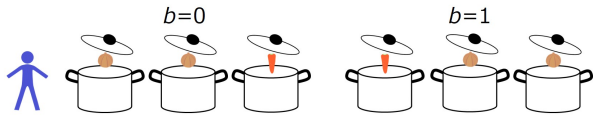
A. Cooking Cryptographers

For simplicity, let us consider the Cooking Cryptographers Problem with two players. Assume that Alice and Bob have private inputs $a, b \in \{0, 1\}$, respectively (the individual input is 0 if he/she was supported by NFSA; otherwise, it is 1). Our goal is to compute the two-input logical AND function $f(a, b) = a \wedge b$ without revealing any information except for the output value. Remembering that they are in the kitchen, let us construct a secure protocol for this function using cooking tools and ingredients. Interestingly, we can give an example of the computation using two carrots, four onions, and three saucepans, as follows.

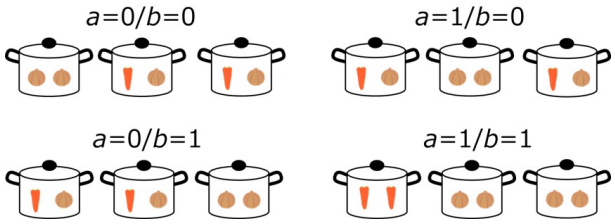
1) There are three saucepans filled with Borscht soup (containing no ingredient yet) on kitchen countertops as well as two carrots and four onions on a tray. Each of Alice and Bob picks one carrot and two onions from the tray.
2) If $a = 0$, Alice puts the carrot into the second saucepan and the onions into the other saucepans privately (so that Bob cannot learn which saucepan contains the carrot). If $a = 1$, she puts the carrot into the first saucepan and the onions into the other saucepans.



The ingredients go to the bottom so that nobody sees them directly in any saucepans.
3) If $b = 0$, Bob privately puts the carrot into the third saucepan and the onions into the other saucepans. If $b = 1$, he puts the carrot into the first saucepan and the onions into the other saucepans.



Note that the two carrots are in the same saucepan if and only if $a = b = 1$.



4) Alice and Bob shuffle the three saucepans so that the resulting order of the three saucepans becomes unknown to them.
5) After simmering the Borscht soup in the three saucepans, they enjoy eating the cooked Borscht soup in the three saucepans; if there is a saucepan of Borscht soup only with carrots, we have $f(a,b) = a \wedge b = 1$ (meaning that none of Alice and Bob was supported by NFSA); otherwise, $a \wedge b = 0$.

B. Contribution

In this paper, we formalize the above two-party protocol with a general setting. Because it is not so easy or realistic to cook Borscht soup whenever people want to perform a secure computation, let us replace ingredients and saucepans (with Borscht soup) by colored balls (such as red and white balls) and non-transparent bags, respectively. Assume that a bag possibly includes balls but the colors of the balls are invisible from the outside. Balls and bags are easy to prepare, and they are also familiar tools for learning Probability in high school. Therefore, basing on balls and bags will be more human-friendly than cooking soup for secure computations.

Then, our problem is reformulated as: *Can we realize secure multiparty computations (MPCs) [2] with such balls and bags?* As will be reviewed in Section I-D, there are MPC protocols with familiar tools, such as a deck of physical cards [3]–[5]. Unlike these existing protocols, we attempt to construct the first protocol using colored balls and bags together with simple actions: putting balls into a bag, shuffling the order of bags, and taking balls from a bag. Notice that, similar to ingredients in Borscht soup, balls in a bag have an interesting property that a collection of balls automatically becomes disordered once they are put into a bag, namely *automatic shuffle*.[i]

We positively answer the above question; we construct simple protocols to establish MPCs of the logical AND function with more than two inputs as well as general MPCs. We also give a formal treatment for our protocols and their security (namely, *ball-based cryptography*). Proposed ball-based protocols are described in both formal and informal ways. To easily confirm correctness and security of a protocol, we construct a diagram showing probability traces of the protocol, which was proposed in [6] for card-based protocols. We believe that ball-based cryptography is a realization of usable security that helps people with understanding the principles of MPCs.

[i]Although we believe that this automatic shuffle works well, shaking a bag after putting balls into it would be another way to achieve this.

C. Outline

The remainder of this paper is organized as follows. In Section II, we propose ball-based cryptography by presenting a formal treatment of protocols using balls and bags. In Section III, we show a pseudocode of the AND protocol with two inputs introduced in Section I-A, and then show a diagram of the protocol, which implies correctness and security of the protocol. In Section IV, we extend the two-input AND protocol to an AND protocol with more than two inputs. In Section V, we further extend the protocols to design general MPCs. In Section VI, we discuss the efficiency of our AND protocols. In Section VII, we show implementation examples for ball-based cryptography. We conclude this work in Section VIII. In Appendix A, we show a pseudocode of card-based AND protocol with five cards for comparison. In Appendix B, we introduce different kinds of balls to have efficient protocols.

D. Related Work

Physical objects enable us to achieve cryptographic tasks, such as MPCs [7], zero-knowledge proofs [8], polling [9], and visual secret sharing [10]. As we perform these secure protocols by hand, their principles can be intuitively understood; hence, they are attractive. There are several researches on this subject (called real-life hands-on cryptography), such as a deck of playing cards [3]–[5] (known as *card-based cryptography*), visual secret sharing sheets [11], coins [12], and a PEZ dispenser [13], [14]. Real-life hands-on cryptography has been attracting many young people to the field of security and privacy.

Physical objects are powerful so that they provide solutions to problems in cryptographic protocols (which cannot be solved with only computational assumptions). Using tamper-evident seals [15], we can implement standard cryptographic protocols such as coin flipping and oblivious transfer which are universally composable. Envelopes [16] can be used to implement collusion-free protocols. The use of a ballot box [17] was considered to perform rational secure computations.

Compared with these researches, our work employs the property of bags (namely, *automatic shuffle*) and utilize simple actions with colored balls and bags for performing MPCs. Related to probability theory, taking a ball out of a bag in ball-based cryptography can be a variant of an *urn problem* by regarding a bag as an urn. An urn problem often appears in probability theory and statistics, where a player takes one or more balls from an urn containing some balls. Some examples of urn problems such as binomial distribution are known. MPCs would be included in the collection of urn problems due to our work.

E. Comparison with Card-Based Cryptography

Among related work introduced above, card-based cryptography is the most famous topic on real-life hands-on cryptography. We note that card-based cryptography and ball-based cryptography are different; a state of balls in a bag is denoted by a multiset (as will be seen in Section II-A) while a playing card is defined as a fraction to represent two states of face-down and face-up [18].

Let us discuss the relation between them. Although it is true that a bag containing one ball can be regarded as a face-down card, balls and bags cannot be used to implement any card-based protocol. This is because, with balls and bags, it seems relatively difficult to realize a cyclic shuffle, which is used in a large number of card-based protocols to cyclically shuffle a sequence of cards. On the other hand, any ball-based protocol can be implemented by using a deck of playing cards because a bag containing balls can be represented by a set of face-down cards. That is, we can regard a pile of cards as a bag containing balls; by completely shuffling the pile of cards, the property of being disordered is guaranteed. For instance, the above two-party protocol can be implemented with two red cards and four white ones. However, such an implementation is no longer efficient because the famous five-card trick proposed by Den Boer [3] requires only five cards; hence, the above two-party protocol cannot be derived from card-based protocols and the protocol is non-trivial. From the above discussion, balls and bags approach relies on a weaker assumption than card-based cryptography. We believe that this is an advantage of our approach because even such a weaker setting provides us a simple way of performing secure computations.

We emphasize that practical usability in ball-based cryptography is as high as in card-based cryptography. Let us compare the above two-party protocol with balls and bags and the five-card trick [3], which is the most simple card-based protocol for computing the logical AND. As will be shown in Protocols 1 and 4, the length of the pseudocode of our proposed protocol is almost the same as that of the five-card trick [3]. Indeed, both protocols proceed in a similar way, i.e., two players privately input balls or cards, shuffle a sequence of bags or cards, and reveal all the colors of the balls or cards. Therefore, executing our proposed protocol is easy for laypeople, and it is a beautiful tool of pedagogical significance as is the five-card trick (a case study of using card-based cryptography in a university lecture was reported in [19]). Moreover, physical assumptions they use are general in daily life; the colors of balls in a bag are assumed to be invisible in ball-based cryptography while the colors of face-down cards are assumed to be invisible in card-based cryptography.

Overall, ball-based cryptography will open a new vista, and we expect that it will contribute to increasing people who are interested in computer security and privacy.

II. Formalizing Protocols Based on Balls and Bags

In this section, we present a formal treatment of protocols based on balls and bags by constructing a model of ball-based secure computations. We also discuss *active security* of our proposed scheme, i.e., how we should deal with malicious players who may deviate arbitrary from a protocol.

A. Notations

Remember the AND protocol (which computes $f(a,b) = a \wedge b$) proposed in Section I-A. We now replace ingredients and saucepans with balls and bags, respectively. That is, we use two red balls, four white balls, and three bags as illustrated in Fig. 1; this is an example of executing our AND protocol using balls and bags when $a = b = 1$.

Let $\bullet$ and $\circ$ denote a red ball and a white ball, respectively. Assume that all balls are indistinguishable except for their colors. Seeing Fig. 1, notice that we have to consider two kinds of multisets of balls, namely a "bag" and a "tray." Thus, we introduce two expressions of a multiset of balls: $\{\!\{\cdot\}\!\}$ is *an invisible multiset* representing a bag possibly containing balls, and $[\cdot]$ is *a visible multiset* representing a tray. We use "invisible multiset" and "bag" interchangeably; we call an invisible multiset of balls $\{\!\{\mathsf{b}_1, \ldots, \mathsf{b}_\ell\}\!\}$ a *bag* (into which balls are put) where $\mathsf{b}_1, \ldots, \mathsf{b}_\ell \in \{\bullet, \circ\}$ for a natural number ℓ. Similarly, we call an visible multiset of balls $[\mathsf{b}_1, \ldots, \mathsf{b}_\ell]$ a *tray* (on which balls are put).

We assume that the size of bags is suitable so that once balls are put into a bag, the order of the balls automatically becomes disordered. We call this property *automatically shuffle*. For example, $\{\!\{\bullet, \circ, \bullet\}\!\}$ and $\{\!\{\bullet, \bullet, \circ\}\!\}$ are indistinguishable to players. In this paper, we write red balls first in an (in)visible multiset to unify the notation. It is interesting that constructing MPCs is possible even with balls and bags having such a property.

Using these notations, at the beginning of the two-party protocol proposed in Section I-A, we have three empty bags (invisible multisets) $\{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}$ and a tray (visible multiset) $[\bullet, \bullet, \circ, \circ, \circ, \circ]$ on a table. Let us describe it with a tuple:

$$([\bullet, \bullet, \circ, \circ, \circ, \circ]; \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [], [], []). \tag{1}$$

We call this tuple a *configuration*. The first part of the configuration (before the first semi-colon) is a multiset of available balls, its second part (between the first and second semi-colons) is a sequence of bags, and its last part is a sequence of trays keeping balls picked from the bags. The last part may be omitted if there is no visible ball at that time.

