Fairness in Secure Two-Party Computation with Rational Players

Arpita Maitra[1], Goutam Paul[2] and Asim K. Pal[1]
[1] Management Information Systems Group, Indian Institute of Management Calcutta, India.
Email: {arpitam, asim}@iimcal.ac.in
[2] Cryptology & Security Research Unit, R. C. Bose Centre for Cryptology & Security,
Indian Statistical Institute, Kolkata, Email: goutam.paul@isical.ac.in

Abstract

A seminal result of Cleve (STOC 1986) showed that fairness, in general, is impossible to achieve in case of two-party computation if one of them is malicious. Later, Gordon et al. (STOC 2008) observed that there exist some functions for which fairness can be achieved even though one of the two parties is malicious. One of the functions considered by Gordon et al. is exactly the millionaires' problem (Yao, FOCS 1982) or, equivalently, the 'greater-than' function. Interestingly, Gordon et al. (JACM, 2011) showed that any function over polynomial-size domains which does not contain an "embedded XOR" can be converted into the greater than function. In the same paper, they also demonstrated feasibility for certain functions that do contain an embedded XOR. In this paper, we revisit both the classes of two-party computation under rational players for the first time. We show that Gordon's protocols no longer remain fair when the players are rational. Further, we design two protocols, one without embedded XOR (the greater-than function) and the other with embedded XOR, and show that with rational players, our protocols achieve fairness under suitable choice of parameters.

Keywords: Cryptography, embedded XOR, Fairness, millionaires' problem, secure computation.

1 Introduction

In a secure two-party computation, two parties or players want to compute a particular function of their inputs along with preserving specific security notions, such as, fairness, correctness etc. Informally, *correctness* means that no party computes a wrong function and complete *fairness* means that either every one or no one computes the function.

In [5], Cleve showed an impossible result that certain functions cannot be computed with complete fairness without an honest majority. From this, the community conjectured that no function can be computed without an honest majority. However, in [3, 4] the authors showed that absolute correctness can be achieved in case of multi-party computation with one-third faulty players. They proposed the solution in broadcasting channel model. After more than two decades, Gordon et al. [7] came with a set of functions for which complete fairness is possible for two-party computation in non-simultaneous channel model, even if one of the players is malicious.

One particular function of interest in Gordon's paper was the Yao's millionaires' problem [13], or more precisely, the 'greater than' function. The problem deals with two millionaires, Alice and Bob, who are interested in finding who amongst them is richer, without revealing their actual wealth to each other. Since the subsequent work [8] by Gordon et al. showed that any function over polynomial-size domains which does not contain an "embedded XOR" can be converted into the greater than function, the millionaires' problem covers all functions without embedded XOR.

In this paper, for the first time we study the fairness in millionaires' problem with rational players. Rational players are neither 'good' nor 'malicious', they are utility maximizing. Each rational party wishes to learn the output while allowing as few others as possible to learn the output. Thus, each rational party chooses to abort as soon as it obtains the output. We show that with rational players, Gordon's solution of the Yao's millionaires' problem no longer remains fair. We also propose a modification in the protocol with the help of a third player so that fairness can be established.

The work by Gordon et al. [7, 8] also studied the equality function that belong to the class of embedded XOR. The equality function simply checks whether the inputs chosen by two players (from a specified domain) are equal or not. They showed that under certain parameter value of a hybrid model, fairness is achieved. In this paper, we also revisit this problem with rational players for the first time and show that fairness is no longer guaranteed. We propose a modified version of the protocol and prove its fairness under rational setting.

Note that we need to introduce an intermediate third party to achieve fairness for the millionaires' problem in rational domain. However, no such requirement is there for the embedded XOR problem. One may think that the use of a third player is no different than the use of a dealer. But the fact is that our third party is less restrictive in comparison with the dealer. The dealer is assumed to be honest (this is a strong assumption), whereas the third party is assumed to be rational in nature. Moreover, the dealer is a special distinct entity from the players. However, the role of our intermediate third party can be adopted by any rational player who is not a party involved in the problem being solved. Only assumption on this player is that it is fail-stop in nature.

1.1 Contributions

We list our key contributions one by one.

1. We revisit fairness in two prominent Secure Two-Party Computation problems, namely, Yao's millionaires' problem [13] and the equality function of the Embedded XOR problems, for the first time with rational players.

2. We show that Gordon's protocol [7, 8] for solving the millionaires' problem no longer remain fair when the players are rational (Theorem 1).

3. We propose a variant of Gordon et al.'s protocol and show that fairness can be regained (Theorem 4). We also establish correctness of the new protocol (Theorem 3).