Let us review the two-player protocol with these notations. At Step 2, Alice holds one red ball and two white balls, i.e., $[\bullet, \circ, \circ]$, and puts them into the three bags depending on the value of a; the above configuration is transformed into as follows:

$$([\bullet, \bullet, \circ, \circ, \circ, \circ]; \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [], [], []) \rightarrow \begin{cases} ([\bullet, \circ, \circ]; \{\!\{\circ\}\!\}, \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\}; [], [], []), & \text{if } a = 0, \\ ([\bullet, \circ, \circ]; \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\}, \{\!\{\circ\}\!\}; [], [], []), & \text{if } a = 1. \end{cases} \tag{2}$$

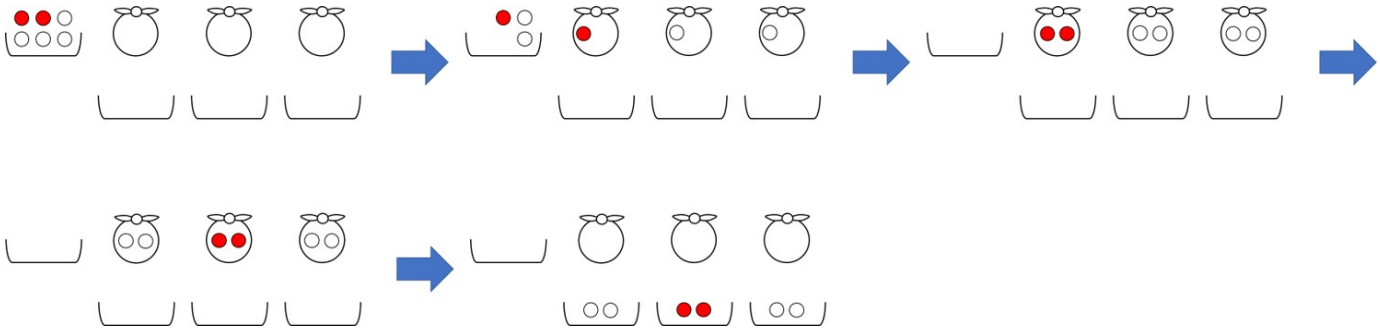


Fig. 1: An example of executing our ball-based AND protocol with two inputs (when $a = b = 1$)

At Step 3, Bob also puts balls depending on b; there are four possibilities:

$$\begin{aligned}
&([]; \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}; [], [], []), && \text{if } (a,b) = (0,0),\\
&([]; \{\!\{\bullet, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [], [], []), && \text{if } (a,b) = (0,1),\\
&([]; \{\!\{\bullet, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}; [], [], []), && \text{if } (a,b) = (1,0),\\
&([]; \{\!\{\bullet, \bullet\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [], [], []), && \text{if } (a,b) = (1,1).
\end{aligned} \tag{3}$$

Let p_{ij} for $i, j \in \{0,1\}$ represent the probability that the input (a,b) is (i,j). Then, the first line in (3) occurs with a probability of p_{00}, the second line occurs with a probability of p_{01}, and so on. Therefore, we now denote the above status (3) by

$$\begin{aligned}
&([]; \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}; [], [], []) && (p_{00}, 0, 0, 0),\\
&([]; \{\!\{\bullet, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [], [], []) && (0, p_{01}, 0, 0),\\
&([]; \{\!\{\bullet, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}; [], [], []) && (0, 0, p_{10}, 0),\\
&([]; \{\!\{\bullet, \bullet\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [], [], []) && (0, 0, 0, p_{11}),
\end{aligned} \tag{4}$$

where a 4-tuple $(q_{00}, q_{01}, q_{10}, q_{11})$ on the right side means that q_{ij} is the conditional probability that $(a,b) = (i,j)$ and the current configuration is the left one.

Next, at Step 4, the three bags are shuffled; the resulting configurations will be:

$$\begin{aligned}
&([]; \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}; [], [], []) && (\tfrac{p_{00}}{3}, \tfrac{p_{01}}{3}, \tfrac{p_{10}}{3}, 0),\\
&([]; \{\!\{\bullet, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [], [], []) && (\tfrac{p_{00}}{3}, \tfrac{p_{01}}{3}, \tfrac{p_{10}}{3}, 0),\\
&([]; \{\!\{\bullet, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}; [], [], []) && (\tfrac{p_{00}}{3}, \tfrac{p_{01}}{3}, \tfrac{p_{10}}{3}, 0),\\
&([]; \{\!\{\bullet, \bullet\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [], [], []) && (0, 0, 0, \tfrac{p_{11}}{3}),\\
&([]; \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \bullet\}\!\}, \{\!\{\circ, \circ\}\!\}; [], [], []) && (0, 0, 0, \tfrac{p_{11}}{3}),\\
&([]; \{\!\{\circ, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \bullet\}\!\}; [], [], []) && (0, 0, 0, \tfrac{p_{11}}{3}).
\end{aligned} \tag{5}$$

Finally, Alice and Bob take all the balls out of the three bags; this is denoted by the last part of the configurations, namely visible multisets, as follows:

$$\begin{aligned}
&([], \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [\circ, \circ], [\bullet, \circ], [\bullet, \circ]) && \left(\tfrac{p_{00}}{p_0}, \tfrac{p_{01}}{p_0}, \tfrac{p_{10}}{p_0}, 0\right),\\
&([], \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [\bullet, \circ], [\bullet, \circ], [\circ, \circ]) && \left(\tfrac{p_{00}}{p_0}, \tfrac{p_{01}}{p_0}, \tfrac{p_{10}}{p_0}, 0\right),\\
&([], \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [\bullet, \circ], [\circ, \circ], [\bullet, \circ]) && \left(\tfrac{p_{00}}{p_0}, \tfrac{p_{01}}{p_0}, \tfrac{p_{10}}{p_0}, 0\right),\\
&([], \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [\bullet, \bullet], [\circ, \circ], [\circ, \circ]) && (0, 0, 0, 1),\\
&([], \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [\circ, \circ], [\bullet, \bullet], [\circ, \circ]) && (0, 0, 0, 1),\\
&([], \{\!\{\}\!\}, \{\!\{\}\!\}, \{\!\{\}\!\}; [\circ, \circ], [\circ, \circ], [\bullet, \bullet]) && (0, 0, 0, 1),
\end{aligned} \tag{6}$$

where $p_0 = p_{00} + p_{01} + p_{10}$. The last part of the configurations, i.e., balls picked from each bag, tells us that the AND value $a \wedge b$ is obtained depending on whether two red balls are taken from the same bag or not and that no information except for the output is leaked (see Section III for more rigorous discussion).

B. Definition of Protocols

Let us introduce a formal definition of protocols based on balls and bags.

Before going into the details, we first introduce extended operations for a sequence of multisets. Let $\mathbf{X} = (X_1, \ldots, X_k)$ and $\mathbf{Y} = (Y_1, \ldots, Y_k)$ be sequences of multisets of the same length k. We define the following two operations: $\mathbf{X} \cup \mathbf{Y} \coloneqq (X_1 \cup Y_1, \ldots, X_k \cup Y_k)$ and $\mathsf{union}(\mathbf{X}) \coloneqq X_1 \cup \cdots \cup X_k$.

Next, we formally define a *configuration* (examples of which were seen in Section II-A).

Definition 1 (Configuration): Let D be a multiset of balls and $k \geq 1$ be an integer (representing the number of bags). We call a triple $(T_0; \mathbf{B}; \mathbf{T}) = (T_0; B_1, \ldots, B_k; T_1, \ldots, T_k)$ a *configuration* if it satisfies the following:

- $T_0 \subseteq D$ is a tray, where all balls in D are put here before the execution of a protocol;
- $\mathbf{B} = (B_1, \ldots, B_k) \subseteq D^k$ is a sequence of k bags;
- $\mathbf{T} = (T_1, \ldots, T_k) \subseteq D^k$ is a sequence of k trays representing that balls in T_i were taken out of B_i for every i, $1 \leq i \leq k$;
- $T_0 \cup \mathsf{union}(\mathbf{B} \cup \mathbf{T}) = D$.

We denote by $C^{(D,k)}$ the set of all configurations derived by fixing D and k.

Given a configuration $(T_0; B_1, \ldots, B_k; T_1, \ldots, T_k)$, balls in trays T_0 and $T_1, \ldots, T_k$ are visible while balls in bags $B_1, \ldots, B_k$ are invisible. We assume that the number of balls inside each bag is known to the public. Bearing this in mind, we define a *visible configuration* $\mathsf{vis}(C)$ for a configuration $C = (T_0; B_1, \ldots, B_k; T_1, \ldots, T_k)$ as follows: $\mathsf{vis}(C) \coloneqq (T_0; |B_1|, \ldots, |B_k|; T_1, \ldots, T_k)$, where $|B_i|$ denotes the number of elements (balls) in B_i. We also define the set of all visible configurations as $\mathsf{Vis}^{(D,k)} = \left\{\mathsf{vis}(C) \mid C \in C^{(D,k)}\right\}$.[ii]

[ii] Information about a configuration transition (e.g., a red ball was moved to the second bag) can be captured by a visible "configuration-trace" that will be mentioned later.

We are now ready to formally define a "protocol" $\mathcal{P}$ achieving MPCs using balls and bags.

Definition 2 (Protocol): A *protocol* $\mathcal{P}$ is a tuple (D, k, n, U, Q, A) satisfying the following:

- D is a multiset over $\{\bullet, \circ\}$, representing balls used in the protocol and $k \geq 1$ is the number of bags; therefore, the initial configuration is $C^0 = (D; \mathbf{B}^0; \mathbf{T}^0)$ such that $\mathsf{union}(\mathbf{B}^0 \cup \mathbf{T}^0) = \phi$ and $|\mathbf{B}^0| = |\mathbf{T}^0| = k$.
- $n \geq 2$ represents the number of players participating in the protocol.
- U is the set of players' possible inputs. In the sequel, we fix it to $U = \{0,1\}^n$, meaning that each player's input is a bit.
- Q is a set of states with two distinguished states, namely, the initial state q_0 and the final state q_f.
- $A : \left(Q \setminus \{q_f\}\right) \times \mathsf{Vis}^{(D,k)} \to Q \times \mathsf{Action}$ is an action function, which specifies the next state and an action, given a current state and a visible configuration. The set Action includes the following actions, where we describe each action for a configuration $C = (T_0; \mathbf{B}; \mathbf{T}) = (T_0; B_1, \ldots, B_k; T_1, \ldots, T_k)$.
  - $(\mathsf{PublicPut}, \mathsf{b}, p)$ for $\mathsf{b} \in T_0$ and $p \in \{1, 2, \ldots, k\}$: This puts the ball b from the tray T_0 into the p-th bag B_p publicly (i.e., the color of the ball is known to all players). That is, it transforms C into the following configuration C':

    $$C' = (T_0 \setminus [\mathsf{b}]; B_1, \ldots, B_{p-1}, B_p \cup \{\!\{\mathsf{b}\}\!\}, B_{p+1}, \ldots, B_k; \mathbf{T}).$$

    Note that the player executing this action must show the ball b to other players before putting it into the bag.
  - $(\mathsf{PrivatePut}, i, \mathbf{I}_0, \mathbf{I}_1)$ for i, $1 \leq i \leq n$, and sequences of k multisets $\mathbf{I}_0 = (I_0^1, \ldots, I_0^k)$ and $\mathbf{I}_1 = (I_1^1, \ldots, I_1^k)$ such that $\mathsf{union}(\mathbf{I}_0) = \mathsf{union}(\mathbf{I}_1) \subseteq T_0$ and $|I_0^j| = |I_1^j|$ for every j, $1 \leq j \leq k$: This makes the i-th player holding an input $x_i \in \{0,1\}$ take balls from T_0 and then privately put them into $\mathbf{B}$ as specified by $\mathbf{I}_{x_i}$. That is, it transforms C into the following configuration C':

    $$C' = (T_0 \setminus \mathsf{union}(\mathbf{I}_{x_i}); \mathbf{B} \cup \mathbf{I}_{x_i}; \mathbf{T}).$$

    Because $\mathsf{union}(\mathbf{I}_0) = \mathsf{union}(\mathbf{I}_1)$, the numbers of $\bullet$ and $\circ$ in $\mathbf{I}_0$ and $\mathbf{I}_1$ are the same.
  - (Shuf, R) for $R \subseteq \{1, 2, \ldots, k\}$: This shuffles bags specified by R so that the resulting order of the bags becomes unknown to all players. That is, it transforms C into the following configuration C':

    $$C' = \left(T_0; B_{\pi^{-1}(1)}, B_{\pi^{-1}(2)}, \ldots, B_{\pi^{-1}(k)}; \mathbf{T}\right),$$

    where π is uniformly drawn at random from the set of all permutations such that all positions except for R are fixed points (i.e., $\pi(i) = i$ for any $i \notin R$).
  - (Take, p) for $p \in \{1, 2, \ldots, k\}$: This takes a ball out of the p-th bag B_p and then the ball is put on the tray T_p. That is, it transforms C into the following configuration C':

    $$C' := (T_0; \mathbf{B}'; \mathbf{T}'), \text{ where}$$
    $$\mathbf{B}' = (B_1, \ldots, B_{p-1}, B_p \setminus \{\!\{\mathsf{b}\}\!\}, B_{p+1}, \ldots, B_k), \text{ and}$$
    $$\mathbf{T}' = (T_1, \ldots, T_{p-1}, T_p \cup [\mathsf{b}], T_{p+1}, \ldots, T_k),$$

    for a (taken) ball $\mathsf{b} \in B_p$. Note that the taken ball was drawn uniformly at random from B_p. Also note that, during this action, no player can get information about other balls in B_p. If we take all balls in all bags, we write this action as $(\mathsf{TakeAll})$.
  - $(\mathsf{Back}, \mathsf{b}, p)$ for $\mathsf{b} \in T_p$ and $p \in \{1, 2, \ldots, k\}$: This puts the ball b on the p-th tray T_p back to the tray T_0. That is, it transforms C into the following configuration C':

    $$C' = (T_0 \cup [\mathsf{b}]; \mathbf{B}; T_1, \ldots, T_{p-1}, T_p \setminus [\mathsf{b}], T_{p+1}, \ldots, T_k).$$

    If we put all balls on all trays back to T_0, we write this action as $(\mathsf{BackAll})$.
  - $(\mathsf{MergeBags}, p_1, p_2)$ for $p_1, p_2 \in \{1, 2, \ldots, k\}$: This merges the p_1-th bag with the p_2-th bag, i.e., all balls in B_{p_1} are moved into B_{p_2} without revealing the colors of the balls. That is, it transforms C into the following configuration C':

    $$C' = (T_0; B_1, \ldots, B_{p_1-1}, \{\!\{\}\!\}, \ldots, B_{p_2} \cup B_{p_1}, \ldots, B_k; \mathbf{T}).$$

  - (Return, e) for some expression e. This special action indicates that the protocol terminates with the output e.

Given inputs $x = (x_1, \ldots, x_n) \in U = \{0,1\}^n$, a protocol $\mathcal{P} = (D, k, n, U, Q, A)$ proceeds as follows. Balls and bags corresponding to D and k, respectively, are on the table, i.e., the initial configuration is $C^0 = (D; \mathbf{B}^0; \mathbf{T}^0)$ where $\mathsf{union}(\mathbf{B}^0 \cup \mathbf{T}^0) = \phi$ and $|\mathbf{B}^0| = |\mathbf{T}^0| = k$. Each i-th player privately holds an input $x_i \in \{0,1\}$, and the state of the protocol is $q_0 \in Q$. Then, the configuration and the state are transformed according to the output of the action function $A(q_0, \mathsf{vis}(C_0))$. The protocol continues to apply the action function A with its current configuration and state being transformed until the state becomes q_f; it terminates with (Return, e).

Let f be a Boolean function such that $f : \{0,1\}^n \to \{0,1\}$. We say that a protocol $\mathcal{P}$ is *correct* for f if e (which is an output of Return) derived by executing $\mathcal{P}$ is always equivalent to the value of $f(x_1, \ldots, x_n)$.

C. Security

Let us consider *security* of a protocol $\mathcal{P}$. We first consider that all players in $\mathcal{P}$ are *semi-honest*, i.e., they correctly follow a protocol's description, but attempt to learn additional information from information derived during the execution of $\mathcal{P}$. In this case, the security of $\mathcal{P}$ intuitively means that no information except for the output is leaked from balls taken out of bags.

For the definition of the security, let us mention some terms. Consider an execution of a protocol $\mathcal{P}$; the enumeration

$(\mathsf{vis}(C_0), \mathsf{vis}(C_1), \ldots, \mathsf{vis}(C_t))$ of all visible configurations from the initial to the final one (where C_{i-1} is transformed into C_i by an action) is called a *visible configuration-trace* (of $\mathcal{P}$).

Definition 3 (Security): Let $\mathcal{P} = (D, k, n, U, Q, A)$ be a protocol that is correct for f. Let V be the random variable representing the visible configuration-trace of $\mathcal{P}$, M be the random variables representing the inputs of $\mathcal{P}$, and F be the random variable representing the output of f. We say that $\mathcal{P}$ is *secure* for f if it satisfies

$$\Pr[M = x \mid F = 0] = \Pr[M = x \mid V = v, F = 0], \text{ and}$$
$$\Pr[M = x \mid F = 1] = \Pr[M = x \mid V = v, F = 1],$$

for any $x \in U$ and visible configuration-trace v.

Next, we consider *malicious* players who arbitrary deviate from the protocol's description. Note that such malicious players are always able to obtain information about a private input by illegally taking balls from bags immediately after an honest player performs PrivatePut. Another possible attack is to mark bags so that each bag can be identified even after the order of the bags are shuffled via Shuffle. To prevent such illegal actions, every player should observe each other to force malicious players to be semi-honest (and immediately abort the protocol if such attacks are detected). However, malicious players can yet deviate by putting different colors[iii] of balls via PrivatePut that is not specified in $\mathcal{P}$ because it is a private action. Therefore, if we consider malicious players in our model, it intuitively suffices to consider whether or not they can violate correctness and security of $\mathcal{P}$ via PrivatePut.

III. AND Protocol with Two Inputs

In this section, based on the definitions in Section II-B, we present a formal description of our AND protocol with two inputs $(x_1, x_2) \in \{0,1\}^2$ introduced in Sections I-A and II-A. We first review the principle of our AND protocol and then present its description. Finally, a diagram of the AND protocol is given in Fig. 2, from which its correctness and security can be confirmed.

A. Principle and Description

Remember the AND protocol introduced in Sections I-A and II-A. Alice and Bob put • into the (first) bag B_1 if his/her private bit is 1; otherwise, they are supposed to put • into different bags, B_2 and B_3.

More formally, Alice holding x_1 acts by $\big(\mathsf{PrivatePut}, 1, ([\circ], [\bullet], [\circ]), ([\bullet], [\circ], [\circ])\big)$, and the possible configurations will be as in (2). Then, Bob holding x_2 acts by $\big(\mathsf{PrivatePut}, 2, ([\circ], [\circ], [\bullet]), ([\bullet], [\circ], [\circ])\big)$, and the possible configurations will be as in (4). That is, sequences of balls they privately put represent the values of their private inputs. After shuffling the three bags and then taking all balls, they can know that the AND value is 1 if there is $[\bullet, \bullet]$; otherwise, 0. No information about the inputs (beyond the output) is leaked.

[iii] Malicious players cannot put a different number of balls into a bag via PrivatePut because that number is fixed and verifiable by observing the bag as defined in Definition 2.

Protocol 1. The two-input AND protocol: $\big([\bullet, \bullet, \circ, \circ, \circ, \circ], 3, 2, \{0,1\}^2, Q, A\big)$.

1) (PrivatePut, 1, ([∘], [•], [∘]), ([•], [∘], [∘]))
2) (PrivatePut, 2, ([∘], [∘], [•]), ([•], [∘], [∘]))
3) (Shuf, $\{1,2,3\}$)
4) (TakeAll)
5) **if** visible conf. includes $[\bullet, \bullet]$ **then**
6) (Return, $x_1 \wedge x_2 = 1$)
7) **else**
8) (Return, $x_1 \wedge x_2 = 0$)

As seen above, our two-input AND protocol uses six balls and three bags; a formal description of the protocol is shown in Protocol 1. See Table I for the performance of the protocol and Section VI is devoted to the discussion about it (as well as that of our multi-input AND protocol which will be presented in Section IV).

B. Security: A diagram of status transitions

To confirm the correctness and security of the AND protocol against semi-honest players, we construct a diagram in Fig. 2 showing status transitions of the protocol. This method was first proposed in [20] for card-based protocols, and then an extended diagram was proposed in [6], which uses the *probability trace* below.

Definition 4 (Probability Trace): Let $\mathcal{P}$ be a protocol with an input set $U = \{0,1\}^n$, and let v be a visible configuration-trace. We regard every input $x \in U = \{0,1\}^n$ as a decimal number x, $1 \le x \le |U| = 2^n$. A $|U|$-tuple $(q_1, q_2, \ldots, q_{|U|})$ such that $q_x = \Pr[M = x, G_v = C \mid V = v]$ for every $x \in U$ is called a *probability trace* for a configuration C and the visible configuration-trace v, where M, G_v, and V are random variables of the original input, configuration when v is seen, and visible configuration-trace, respectively.

See Fig. 2 again. Each "box" in the figure including several pairs of a configuration and a probability trace is called a *status*. Each status is associated with a prefix of the visible configuration-trace. As stated in Section II-A and formally defined in Definition 4, a probability trace next to a configuration represents the conditional probability that the current configuration is the configuration, given the prefix of the visible sequence-trace.

A status in Fig. 2 is transformed into the next status by an action as follows:

- The topmost status consists of a single pair of the initial configuration C^0 and the probability trace $(p_{00}, p_{01}, p_{10}, p_{11})$.
- The first (and second) action is PrivatePut. For instance, when $x_1 = 0$, three balls specified by $\mathbf{I}_0^1 = ([\circ], [\bullet], [\circ])$ are privately put into the bags, and the probability trace becomes $(p_{00}, p_{01}, 0, 0)$.
- The third action is Shuf. After the action, there are three possible configurations with the equal probability, i.e.,

1/3. This can be seen in the coefficient in the probability traces for the three configurations.

- The fourth action is TakeAll, yielding six visible configurations. Assume for example that we have observed $([\circ,\circ],[\bullet,\circ],[\bullet,\circ])$. The fourth coordinate in the probability trace is 0, and other coordinates are all $\frac{p_{00}+p_{01}+p_{10}}{3}$, and hence, we know that the inputs must not be (1,1). It means that the output is "$x_1 \wedge x_2 = 0$". Furthermore, no information about the inputs is leaked because the distribution of the conditional probability that the inputs are (0,0), (0,1), and (1,0) is the same as the one of knowing $x_1 \wedge x_2$ before the execution of the protocol.

This AND protocol is also secure against malicious players because although malicious Alice (or Bob) could put a red ball into a bag of different position that contradicts to the value of her private input, this attack can be regarded as she inputs the negation of her input. That is, it can be simulated in ideal/real simulation paradigm [21] (although its rigorous proof is omitted).

IV. AND Protocol with More Than Two Inputs

In this section, we deal with secure AND computation with more than two inputs. That is, we present a general AND protocol that securely computes $x_1 \wedge \cdots \wedge x_n$, given that n players $P_1,\ldots,P_n$ hold private input bits $x_1,\ldots,x_n \in \{0,1\}$, respectively.

A. Idea

Simply extending the two-input AND does not work.: As stated in Section III-A, our two-input AND protocol computes the AND value by making two players privately put $\bullet$ into the same bag if and only if their private bits both are 1. If we simply extend this principle, can we construct an n-input AND protocol for any n? Consider for instance that there are three players P_1, P_2, and P_3 where P_i holds his/her private bit $x_i \in \{0,1\}$ for every i, $1 \le i \le 3$. We prepare four (empty) bags B_1, B_2, B_3, and B_4 and make each player P_i privately put $\bullet$ into the (first) bag B_1 if x_i is 1 (otherwise, into the $(i{+}1)$-st bag B_{i+1}) and $\circ$ into the remaining bags. Then, the resulting configurations will be as follows:

$$\begin{array}{ll}
(;\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\};) & (p_{000},0,0,0,0,0,0,0),\\
(;\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\};) & (0,p_{001},0,0,0,0,0,0),\\
(;\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\};) & (0,0,p_{010},0,0,0,0,0),\\
(;\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\};) & (0,0,0,p_{100},0,0,0,0),\\
(;\{\!\{\bullet,\bullet,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\};) & (0,0,0,0,p_{011},0,0,0),\\
(;\{\!\{\bullet,\bullet,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\};) & (0,0,0,0,0,p_{101},0,0),\\
(;\{\!\{\bullet,\bullet,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\bullet,\circ,\circ\}\!\};) & (0,0,0,0,0,0,p_{110},0),\\
(;\{\!\{\bullet,\bullet,\bullet\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\},\{\!\{\circ,\circ,\circ\}\!\};) & (0,0,0,0,0,0,0,p_{111}),
\end{array} \tag{7}$$

where we omit the tray T_0 and empty trays. Therefore, after shuffling the four bags and taking all balls, all the players can know that the AND value is 1 if there is $[\bullet,\bullet,\bullet]$; otherwise, 0. As seen from (7), however, they obtain additional information about the inputs from the number of $\bullet$ in a tray; for example, if there is $[\bullet,\bullet,\circ]$, it means that the number of 1 among the inputs x_1, x_2, and x_3 is two. Therefore, this straightforward extension is not a secure computation of AND.