4. In order to establish fairness of our protocol, we introduce a third player, who is also rational and not a trusted third-party such as dealer. This helps to keep the dealer offline.

5. We show that the equality problem in the embedded XOR category [7, 8] also no longer remains fair with rational players (Theorem 5).

6. We propose a variant of Gordon et al.'s protocol and show that fairness can be guaranteed under certain practical assumptions (Theorem 6).

7. For both the problems, we also discuss the issues with unequal vs. equal domain sizes.

2 Preliminaries

In this section, we briefly describe the concepts of rationality, fairness, fail-stop and Byzantine setting used in this work.

We define a *function reconstruction protocol with rational adversary* to be a pair $(\Gamma, \overrightarrow{\sigma})$, where Γ is the game (i.e., specification of allowable actions) and $\overrightarrow{\sigma}=(\sigma_1, \ldots, \sigma_n)$ denotes the strategies followed by n number of players. We use the notations $\overrightarrow{\sigma}_{-w}$ and $(\sigma'_w, \overrightarrow{\sigma}_{-w})$ respectively for $(\sigma_1, \ldots, \sigma_{w-1}, \sigma_{w+1}, \ldots, \sigma_n)$ and $(\sigma_1, \ldots, \sigma_{w-1}, \sigma'_w, \sigma_{w+1}, \ldots, \sigma_n)$. The outcome of the game is denoted by $\overrightarrow{o}(\Gamma, \overrightarrow{\sigma})=(o_1, \ldots, o_n)$. The set of possible outcomes with respect to a party P_w is as follows. 1) P_w correctly computes f, while others do not; 2) everybody correctly computes f; 3) nobody computes f; 4) others computes f correctly, while P_w does not and 5) others believe in a wrong functional value, while P_w does not.

The output that no function is computed is denoted by $\perp$ (i.e., *null* as in [7]) and output of wrong computation is denoted by $\neg$.

In classical domain, the adversary that controls a player may be computationally bounded. Here, we assume the adversary has probabilistic polynomial time complexity.

2.1 Utilities and Preferences

The utility function u_w of each party P_w is defined over the set of possible outcomes of the game. The outcomes and corresponding utilities for two parties are described in Table 1. We here assume Bernoulli utility function.

Table 1: Outcomes and Utilities for $(2,2)$ rational function reconstruction

P_1's outcome (o_1)	P_2's outcome (o_2)	P_1's Utility $U_1(o_1, o_2)$	P_2's Utility $U_2(o_1, o_2)$
$o_1=f$	$o_2=f$	U_1^{TT}	U_2^{TT}
$o_1=\perp$	$o_2=\perp$	U_1^{NN}	U_2^{NN}
$o_1=f$	$o_2=\perp$	U_1^{TN}	U_2^{NT}
$o_1=\perp$	$o_2=f$	U_1^{NT}	U_2^{TN}
$o_1=\perp$	$o_2=\neg$	U_1^{NF}	U_2^{FN}
$o_1=\neg$	$o_2=\perp$	U_1^{FN}	U_2^{NF}

Players have their preferences based on the different possible outcomes. In this work, a rational player w is assumed to have the following preference:

$$\mathcal{R}_1 : U_w^{TN} > U_w^{TT} > U_w^{NN} > U_w^{NT}.$$

Some players may have the additional preference $U_w^{NF} \geq U_w^{TT}$, whereas the rest have $U_w^{NF} < U_w^{TT}$.

2.2 Fairness

In non-rational setting, the security of a protocol is analyzed [11, 7, 8] by comparing what an adversary can do in a *real* protocol execution to what it can do in an *ideal* scenario that is secure by definition. This is formalized by considering an ideal computation involving an incorruptible trusted party to whom the parties send their inputs. The trusted party computes the functionality on the inputs and returns to each party its respective output. Loosely speaking, a protocol is secure if any adversary interacting in the real protocol (where no trusted party exists) can do no more harm than if it were involved in the above-described ideal computation.

A rational player, being selfish, desires an unfair outcome, i.e., computing the function alone. Therefore, the basic aim of rational computation has been to achieve fairness. According to Von

Neumann and Morgenstern *expected utility theorem* [12], under natural assumptions, the individual would prefer one prospect $\mathcal{O}_1$ over another prospect $\mathcal{O}_2$ if and only if $E[U(\mathcal{O}_1) \geq E[U(\mathcal{O}_2)]$. The work [1] implicitly uses the expected utility theorem to derive its results. We also use the same approach and accordingly redefine fairness as follows.