Our idea: Let us go back to Step 4 in the AND protocol introduced in Section I-A (or the fourth status in Fig. 2). Suppose that we replace the action (TakeAll) with (Take, 1), i.e., taking a ball out of the (first) bag B_1 in (5). There are two possibilities, i.e., $\bullet$ is taken with a probability of 1/3 or $\circ$ is taken with a probability of 2/3.[iv] If it is $\bullet$, the configurations in (5) are transformed into the followings:[v]

$$\begin{array}{ll}
([];\{\!\{\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],[],[]) & (\frac{p_{00}}{2},\frac{p_{01}}{2},\frac{p_{10}}{2},0),\\
([];\{\!\{\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\};[\bullet],[],[]) & (\frac{p_{00}}{2},\frac{p_{01}}{2},\frac{p_{10}}{2},0),\\
([];\{\!\{\bullet\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],[],[]) & (0,0,0,p_{11}).
\end{array} \tag{8}$$

Note that this (Take, 1) action does not leak any information about the inputs because the coordinate-wise sum of the probability traces in (8) is equal to $(p_{00},p_{01},p_{10},p_{11})$, meaning that the (conditional) distribution on inputs does not change. Then, consider that we add the fourth bag and make the third player P_3 act by

$$\Big(\mathsf{PrivatePut},3,x_3,([\circ],[],[],[\bullet,\circ]),([\bullet],[],[],[\circ,\circ])\Big).$$

That is, if $x_3 = 0$, the configurations in (8) are transformed into

$$\begin{array}{ll}
([];\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\};[\bullet],) & (\frac{p_{000}}{2},0,\frac{p_{010}}{2},\frac{p_{100}}{2},0,0,0,0),\\
([];\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\bullet,\circ\}\!\};[\bullet],) & (\frac{p_{000}}{2},0,\frac{p_{010}}{2},\frac{p_{100}}{2},0,0,0,0),\\
([];\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\};[\bullet],) & (0,0,0,0,0,0,p_{110},0).
\end{array}$$

If $x_3 = 1$, the configurations in (8) are transformed into

$$\begin{array}{ll}
([];\{\!\{\bullet,\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],) & (0,\frac{p_{001}}{2},0,0,\frac{p_{011}}{2},\frac{p_{101}}{2},0,0),\\
([];\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],) & (0,\frac{p_{001}}{2},0,0,\frac{p_{011}}{2},\frac{p_{101}}{2},0,0),\\
([];\{\!\{\bullet,\bullet\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],) & (0,0,0,0,0,0,0,p_{111}).
\end{array}$$

As seen from the above configurations, there is $\{\!\{\bullet,\bullet\}\!\}$ if the AND value $x_1 \wedge x_2 \wedge x_3$ is 1; otherwise, the four bags are a permuted sequence of $\{\!\{\bullet,\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\}$. Thus, after (Shuf, $\{1,2,3,4\}$) and (TakeAll), all the players can obtain the AND value of the three inputs $x_1 \wedge x_2 \wedge x_3$ without revealing any information except for the output.

Next, let us consider $n = 4$. In the same way as $n = 3$, assume that we replace the above action (TakeAll) (in the three-input protocol) with (Take, 1) and then a red ball is taken. The resulting configurations will be the followings:

$$\begin{array}{ll}
(;\{\!\{\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],) & (\frac{p_{000}}{3},\frac{p_{001}}{3},\frac{p_{010}}{3},\frac{p_{100}}{3},\frac{p_{011}}{3},\frac{p_{101}}{3},\frac{p_{110}}{3},0),\\
(;\{\!\{\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],) & (\frac{p_{000}}{3},\frac{p_{001}}{3},\frac{p_{010}}{3},\frac{p_{100}}{3},\frac{p_{011}}{3},\frac{p_{101}}{3},\frac{p_{110}}{3},0),\\
(;\{\!\{\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\bullet,\circ\}\!\};[\bullet],) & (\frac{p_{000}}{3},\frac{p_{001}}{3},\frac{p_{010}}{3},\frac{p_{100}}{3},\frac{p_{011}}{3},\frac{p_{101}}{3},\frac{p_{110}}{3},0),\\
(;\{\!\{\bullet\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\},\{\!\{\circ,\circ\}\!\};[\bullet],) & (0,0,0,0,0,0,0,p_{111}).
\end{array} \tag{9}$$

These are the "same" configurations as in (8), i.e., the first bag contains a red ball if and only if $x_1 = x_2 = x_3 = 1$. Thus, a four-input AND protocol can be constructed by making P_4 privately put balls in a similar way to P_3 when $n = 3$.

In this way, we can extend the two-input AND protocol to the n-input one for $n \ge 3$.

[iv] Remember that in Step 3, the three bags have been shuffled. Hence, taking a ball out of B_1 does not leak information about x_1 and x_2.

[v] If it is $\circ$, we return $\circ$ into B_1 and then repeat shuffling bags and taking a ball out of B_1 until $\bullet$ is taken.

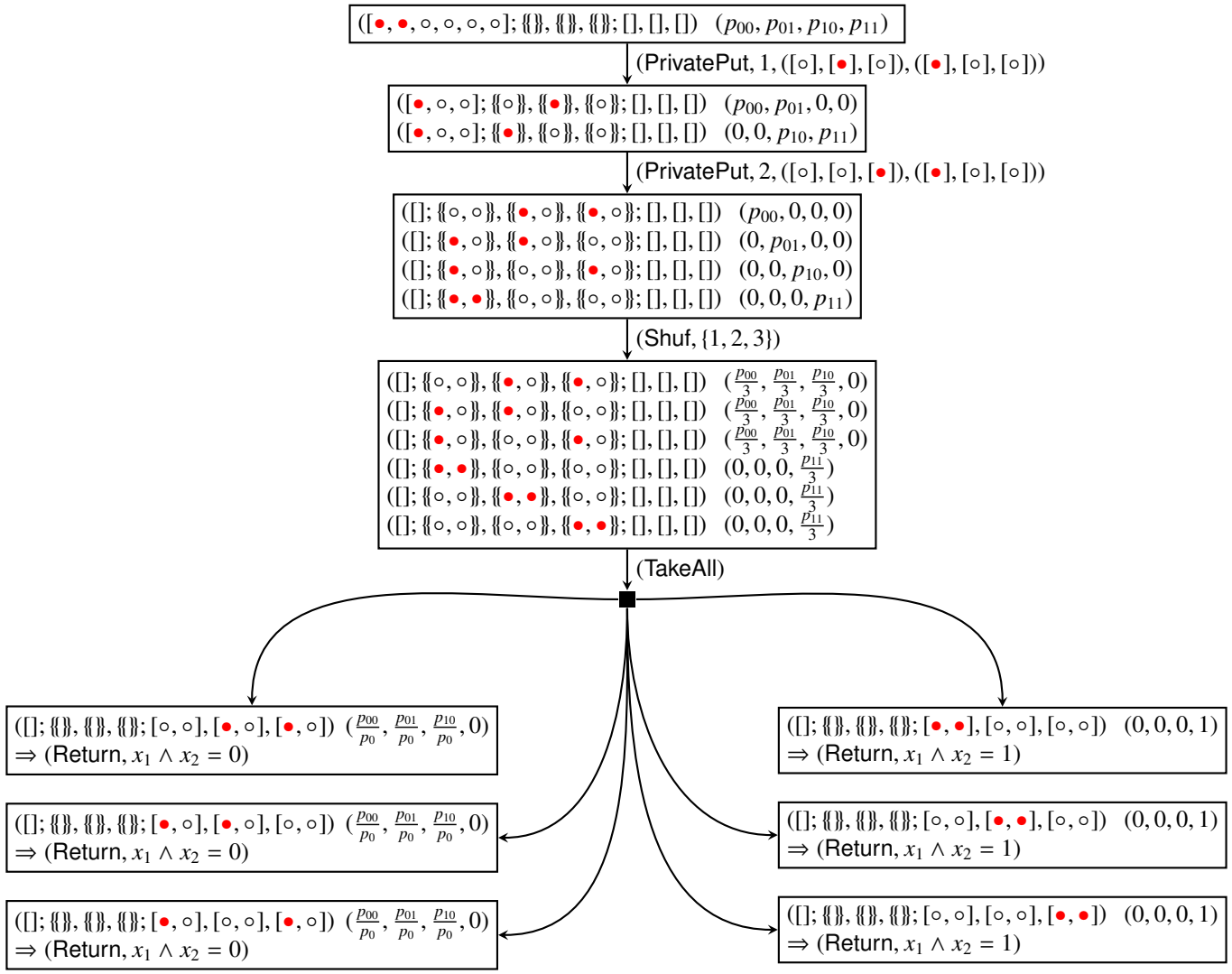


Fig. 2: A diagram of the two-input AND protocol with two inputs $x_1, x_2 \in \{0, 1\}$ introduced in Sections I-A and II-A, where $p_0 = p_{00} + p_{01} + p_{10}$.

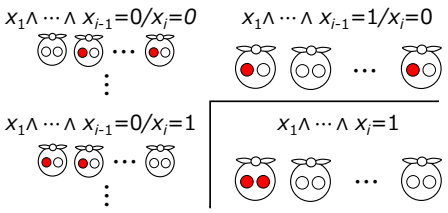
B. Description

Based on the idea explained above, we present an informal description of our AND protocol with $n\,(\geq 3)$ inputs, in which correctness and security are also described.

1) Players prepare $n+1$ empty bags, two red balls, and $2n$ white balls.
2) If $x_1 = 0$, a player P_1 privately puts a red ball into the second bag and a white ball into each of the first and third bags. If $x_1 = 1$, P_1 privately puts a red ball into the first bag and a white ball into each of the second and third bags.
3) If $x_2 = 0$, a player P_2 privately puts a red ball into the third bag and a white ball into each of the first and second bags. If $x_2 = 1$, P_2 privately puts a red ball into the first bag and a white ball into each of the second and third bags.
4) For $i = 3$ to n, repeat the following.
   a) Shuffle the first i bags (which are not empty).
   b) Take a ball from the first bag. If its color is red, then player P_i holds the red ball and other two white balls. If its color is white, then they puts the white ball into the first bag again and go back to the previous step. (Note that since the bags have been shuffled in the previous step, the probability that the color of the taken ball is red is $\frac{1}{n+1}$, which is independent of the input.)
   c) If $x_i = 0$, P_i privately puts the red ball into the $(i+1)$-st bag and the white ball into the first bag. If $x_i = 1$, P_i privately puts the red ball into the first bag and the white ball into the $(i+1)$-st bag. Another white ball is put into the $(i+1)$-st bag. Note that the two red balls are in the first bag if and only if $x_1 \wedge x_2 \wedge \cdots \wedge x_i = 1$.

Protocol 2. The AND protocol with $n\,(\geq 3)$ inputs: $([\bullet, \bullet, \circ, \circ, \ldots, \circ], n+1, n, \{0,1\}^n, Q, A)$.

1) (PrivatePut, $1, \mathbf{I}_0^1, \mathbf{I}_1^1$)
2) (PrivatePut, $2, \mathbf{I}_0^2, \mathbf{I}_1^2$)
3) **for** $i \leftarrow 3$ to n **do**
4) **while**(1)
5) (Shuf, $\{1, 2, \ldots, i\}$)
6) (Take, 1)
7) **if** taken ball = $\bullet$ **then**
8) (Back, $\bullet$, 1)
9) (PrivatePut, $i, \mathbf{I}_0^i, \mathbf{I}_1^i$)
10) **break**
11) **else if** taken ball = $\circ$ **then**
12) (Back, $\circ$, 1)
13) (PublicPut, $\circ$, 1)
14) (Shuf, $\{1, 2, \ldots, n+1\}$)
15) (TakeAll)
16) **if** visible conf. includes $[\bullet, \bullet]$ **then**
17) (Return, $x_1 \wedge \cdots \wedge x_n = 1$)
18) **else**
19) (Return, $x_1 \wedge \cdots \wedge x_n = 0$)



5) Shuffle the $n+1$ bags.
6) Take all the balls from the $n+1$ bags. If there is a bag including two red balls, then we have $x_1 \wedge \cdots \wedge x_n = 1$; otherwise, $x_1 \wedge \cdots \wedge x_n = 0$.

A formal description is shown in Protocol 2. Here,

$$\mathbf{I}_0^1 = \big([\circ], [\bullet], [\circ], [], \ldots, []\big),$$
$$\mathbf{I}_0^2 = \big([\circ], [\circ], [\bullet], [], \ldots, []\big),$$
$$\mathbf{I}_1^1 = \mathbf{I}_1^2 = \big([\bullet], [\circ], [\circ], [], \ldots, []\big),$$

and for every i, $3 \leq i \leq n$,

$$\mathbf{I}_0^i = \big(\overset{1}{[\circ]}, \overset{2}{[]}, \ldots, \overset{i-1}{[]}, \overset{i}{[\bullet, \circ]}, \overset{i+1}{[]}, \ldots, \overset{k}{[]}\big),$$
$$\mathbf{I}_1^i = \big([\bullet], [], \ldots, [], [\circ, \circ], [], \ldots, []\big).$$

C. Correctness and Security

Figure 3 shows a part of the diagram of the n-input AND protocol described in Section IV-A; the partial diagram corresponds to Steps 3–13 in Protocol 2. From this figure, we can see that there is $\{\!\{\bullet, \bullet\}\!\}$ if and only if $x_1 = x_2 = \cdots = x_i = 1$.

We can furthermore see that (Take, 1) leaks no information about the inputs because the probability of taking a red ball is always $\frac{1}{i+1}$, which is independent of the inputs. Therefore, this protocol is secure against semi-honest players.

This multi-input AND protocol is also secure against malicious players in the same reason for the two-input AND protocol mentioned in Section III-B. Moreover, the protocol is a *collusion-free* protocol [16], i.e., it prevents malicious players from obtaining additional information by coordinating their actions during the execution of the protocol. This is because players are assumed to be in the same place, and hence, they can observe each other's actions to be performed in a correct manner. That is, actions used in our protocols satisfy observability (as in [16, Theorem 4]). Although colluding players can share information about the colors of balls they put into bags via PrivatePut, it does not help them obtain extra information because the number of red balls in bags is always two in the protocol (when taking balls).

V. Protocols for Any Boolean Function

In this section, we will explain how to realize protocols for any Boolean function with balls and bags. To achieve it, we will propose *committed-format* AND and NOT protocols that are known to be functionally complete. Note that in our AND protocols shown in Sections III and IV, the balls should be revealed to obtain the AND values in the final step. Namely, the previous AND protocols cannot be used as building blocks for a composition of another protocol. As a building block for a composite protocol, the following property called *committed-format* is required: We say a protocol is committed-format if its output can be an input for another (next) protocol without revealing balls themselves.

Note that in the sequel, we will construct committed-format protocols with balls, where sizes of all balls are the same. On the other hand, in Appendix B, we will discuss efficient committed-format AND, NOT, and COPY protocols that can also realize any Boolean function by introducing two kinds of balls with different sizes.

A. Committed-Format AND protocol

Encoding: We encode a Boolean value with two bags, each of which includes the same number of balls, as follows:

$$\begin{aligned} &\{\!\{\circ, \circ, \circ, \ldots, \circ\}\!\}, \{\!\{\bullet, \circ, \circ, \ldots, \circ\}\!\} = 0, \\ &\{\!\{\bullet, \circ, \circ, \ldots, \circ\}\!\}, \{\!\{\circ, \circ, \circ, \ldots, \circ\}\!\} = 1. \end{aligned} \tag{10}$$

That is, a pair of bags where the second (resp. first) bag contains exactly one red ball $\bullet$ represents 0 (resp. 1). A pair of bags representing a bit $x \in \{0,1\}$ according to the above encoding rule is called a *commitment to* x.

Let us replace a Return action in Definition 2 with a Result action for a committed-format protocol: (Result, p_1, p_2) for $p_1, p_2 \in \{1, \ldots, k\}$. This means that the protocol terminates with the commitment consisting of the p_1-th and p_2-th bags.

Idea: Given two commitments to $x_1 \in \{0,1\}$ and $x_2 \in \{0,1\}$, a committed-format AND protocol produces a commitment to $x_1 \wedge x_2$. Let B_1 and B_2 be two bags constituting a commitment to x_1, and B_3 and B_4 be those constituting a commitment to x_2 such that $|B_1| = |B_2| = |B_3| = |B_4| = m$. Suppose that we merge B_3 with B_1 via MergeBags (*i.e.*, B_1 and B_3 become such that $|B_1| = 2m$ and $|B_3| = 0$) and put m

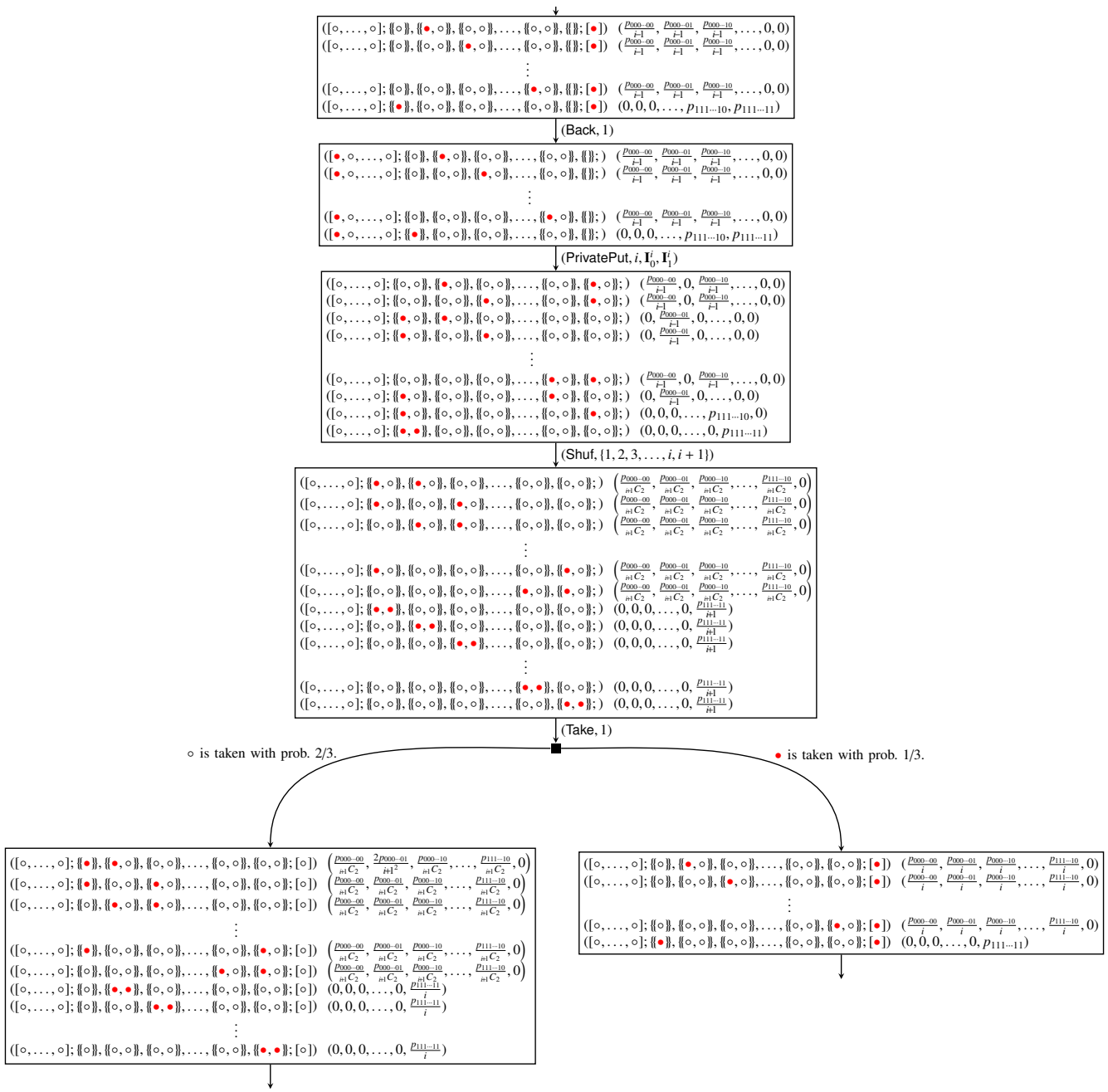


Fig. 3: A crucial part of a diagram of the AND protocol with n inputs $x_1, x_2, \ldots, x_n \in \{0,1\}^n$ for the loop index $i \in \{3, 4, \ldots, n-1\}$, where empty trays and empty bags (over the $(i+2)$-nd) are omitted for simplicity.

white balls into each of B_2 and B_4 (*i.e.*, they become such that $|B_2| = 2m$ and $|B_4| = 2m$). Then, the resulting configurations will be as follows:

$$
\begin{aligned}
&(;\{\!\{\circ,\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\circ,\ldots,\circ\}\!\},\{\!\{\}\!\},\{\!\{\bullet,\circ,\circ,\ldots,\circ\}\!\};) && (p_{00},0,0,0),\\
&(;\{\!\{\bullet,\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\circ,\ldots,\circ\}\!\},\{\!\{\}\!\},\{\!\{\circ,\circ,\circ,\ldots,\circ\}\!\};) && (0,p_{01},0,0),\\
&(;\{\!\{\bullet,\circ,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\circ,\ldots,\circ\}\!\},\{\!\{\}\!\},\{\!\{\bullet,\circ,\circ,\ldots,\circ\}\!\};) && (0,0,p_{10},0),\\
&(;\{\!\{\bullet,\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\circ,\ldots,\circ\}\!\},\{\!\{\}\!\},\{\!\{\circ,\circ,\circ,\ldots,\circ\}\!\};) && (0,0,0,p_{11}),
\end{aligned}
$$

where we omit the tray T_0. Note that these are similar to (4) if we eliminate the empty bag B_3, i.e., two red balls are in B_1 if and only if $x_1 = x_2 = 1$. Therefore, we can compute an AND value via (Shuf, $\{1,2,4\}$) and (TakeAll) in the same way to our two-input AND protocol.

Next, let us consider how to produce a commitment to $x_1 \wedge x_2$ from the above configurations (where B_3 is removed). Suppose that we repeat applying (Shuf, $\{1,2,3\}$) and (Take, 1) until the color of a taken ball is red[vi]. Then, the resulting configurations

[vi] If it is ∘, we return ∘ into B_1.

Protocol 3. The committed-format AND protocol, where the number of balls in each bag is denoted by m: $\big([\circ,\ldots,\circ],4,2,\{0,1\}^2,Q,A\big)$.

Initial state (where empty trays are omitted):

$([\circ,\ldots,\circ];\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\};)$ $(p_{00},0,0,0)$

$([\circ,\ldots,\circ];\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\};)$ $(0,p_{01},0,0)$

$([\circ,\ldots,\circ];\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\};)$ $(0,0,p_{10},0)$

$([\circ,\ldots,\circ];\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\};)$ $(0,0,0,p_{11})$

Steps:

1) (MergeBags, 3, 1)
2) (PublicPut, $\circ\cdots\circ$, 2)
3) (PublicPut, $\circ\cdots\circ$, 4)
4) **while**(1)
5) (Shuf, $\{1,2,4\}$)
6) (Take, 1)
7) **if** taken ball = $\bullet$ **then**
8) **break**
9) **else**
10) (Back, $\circ$, 1)
11) (PublicPut, $\circ$, 1)
12) (MergeBags, 4, 2)
13) (PublicPut, $\circ\cdots\circ$, 1)
14) (Result, 1, 2)

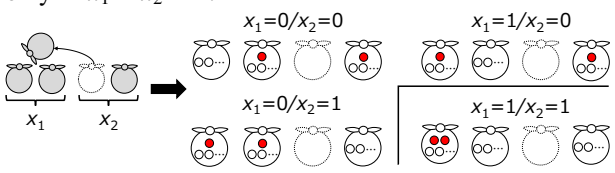
will be as follows:

$$\begin{aligned}
&(;\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\};[\bullet]) \quad (\tfrac{p_{00}}{2},\tfrac{p_{01}}{2},\tfrac{p_{10}}{2},0),\\
&(;\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\bullet,\circ,\ldots,\circ\}\!\};[\bullet]) \quad (\tfrac{p_{00}}{2},\tfrac{p_{01}}{2},\tfrac{p_{10}}{2},0), \quad (11)\\
&(;\{\!\{\bullet,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\},\{\!\{\circ,\circ,\ldots,\circ\}\!\};[\bullet]) \quad (0,0,0,p_{11}),
\end{aligned}$$

where we omit the empty bag and empty trays. From the above configurations, notice that if we merge B_2 and B_3 into one bag, say B_2, then a pair of B_1 and B_2 can be a commitment to $x_1 \wedge x_2$. Thus, we can obtain the output commitment by moving all the balls in B_3 into B_2 (and put $2m$ white balls into B_1 to make the numbers of balls be the same).