Definition 1. *(Fairness) A rational function reconstruction mechanism $(\Gamma, \overrightarrow{\sigma})$ is said to be completely fair if a party P_w, $(w \in \{1, \ldots, n\})$, who is corrupted by a probabilistic polynomial time adversary, the following holds:*

$$U_w^{TT} \geq E[U_w(\mathcal{O}_l)],$$

where $\mathcal{O}_l = \{o_w^1, \ldots, o_w^{n'}; p_1, \ldots, p_{n'}\}$ is any arbitrary prospect and n' is the number of possible outcomes.

2.3 Fail-stop and Byzantine settings

In the fail-stop setting, each party follows the protocol as directed except that it may choose to abort at any time [9] and a party is assumed not to change its input when running the protocol. On the other hand, in Byzantine setting, a deviating party may behave arbitrarily. It may change the inputs or may choose to abort. For our analysis, we consider both the settings.

3 Millionaires' Problem with Rational Players

In this section, we first describe the millionaires' problem or, more precisely, the greater than function, proposed by Gordon et al. [7, 8]. We, then, will show how fairness condition is affected in the presence of the rational players having the preferences $\mathcal{R}_1$. Let us denote two players by P_1 and P_2. Suppose P_1 has the secret i and P_2 has the secret j, $1 \leq i \leq M$, $1 \leq j \leq M$. The dealer gives an ordered list $X = \{x_1, x_2, \ldots, x_M\}$ to P_1 and another ordered list $Y = \{y_1, y_2, \ldots, y_M\}$ to P_2. Then P_1 sends x_i to the dealer and P_2 sends y_j to the dealer. Let f be a deterministic function which maps $X \times Y \to \{0, 1\} \times \{0, 1\}$. The function $f(x_i, y_j)$ can be defined as a pair of outputs, i.e., $f(x_i, y_j) = (f_1(x_i, y_j), f_2(x_i, y_j))$, where $f_1(x_i, y_j)$ is the output of the first party and $f_2(x_i, y_j)$ is the output of the second party. For millionaires' problem, the function is defined as follows [7, 8]. For $w = 1, 2$,

$$f_w(x_i, y_j) = \begin{cases} 1 & \text{if } i > j; \\ 0 & \text{if } i \leq j. \end{cases} \tag{1}$$

The protocol proceeds in a series of M iterations. The dealer creates two sequences $\{a_l\}$ and $\{b_l\}$, $l = 1, 2, \ldots, M$, as follows.

$$a_i = b_j = f_1(x_i, y_j) = f_2(x_i, y_j).$$

For $l \neq i$, $a_l = \perp$ and for $l \neq j$, $b_l = \perp$.

Next, the dealer splits the secret a_l into the shares a_l^1 and a_l^2, and the secret b_l into the shares b_l^1 and b_l^2, so that $a_l = a_l^1 \oplus a_l^2$ and $b_l = b_l^1 \oplus b_l^2$, and gives the shares $\{(a_l^1, b_l^1)\}$ to P_1 and the shares $\{(a_l^2, b_l^2)\}$ to P_2. In each round l, P_2 sends a_l^2 to P_1, who, in turn sends b_l^1 to P_2. P_1 learns the output value $f_1(x_i, y_j)$ in iteration i, and P_2 learns the output in iteration j. As we require three elements, 0, 1 and $\perp$, we define 0 by 00, 1 by 11 and $\perp$ by 01. The algorithm for the functionality share generation in fail-stop setting is revisited in Algorithm 1. Here we assume that the dealer who will distribute the shares is honest and can compute the function described in Equation (1). The protocol for computing f is described in Algorithm 2.

The algorithms in the Byzantine setting are the same as those in the fail-stop setting except some additional steps. In Byzantine setting, the shares are signed by the dealer. Along with the

Inputs:

x_i from P_1 and y_j from P_2. If one of the received input is not in the correct domain, then both the parties are given $\perp$.

Computation:

The dealer does the following:

Prepares a list $list_w$ of shares for each party P_w, where $w \in \{1, 2\}$ such that

P_1 receives the values of $a_1^1, a_2^1, \ldots, a_M^1$ and $b_1^1, b_2^1, \ldots, b_M^1$.

P_2 receives the values of $a_1^2, a_2^2, \ldots, a_M^2$ and $b_1^2, b_2^2, \ldots, b_M^2$.

Output:

$a_l = a_l^1 \oplus a_l^2$.

$b_l = b_l^1 \oplus b_l^2$.

For $l \in \{1, \ldots, M\}, l \neq i$, set $a_l = \perp$.