Description: Given two commitments to $x_1 \in \{0,1\}$ and $x_2 \in \{0,1\}$, our committed-format AND protocol informally proceeds as follows.

1) Merge the first bag constituting the commitment to x_2 with the first bag constituting the commitment to x_1. Note that there is a bag including two red balls if and only if $x_1 \wedge x_2 = 1$.



2) Put white balls into the second bags constituting the commitments to x_1 and x_2 so that the numbers of balls in those bags become the same as in the merged bag.
3) Shuffle the three bags.
4) Take a ball from the first bag. If its color is red, then merge the third bag with the second bag. Note that the first bag includes a red ball if and only if $x_1 \wedge x_2 = 1$. If its color is white, then put it into the first bag again and go back to the previous step.
5) Put white balls into the first bag so that the number of balls in that bag becomes the same as in the merged bag. The first and second bag constitute a commitment to $x_1 \wedge x_2$.

A formal description is shown in Protocol 3. We denote by B_1 and B_2 two bags constituting a commitment to x_1 and by B_3 and B_4 those constituting a commitment to x_2. Hereinafter, for successive two actions $(\mathsf{PublicPut}, \mathsf{b}_1, p)$ and $(\mathsf{PublicPut}, \mathsf{b}_2, p)$, we simply write $(\mathsf{PublicPut}, \mathsf{b}_1\mathsf{b}_2, p)$.

We omit a diagram of our committed-format AND protocol as it is similar to that of our multi-input AND protocol shown in Fig. 3. We discuss the performance of the protocol in Section VI.

B. How to Construct a Protocol for Any Function

See (10) again. We note that a committed-format NOT protocol can be easily constructed; just swapping two bags constituting a commitment to $x \in \{0,1\}$ results in a commitment to the negation $\bar{x}$.

We now have committed-format AND and NOT protocols as shown above. Based on these protocols, we can construct a protocol for any Boolean function f as follows.

1) Create a Boolean circuit representing f (with AND and NOT gates).
2) Each player prepares a required number of commitments to his/her private input via PrivatePut according to the circuit.
3) Obtain a commitment to the output value of f by evaluating the circuit using our committed-format AND/NOT protocols. If the number of balls is different between input commitments when performing the AND protocol, players put white balls such that the number of balls becomes the same.

Since our committed-format AND protocol is secure and any information about the inputs and output does not leak (because of committed-format), the above protocol for f is also secure against semi-honest players. However, unlike our proposed AND protocols, the above protocol is not secure against malicious players because they possibly perform PrivatePut multiple times, and hence, the protocol cannot ensure that they consistently perform PrivatePut without contradicting to their private inputs. (We solve this security issue in Appendix B by constructing a copy protocol that duplicates a commitment.)

VI. Performance of Our Protocols

This section discusses the efficiency of our AND protocols shown in Sections III, IV, and V-A. Table I summarizes the performance of our proposed protocols. We evaluated them in terms of two items: the number of balls and bags, and runtime.

The Number of Balls and Bags: Our two-input AND protocol requires six balls (namely, two red and four white balls) and three bags. Our conjecture is that two bags would be insufficient for a secure computation of the two-input AND computation. On the other hand, our AND protocol with n

inputs requires $2n + 2$ balls (namely, two red and $2n$ white balls) and $n + 1$ bags. Our committed-format AND protocol would be inefficient because it requires $8m$ balls (and four bags) where m denotes the number of balls in each of bags constituting input commitments.

Let C be a Boolean circuit with the logical AND/NOT gates and $d(C)$ denote the depth of C. Let us consider the number of required balls for our committed-format AND protocol that evaluates the "final" AND gate for C (i.e., it is of the depth $d(C)$), denoted by $\alpha_{d(C)}$. Remember that the number of balls in each of bags constituting the output commitment is four times greater than that of input commitments in our committed-format AND protocol. Thus, we have $\alpha_{d(C)} = 2 \times 4^{d(C)}$, i.e., $\alpha_{d(C)} = O(4^{d(C)})$.

Runtime: Our two-input AND protocol has a deterministic runtime. By contrast, our AND protocol with n inputs and committed-format AND protocol are Las Vegas algorithms. Each protocol includes a repetition of shuffling bags and taking a ball until • appears. Let us estimate the expected number of the repetition.

For our AND protocol with n inputs, let us first consider the case of $n = 3$ as a simple case. Remember that we repeat $(\mathsf{Shuf}, \{1, 2, 3\})$ and $(\mathsf{Take}, 1)$ (and $(\mathsf{PublicPut}, \bullet, 1)$) from (4) until • appears. The expected number of the repetition is three because the probability of taking • is exactly 1/3. Then, let us consider the case of $n = 4$. In addition to the actions for $n = 3$, we repeat $(\mathsf{Shuf}, \{1, 2, 3, 4\})$ and $(\mathsf{Take}, 1)$ from four bags including eight balls (namely, two red and six white balls) until • is taken. Therefore, the expected number of the repetition is $3 + 4 = 7$.

When we have n (≥ 3) inputs, the expected number of the repetition can be written as follows:

$$\sum_{k=3}^{n} k = \frac{(n-2)(n+3)}{2}.$$

That is, it is $O(n^2)$.

For our committed-format AND protocol, let us also show the expected number of the repetition. The probability of taking • (as in (11)) is $2/(6m-2)$ because the total numbers of red and white balls in the bags are 2 and $6m-2$, respectively. Thus, the expected number of the repetition is $3m - 1$. As for the number of balls, let us consider the expected number of the repetition for our committed-format AND protocol that evaluates the final AND gate for C, denoted by $\beta_{d(C)}$. We have $\beta_{d(C)} = 3 \times 4^{d(C)-1} - 1$, i.e., $\beta_{d(C)} = O(4^{d(C)})$.

VII. Implementation Examples

In this section, we show implementation examples for ball-based cryptography to show the feasibility of using (physical) balls and bags. For this, we purchased two-colored balls for approximately \$1 and three bags for approximately \$3 as shown in Fig. 4. Using these balls and bags, we implemented all the actions defined in Definition 2 (except for Back and Result[vii]), as shown in Fig. 5. As a result, we confirmed that ball-based cryptography proposed in this study is based on realistic physical operations and is feasible for humans.

[vii] Back can be performed without using a bag, and Result requires no ball and bag.

Fig. 4: Two-colored balls and three bags we purchased. The bags are closed at the top with drawstrings.

VIII. Conclusion

In this work, we showed that balls and bags provide us a simple way for achieving a secure computation of the logical AND. We formalized protocols with balls and bags and showed how to construct a diagram of a protocol, from which its correctness and security can be confirmed. Moreover, we presented general MPCs by constructing committed-format AND and NOT protocols. In Appendix B, we also proposed *efficient* committed-format protocols using different kinds of balls.

Acknowledgements

We thank the anonymous referees, whose comments have helped us to improve the presentation of the paper. This work was supported in part by JSPS KAKENHI Grant Numbers JP18H05289, JP19J21153, and JP21K11881.

References

[1] D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *Journal of Cryptology*, vol. 1, no. 1, pp. 65–75, 1988. [Online]. Available: https://doi.org/10.1007/BF00206326

[2] A. C. Yao, "Protocols for secure computations," in *IEEE Symposium on Foundations of Computer Science*, ser. FOCS '82. Washington: IEEE Computer Society, 1982, pp. 160–164. [Online]. Available: https://doi.org/10.1109/SFCS.1982.88

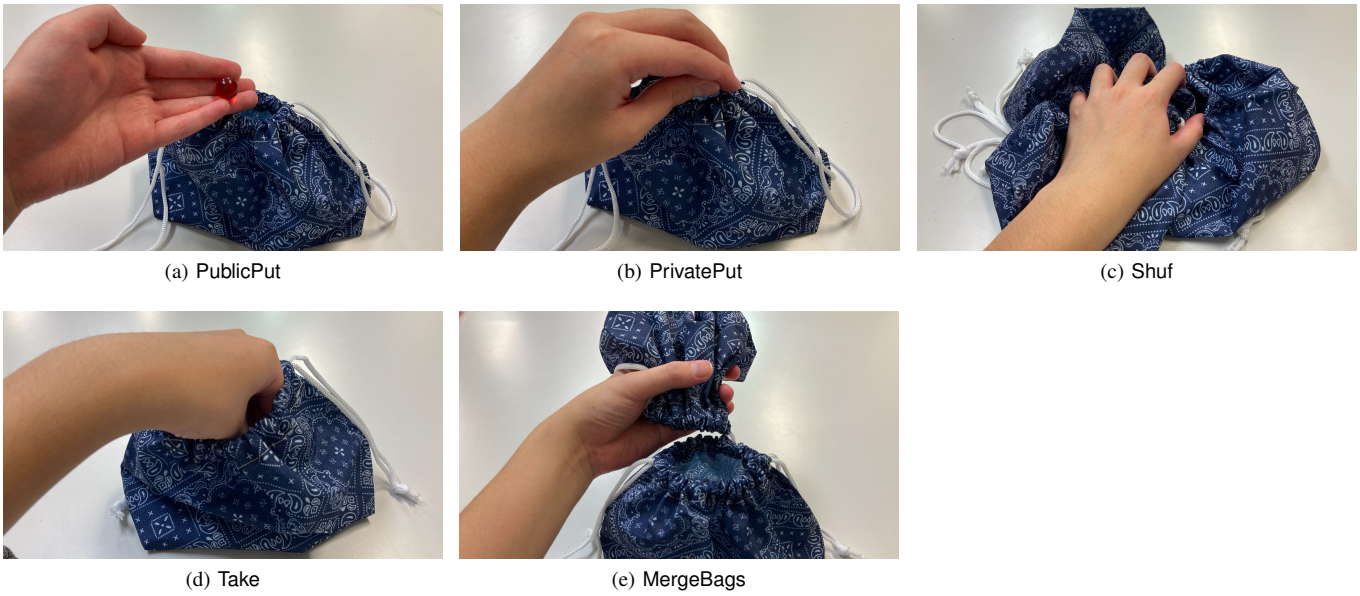
[3] B. Den Boer, "More efficient match-making and satisfiability the five card trick," in *Advances in Cryptology—EUROCRYPT '89*, ser. Lecture Notes in Computer Science, J.-J. Quisquater and J. Vandewalle, Eds., vol. 434. Berlin, Heidelberg: Springer, 1990, pp. 208–217. [Online]. Available: https://doi.org/10.1007/3-540-46885-4_23

[4] C. Crépeau and J. Kilian, "Discreet solitary games," in *Advances in Cryptology—CRYPTO' 93*, ser. Lecture Notes in Computer Science, D. R. Stinson, Ed., vol. 773. Berlin, Heidelberg: Springer, 1994, pp. 319–330. [Online]. Available: https://doi.org/10.1007/3-540-48329-2_27

[5] T. Mizuki, M. Kumamoto, and H. Sone, "The five-card trick can be done with four cards," in *Advances in Cryptology—ASIACRYPT 2012*, ser. Lecture Notes in Computer Science, X. Wang and K. Sako, Eds., vol. 7658. Berlin, Heidelberg: Springer, 2012, pp. 598–606. [Online]. Available: https://doi.org/10.1007/978-3-642-34961-4_36

TABLE I: The efficiency of our proposed protocols, where m denotes the number of balls in each of bags constituting an input commitment.

Function		# of Balls	# of Bags	Runtime
two-input AND	§III	6	3	Deterministic
n-input AND	§IV	$2n+2$	$n+1$	Expected
committed-format AND	§V-A	$8m$	4	Expected
Boolean circuit C	§V-B	$2 \times 4^{d(C)}$	$2 \times$ #input-node	Expected



(a) PublicPut (b) PrivatePut (c) Shuf

(d) Take (e) MergeBags

Fig. 5: Implementing all the actions defined in Definition 2

[6] T. Mizuki and Y. Komano, "Analysis of information leakage due to operative errors in card-based protocols," in *Combinatorial Algorithms*, ser. Lecture Notes in Computer Science, C. Iliopoulos, H. W. Leong, and W.-K. Sung, Eds., vol. 10979. Cham: Springer, 2018, pp. 250–262. [Online]. Available: https://doi.org/10.1007/978-3-319-94667-2_21

[7] R. Fagin, M. Naor, and P. Winkler, "Comparing information without leaking it," *Commun. ACM*, vol. 39, no. 5, pp. 77–85, 1996. [Online]. Available: https://doi.acm.org/10.1145/229459.229469

[8] R. Gradwohl, M. Naor, B. Pinkas, and G. N. Rothblum, "Cryptographic and physical zero-knowledge proof systems for solutions of Sudoku puzzles," *Theory of Computing Systems*, vol. 44, no. 2, pp. 245–268, 2009. [Online]. Available: https://doi.org/10.1007/s00224-008-9119-9

[9] T. Moran and M. Naor, "Polling with physical envelopes: A rigorous analysis of a human-centric protocol," in *Advances in Cryptology—EUROCRYPT 2006*, ser. Lecture Notes in Computer Science, S. Vaudenay, Ed., vol. 4004. Berlin, Heidelberg: Springer, 2006, pp. 88–108. [Online]. Available: https://doi.org/10.1007/11761679_7

[10] M. Naor and A. Shamir, "Visual cryptography," in *Advances in Cryptology—EUROCRYPT '94*, ser. Lecture Notes in Computer Science, A. De Santis, Ed., vol. 950. Berlin, Heidelberg: Springer, 1995, pp. 1–12. [Online]. Available: https://doi.org/10.1007/BFb0053419

[11] P. D'Arco and R. De Prisco, "Secure computation without computers," *Theoretical Computer Science*, vol. 651, pp. 11–36, 2016. [Online]. Available: https://doi.org/10.1016/j.tcs.2016.08.003

[12] Y. Komano and T. Mizuki, "Multi-party computation based on physical coins," in *Theory and Practice of Natural Computing*, ser. Lecture Notes in Computer Science, D. Fagan, C. Martín-Vide, M. O'Neill, and M. A. Vega-Rodríguez, Eds., vol. 11324. Cham: Springer, 2018, pp. 87–98. [Online]. Available: https://doi.org/10.1007/978-3-030-04070-3_7

[13] J. Balogh, J. A. Csirik, Y. Ishai, and E. Kushilevitz, "Private computation using a PEZ dispenser," *Theoretical Computer Science*, vol. 306, no. 1, pp. 69–84, 2003. [Online]. Available: https://doi.org/10.1016/S0304-3975(03)00210-X

[14] Y. Abe, M. Iwamoto, and K. Ohta, "Efficient private PEZ protocols for symmetric functions," in *Theory of Cryptography*, ser. Lecture Notes in Computer Science, D. Hofheinz and A. Rosen, Eds., vol. 11891. Cham: Springer, 2019, pp. 372–392. [Online]. Available: https://doi.org/10.1007/978-3-030-36030-6_15

[15] T. Moran and M. Naor, "Basing cryptographic protocols on tamper-evident seals," *Theoretical Computer Science*, vol. 411, no. 10, pp. 1283 – 1310, 2010. [Online]. Available: https://doi.org/10.1016/j.tcs.2009.10.023

[16] M. Lepinksi, S. Micali, and a. shelat, "Collusion-free protocols," in *ACM Symposium on Theory of Computing*, ser. STOC '05. New York: ACM, 2005, pp. 543–552. [Online]. Available: https://doi.org/10.1145/1060590.1060671

[17] S. Izmalkov, S. Micali, and M. Lepinski, "Rational secure computation and ideal mechanism design," in *IEEE Symposium on Foundations of Computer Science*, ser. FOCS '05. USA: IEEE Computer Society, 2005, pp. 585–594. [Online]. Available: https://doi.org/10.1109/SFCS.2005.64

[18] T. Mizuki and H. Shizuya, "A formalization of card-based cryptographic protocols via abstract machine," *International Journal of Information Security*, vol. 13, no. 1, pp. 15–23, 2014. [Online]. Available: https://doi.org/10.1007/s10207-013-0219-4

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

[20] A. Koch, S. Walzer, and K. Härtel, "Card-based cryptographic protocols using a minimal number of cards," in *Advances in Cryptology—ASIACRYPT 2015*, ser. Lecture Notes in Computer

Protocol 4. The five-card trick [3]

1) $\big(\mathsf{PrivatePut}, 1, (\boxed{\heartsuit}\boxed{\clubsuit}\epsilon\,\epsilon), (\boxed{\clubsuit}\boxed{\heartsuit}\epsilon\,\epsilon)\big)$
2) $\big(\mathsf{PrivatePut}, 2, (\epsilon\,\epsilon\,\epsilon\boxed{\clubsuit}\boxed{\heartsuit}), (\epsilon\,\epsilon\,\epsilon\boxed{\heartsuit}\boxed{\clubsuit})\big)$
3) $\big(\mathsf{PublicPut}, (\epsilon\,\epsilon\boxed{\heartsuit}\epsilon\,\epsilon)\big)$
4) $(\mathsf{Turn}, \{3\})$
5) $(\mathsf{Shuf}, \{\pi \mid \pi = (1\,2\,3\,4\,5)^i, 0 \le i \le 4\})$
6) $(\mathsf{Turn}, \{1,2,3,4,5\})$
7) **if** visible seqence includes $\boxed{\heartsuit}\boxed{\heartsuit}\boxed{\heartsuit}$ **then**
8) $\quad(\mathsf{Return}, x_1 \wedge x_2 = 1)$
9) **else**
10) $\quad(\mathsf{Return}, x_1 \wedge x_2 = 0)$

Science, T. Iwata and J. H. Cheon, Eds., vol. 9452. Berlin, Heidelberg: Springer, 2015, pp. 783–807. [Online]. Available: https://doi.org/10.1007/978-3-662-48797-6_32

[21] O. Goldreich, S. Micali, and A. Wigderson, "How to play ANY mental game," in *ACM Symposium on Theory of Computing*, ser. STOC '87. New York: ACM, 1987, pp. 218–229. [Online]. Available: https://doi.org/10.1145/28395.28420

Appendix A
A Pseudocode of Card-based AND protocol with Five Cards

We show a pseudocode of the five-card trick proposed by Den Boer [3] in Protocol 4, where formal descriptions of actions are derived in [18], and PrivatePut and PublicPut are defined in a similar way to our ball-based model.

Appendix B
Protocols with Different Kinds of Balls

In this section, we discuss efficient committed-format AND/NOT protocols that can also realize any Boolean function by introducing two kinds of balls with different sizes. Moreover, we propose a "COPY" protocol that duplicates an input commitment without revealing any information about the input value.

A. Extended Model

We hereinafter encode a Boolean value with two bags, each of which contains a single ball, as follows:

$$\{\!\{\circ\}\!\}, \{\!\{\bullet\}\!\} = 0, \quad \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\} = 1. \tag{12}$$

Note that this encoding rule is a special version of (10).

Let us introduce two kinds of balls with different sizes. We denote by $\blacksquare$ and $\square$ a red ball and a white ball, respectively, which are bigger than $\bullet$ and $\circ$. We hereinafter call $\{\bullet, \circ\}$ *small balls* and $\{\blacksquare, \square\}$ *big balls*. We assume that we can *distinguish* small balls and big balls in a bag by hands. For example, if we have a bag $\{\!\{\blacksquare, \square, \bullet, \circ\}\!\}$, we can take only big (or small) balls out of the bag.

For the discussion about a protocol with different kinds of balls, let us extend actions in Definition 2 as follows.

- (Take, p, S) for $p \in \{1, 2, \ldots, k\}$ and $S \in \{$"small","big"$\}$: This takes a ball out of the p-th bag, such that the size of the taken ball is specified by S. If the p-th bag includes only balls of the same size, the description of S is omitted. If we take all big or small balls in all bags, we write it as $(\mathsf{TakeAll}, S)$.
- $(\mathsf{MoveBetweenBags}, p_1, p_2, S)$ for $p_1, p_2 \in \{1, 2, \ldots, k\}$ and $S \in \{$"small","big"$\}$: This moves a ball (whose size is specified by S) in the p_1-th bag into the p_2-th bag without revealing the color of the moved ball. If the p_1-th bag includes only balls of the same size, S is omitted.

B. Efficient Committed-Format AND Protocol

In this subsection, we present an AND protocol that produces a commitment to the logical AND of a given input commitments. We call such a protocol a *committed-format* AND protocol. Before describing it, let us mention the idea behind the protocol.

a) Idea: See (11) again. Remember that in our committed-format AND protocol shown in Section V-A, B_2 and B_3 are merged in the final step to produce an output commitment, increasing the number of required (white) balls for the protocol. Let us consider how to make it efficient.

Suppose that given input commitments to $x_1 \in \{0,1\}$ and $x_2 \in \{0,1\}$, we proceed in the same way to the committed-format AND protocol until merging B_3 with B_2. The resulting configurations will be as follows:

$$\begin{aligned}
&(;\{\!\{\circ\}\!\}, \{\!\{\bullet, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [\bullet], [], []) \quad (\tfrac{p_{00}}{2}, \tfrac{p_{01}}{2}, \tfrac{p_{10}}{2}, 0),\\
&(;\{\!\{\circ\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\bullet, \circ\}\!\}; [\bullet], [], []) \quad (\tfrac{p_{00}}{2}, \tfrac{p_{01}}{2}\tfrac{p_{10}}{2}, 0),\\
&(;\{\!\{\bullet\}\!\}, \{\!\{\circ, \circ\}\!\}, \{\!\{\circ, \circ\}\!\}; [\bullet], [], []) \quad (0, 0, 0, p_{11}),
\end{aligned}$$

where empty bags and trays are omitted. Our idea is that *decreasing* the number of white balls in B_2 and B_3 by three yields a commitment to $x_1 \wedge x_2$. To achieve this, we first move balls in B_3 into B_2; the resulting configurations will be as follows:

$$\begin{aligned}
&(;\{\!\{\circ\}\!\}, \{\!\{\bullet, \circ, \circ, \circ\}\!\}; [\bullet], []) \quad (p_{00}, p_{01}, p_{10}, 0),\\
&(;\{\!\{\bullet\}\!\}, \{\!\{\circ, \circ, \circ, \circ\}\!\}; [\bullet], []) \quad (0, 0, 0, p_{11}),
\end{aligned}$$

Then, we put $\blacksquare$ and $\square$ into B_1 and B_2, respectively (and put three $\circ, \circ, \circ$ into B_1). As seen later, these big balls "preserve" the current order of the two bags. The resulting configurations will be as follows:

$$\begin{aligned}
&(;\{\!\{\blacksquare, \circ, \circ, \circ, \circ\}\!\}, \{\!\{\square, \bullet, \circ, \circ, \circ\}\!\}; [\bullet], []) \quad (p_{00}, p_{01}, p_{10}, 0),\\
&(;\{\!\{\blacksquare, \bullet, \circ, \circ, \circ\}\!\}, \{\!\{\square, \circ, \circ, \circ, \circ\}\!\}; [\bullet], []) \quad (0, 0, 0, p_{11}).
\end{aligned}$$