For $l \in \{1, \ldots, M\}, l \neq j$, set $b_l = \perp$.

$a_i = b_j = f_1(x_i, y_j) = f_2(x_i, y_j)$.

$a_1, a_2, \ldots, a_M$ and $b_1, b_2, \ldots, b_M$ correspond to the outputs of P_1 and P_2 respectively for $1 \leq l \leq M$.

Algorithm 1: ShareGen

Inputs:

P_1 obtains $a_1^1, a_2^1, \ldots, a_M^1$ and $b_1^1, b_2^1, \ldots, b_M^1$.

P_2 obtains $a_1^2, a_2^2, \ldots, a_M^2$ and $b_1^2, b_2^2, \ldots, b_M^2$.

Computation:

There are M number of iterations. In each iteration $l \in \{1, 2, \ldots, M\}$ do:

P_2 sends a_l^2 to P_1 and P_1 computes $a_l = a_l^1 \oplus a_l^2$.

P_1 sends b_l^1 to P_2 and P_2 computes $b_l = b_l^1 \oplus b_l^2$.

Output:

If P_2 aborts in round l, i.e., does not send its share at that round and $l \leq i$, P_1 outputs 1. If $l > i$, P_1 has already determined the output in some earlier iteration. Thus it outputs that value.

If P_1 aborts in round l, i.e., does not send its share at that round and $l \leq j$, P_2 outputs 0. If $l > j$, P_2 has already determined the output in some earlier iteration. Thus it outputs that value.

Algorithm 2: $\mathbf{\Pi^{CMP}}$

shares of the function, the dealer also distributes some secret keys $k_a, k_b \leftarrow Gen(1^\lambda)$, where λ is the security parameter. For $1 \leq l \leq M$, let $t_l^a = Mac_{k_a}(l \parallel a_l^2)$ and $t_l^b = Mac_{k_b}(l \parallel b_l^1)$. P_1 receives $a_1^1, a_2^1, \ldots, a_M^1$ and $(b_1^1, t_1^b), (b_2^1, t_2^b), \ldots, (b_M^1, t_M^b)$ and MAC key k_a. Similarly P_2 is given $(a_1^2, t_1^a), (a_2^2, t_2^a), \ldots, (a_M^2, t_M^a)$ and $b_1^2, b_2^2, \ldots, b_M^2$ and MAC key k_b. After receiving the share in the round l from P_2, P_1 verifies by the algorithm $Vrfy_{k_a}(l \parallel a_l^2, t_l^a)$. If $Vrfy_{k_a}(l \parallel a_l^2, t_l^a) = 0$, P_1 halts. Similarly, after receiving the share in the round l from P_1, P_2 verifies by the algorithm $Vrfy_{k_b}(l \parallel b_l^1, t_l^b)$. If $Vrfy_{k_b}(l \parallel b_l^1, t_l^b) = 0$, P_2 halts. Otherwise both continues the protocol $\mathbf{\Pi^{CMP}}$ which outputs $a_i(b_j)$ for $P_1(P_2)$.

Exploiting the MAC signature, we can resist the players to send a false share.

3.1 $\mathbf{\Pi^{CMP}}$ is not fair when players are rational

In this section, we revisit the fairness issue in the millionaires' problem [7] considering the rational players. We also assume that the players, P_1 and P_2 have the preferences $\mathcal{R}_1$. Either of the players also has $U_w^{NF} \geq U_w^{TT}$. We observed that Gordon's protocol [7, 8] is no longer fair in this case.

Theorem 1. *Provided $\mathcal{R}_1$ and $U_w^{NF} \geq U_w^{TT}$ for some player P_w, the protocol $\mathbf{\Pi^{CMP}}$ is not completely fair.*

Proof. Suppose P_1 aborts before giving its share in round l, where $1 \leq l \leq M$. Now, if $i \leq j$, we list all possible mutually exclusive and exhaustive outcomes as follows:

1. When $1 \leq l < i$, P_2 outputs 0 and correctly concludes that $i \leq j$, but P_1 outputs $\perp$.
2. When $i \leq l \leq M$, P_1 obtains the function and both correctly conclude that $i \leq j$.

In this case, the utility of P_1 is given by

$$U_1^{\leq} = \begin{cases} U_1^{NT} & \text{if } 1 \leq l < i; \\ U_1^{TT} & \text{if } i \leq l \leq M; \end{cases} \tag{2}$$

If $i > j$, all possible mutually exclusive and exhaustive outcomes are:

1. When $1 \leq l \leq j$, P_2 outputs 0 and wrongly concludes that $i \leq j$, but P_1 outputs $\perp$.
2. When $j < l < i$, P_1 outputs $\perp$, but P_2 correctly concludes that $i > j$.
3. When $i \leq l \leq M$, both computes the function and both correctly conclude that $i > j$.