To decrease the number of white balls in B_1 and B_2, we first shuffle B_1 and B_2 (after moving the ball on T_1):

$$\begin{aligned}
&(;\{\!\{\blacksquare, \circ, \circ, \circ, \circ\}\!\}, \{\!\{\square, \bullet, \circ, \circ, \circ\}\!\}; [], []) \quad (\tfrac{p_{00}}{2}, \tfrac{p_{01}}{2}, \tfrac{p_{10}}{2}, 0),\\
&(;\{\!\{\square, \bullet, \circ, \circ, \circ\}\!\}, \{\!\{\blacksquare, \circ, \circ, \circ, \circ\}\!\}; [], []) \quad (\tfrac{p_{00}}{2}, \tfrac{p_{01}}{2}, \tfrac{p_{10}}{2}, 0),\\
&(;\{\!\{\blacksquare, \bullet, \circ, \circ, \circ\}\!\}, \{\!\{\square, \circ, \circ, \circ, \circ\}\!\}; [], []) \quad (0, 0, 0, \tfrac{p_{11}}{2}),\\
&(;\{\!\{\square, \circ, \circ, \circ, \circ\}\!\}, \{\!\{\blacksquare, \bullet, \circ, \circ, \circ\}\!\}; [], []) \quad (0, 0, 0, \tfrac{p_{11}}{2}).
\end{aligned}$$

Then, we can take all small balls without revealing any information about the inputs; if B_1 includes $\bullet$, the resulting configuration will be as follows:

$$\begin{aligned}
&(;\{\!\{\square\}\!\}, \{\!\{\blacksquare\}\!\}; [\bullet, \circ, \circ, \circ], [\circ, \circ, \circ, \circ]) \quad (p_{00}, p_{01}, p_{10}, 0),\\
&(;\{\!\{\blacksquare\}\!\}, \{\!\{\square\}\!\}; [\bullet, \circ, \circ, \circ], [\circ, \circ, \circ, \circ]) \quad (0, 0, 0, p_{11}),
\end{aligned}$$

Now, one can see that B_1 and B_2 is a "commitment" to $x_1 \wedge x_2$ with big balls. Then, we put $\bullet$ and $\circ$ into B_1 and B_2, respectively:

$$(;\{\!\{\square, \bullet\}\!\}, \{\!\{\blacksquare, \circ\}\!\}; [], []) \quad (p_{00}, p_{01}, p_{10}, 0),$$
$$(;\{\!\{\blacksquare, \bullet\}\!\}, \{\!\{\square, \circ\}\!\}; [], []) \quad (0, 0, 0, p_{11}).$$

Now, one can see that the number of white balls has been decreased by three from (8). Then, we shuffle B_1 and B_2:

$$(;\{\!\{\square, \bullet\}\!\}, \{\!\{\blacksquare, \circ\}\!\}; [], []) \quad (\tfrac{p_{00}}{2}, \tfrac{p_{01}}{2}, \tfrac{p_{10}}{2}, 0),$$
$$(;\{\!\{\blacksquare, \circ\}\!\}, \{\!\{\square, \bullet\}\!\}; [], []) \quad (\tfrac{p_{00}}{2}, \tfrac{p_{01}}{2}, \tfrac{p_{10}}{2}, 0),$$
$$(;\{\!\{\blacksquare, \bullet\}\!\}, \{\!\{\square, \circ\}\!\}; [], []) \quad (0, 0, 0, \tfrac{p_{11}}{2}),$$
$$(;\{\!\{\square, \circ\}\!\}, \{\!\{\blacksquare, \bullet\}\!\}; [], []) \quad (0, 0, 0, \tfrac{p_{11}}{2}).$$

As seen from the above configurations, we can obtain a commitment to $x_1 \wedge x_2$ after taking all big balls and rearranging B_1 and B_2 according to the order of the taken big balls.

b) Description: Given two commitments to $x_1 \in \{0, 1\}$ and $x_2 \in \{0, 1\}$, our efficient committed-format AND protocol proceeds as shown in Protocol 5. We denote by B_1 and B_2 two bags constituting a commitment to x_1 and by B_3 and B_4 those constituting a commitment to x_2. Hereinafter, for successive actions $(\mathsf{PublicPut}, \mathsf{b}_1, p)$ and $(\mathsf{PublicPut}, \mathsf{b}_2, p)$, we simply write $(\mathsf{PublicPut}, \mathsf{b}_1\mathsf{b}_2, p)$.

The correctness and security of the protocol can be confirmed from its diagram dipicted in Fig. 6.

Note that in contrast to the committed-format AND protocol shown in Section V-A, our efficient one shown in this subsection always requires 11 (two big and nine small) balls, four bags, and a expected number of four repetitions, i.e., constant.

C. COPY Protocol

In this subsection, we present a COPY protocol. Given a commitment to $x \in \{0, 1\}$, our COPY protocol produces two commitments to x. Remember that we decrease the number of balls by using $\blacksquare$ and $\square$ in the above committed-format AND protocol. Our COPY protocol proceeds in a similar way that we "increase" the number of balls in each bag constituting the input commitment to x.

The correctness and security of our COPY protocol can be confirmed from its diagram. Due to the page length limitation, its description and diagram are omitted.

D. How to Construct a Protocol for Any Function

We now have committed-format AND, COPY, and NOT protocols as shown in Sections B-B and B-C. Based on these protocols, we can construct a protocol for any function as follows. If we want to compute an arbitrary function f given commitments to each of the inputs, we first create a Boolean circuit representing f, and then construct a protocol based on the circuit by combining AND/COPY/NOT protocols.

Protocol 5. The efficient AND protocol in committed format: $\Big([\blacksquare, \square, \circ, \circ, \circ, \circ, \circ, \circ], 4, 2, \{0, 1\}^2, Q, A\Big)$.

Initial state:

$([\blacksquare, \square, \circ, \circ, \circ, \circ, \circ, \circ]; \{\!\{\circ\}\!\}, \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\}, \{\!\{\bullet\}\!\}; [], [], [], [])$ $(p_{00}, 0, 0, 0)$
$([\blacksquare, \square, \circ, \circ, \circ, \circ, \circ, \circ]; \{\!\{\circ\}\!\}, \{\!\{\bullet\}\!\}, \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\}; [], [], [], [])$ $(0, p_{01}, 0, 0)$
$([\blacksquare, \square, \circ, \circ, \circ, \circ, \circ, \circ]; \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\}, \{\!\{\circ\}\!\}, \{\!\{\bullet\}\!\}; [], [], [], [])$ $(0, 0, p_{10}, 0)$
$([\blacksquare, \square, \circ, \circ, \circ, \circ, \circ, \circ]; \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\}, \{\!\{\bullet\}\!\}, \{\!\{\circ\}\!\}; [], [], [], [])$ $(0, 0, 0, p_{11})$

Steps:

1) $(\mathsf{MergeBags}, 3, 1)$
2) $(\mathsf{PublicPut}, \circ, 2)$
3) $(\mathsf{PublicPut}, \circ, 4)$
4) **while**(1)
5) $\quad(\mathsf{Shuf}, \{1, 2, 4\})$
6) $\quad(\mathsf{Take}, 1, \text{"}small''\text{)}$
7) $\quad$**if** taken ball = $\bullet$ **then**
8) $\quad\quad$**break**
9) $\quad$**else**
10) $\quad\quad(\mathsf{Back}, \circ, 1)$
11) $\quad\quad(\mathsf{PublicPut}, \circ, 1)$
12) $(\mathsf{MergeBags}, 4, 2)$
13) $(\mathsf{PublicPut}, \circ\circ\circ\blacksquare, 1)$
14) $(\mathsf{PublicPut}, \square, 2)$
15) $(\mathsf{Shuf}, \{1, 2\})$
16) $(\mathsf{TakeAll}, \text{"small"})$
17) **if** visible conf. = $([\bullet, \circ, \circ, \circ], [\circ, \circ, \circ, \circ])$ **then**
18) $\quad(\mathsf{BackAll})$
19) $\quad(\mathsf{PublicPut}, \bullet, 1)$
20) $\quad(\mathsf{PublicPut}, \circ, 2)$
21) **else**
22) $\quad(\mathsf{BackAll})$
23) $\quad(\mathsf{PublicPut}, \bullet, 2)$
24) $\quad(\mathsf{PublicPut}, \circ, 1)$
25) $(\mathsf{Shuf}, \{1, 2\})$
26) $(\mathsf{TakeAll}, \text{"big"})$
27) **if** visible conf. = $([\blacksquare], [\square])$ **then**
28) $\quad(\mathsf{Result}, 1, 2)$
29) **else**
30) $\quad(\mathsf{Result}, 2, 1)$

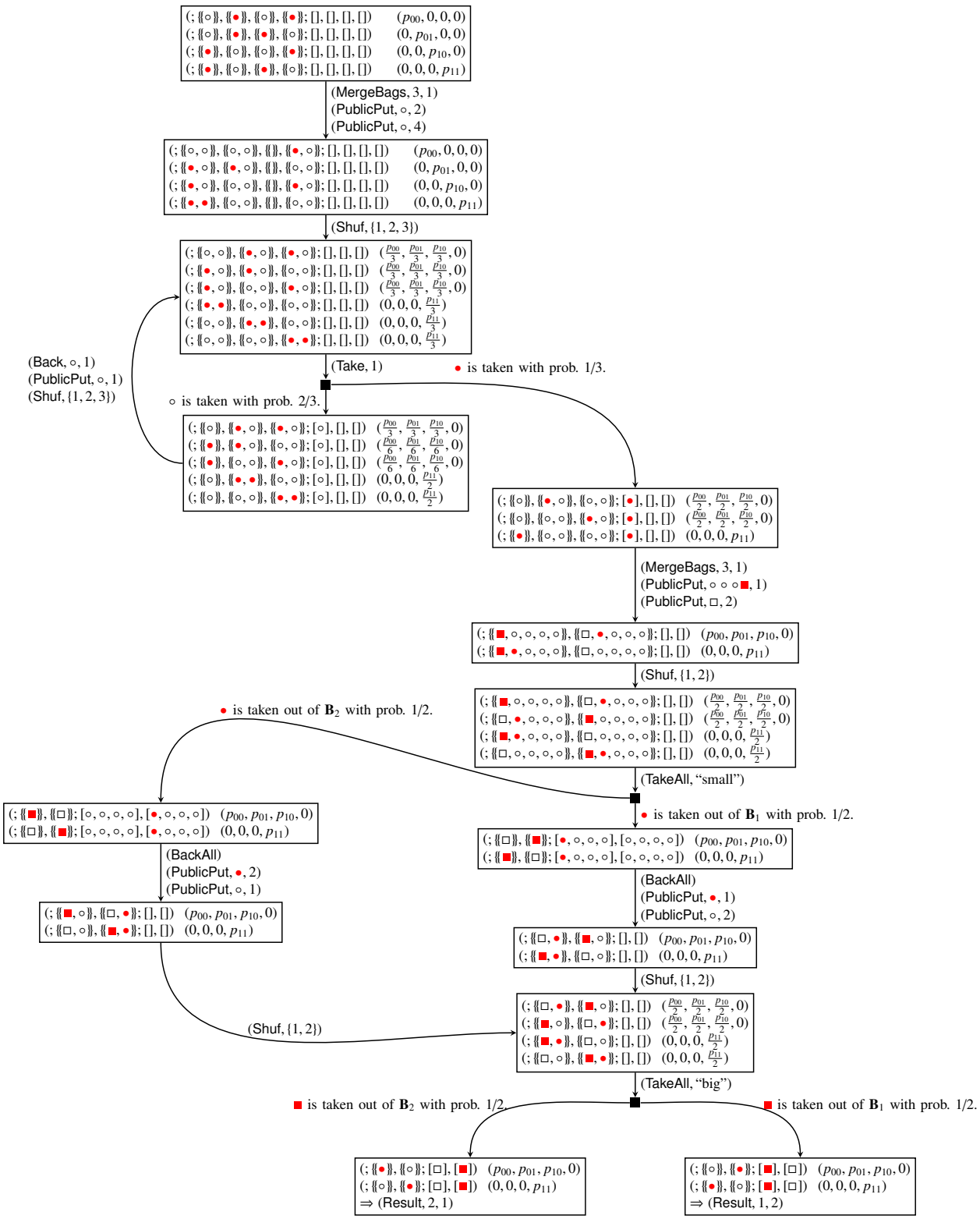


Fig. 6: A diagram of the committed-format AND protocol, given two commitments to $x_1 \in \{0, 1\}$ and $x_2 \in \{0, 1\}$, where we omit the tray T_0 and renumber the positions of the bags after deleting unused empty bags.