Thus, the corresponding utility for this event is given by

$$U_1^{>} = \begin{cases} U_1^{NF} & \text{if } 1 \leq l \leq j; \\ U_1^{NT} & \text{if } j < l < i; \\ U_1^{TT} & \text{if } i \leq l \leq M; \end{cases} \tag{3}$$

Since i is known to P_1, the expected utility of P_1 is given by

$$E[U_1] = \Pr(i \leq j) \cdot E[U_1^{\leq}] + \Pr(i > j) \cdot E[U_1^{>}], \tag{4}$$

where $\Pr(i \leq j) = \frac{M-i+1}{M}$ and $\Pr(i > j) = \frac{i-1}{M}$. Plugging in the values from Equation (2) and (3) into Equation (4), we get

$$
\begin{aligned}
E[U_1] &= \begin{cases} \left(\frac{M-i+1}{M}\right) U_1^{NT} + \left(\frac{i-1}{M}\right)\left(\left(\frac{l-1}{i-1}\right) U_1^{NT} + \left(\frac{i-l}{i-1}\right) U_1^{NF}\right) & \text{if } 1 \leq l < i; \\ \left(\frac{M-i+1}{M}\right) U_1^{TT} + \left(\frac{i-1}{M}\right) U_1^{TT} & \text{if } i \leq l \leq M. \end{cases} \\
&= \begin{cases} \left(\frac{M-i+l}{M}\right) U_1^{NT} + \left(\frac{i-l}{M}\right) U_1^{NF} & \text{if } 1 \leq l < i; \\ U_1^{TT} & \text{if } i \leq l \leq M. \end{cases}
\end{aligned}
$$

Note that in the first case, i.e., for $1 \leq l < i$, the second term corresponding to $i > j$ involves two sub cases, namely, $1 \leq j < l < i$ and $l \leq j < i$.

Observe that when $i \leq l \leq M$, P_1 has already obtained the secret, but by aborting it cannot increase its utility beyond U_1^{TT}.

However, when $l < i$, we may have $E[U_1] > U_1^{TT}$, depending on the value of U_1^{NF}.Thus, dependence on U_1^{NF} prevents the protocol to achieve fairness in this case. On other words, we can say that when a party aborts before it obtains the output, the only reason would be if he is significantly more interested in cheating the other party rather than him not getting it.

The analysis for P_2 is similar, except we have the role of i and j interchanged. □

3.2 How to make $\mathbf{\Pi^{CMP}}$ fair when players are rational

In this section, we propose a variant of the Gordon's [7, 8] protocol. In the earlier section, we have observed that $\mathbf{\Pi^{CMP}}$ suffers from early abort. In [9] it is shown that two party fair computation is possible. However, their scheme exploits the concepts of online dealer, which is not very practical, as in each iteration the dealer has to interact with the players and has to ask them whether they will choose abort. Another restriction in their scheme is that the deviating player can not escape from its decision knowing that the round it has chosen to abort is less than or equal to the revelation round. Exploiting the idea of the indicator bit (a bit in [10], a signal in [6]), one can make the dealer offline. We propose a new protocol with a rational intermediate player for offline dealer and show that the protocol is U^{NF}-independent and hence correct [2]. We also prove fairness for our protocol. Our protocol is described in Algorithm 3 and Algorithm 4.

Though our protocol initially addresses towards the millionaires' problem, it is applicable for any function which does not have any embedded XOR [8].

Here, the intermediate player, P_3, is considered as a rational player who is guided by his expected utility or revenue at the end of the game. He will participate in the game in the motivation towards maximizing his utility. In non-rational setting, P_3 is termed as an 'untrusted third party'. Only assumption on this player is that it is fail-stop in nature.

P_3 has been given two options at the beginning of the game.

- **Option** 1: Follow the protocol (i.e., send the shares to both the parties) and obtain a positive reputation value, say δ, from the dealer.
- **Option** 2: Before delivering the shares in any round, approach to one of the players to give δ amount of money, in exchange of sending the share to him only.

We assume that $\delta \leq U_w^{TN} - U_w^{TT}$ for $w \in \{1, 2\}$.

As P_3 is rational and hence utility maximizer, he first checks whether choosing **Option 2** would be meaningful to him. Without loss of generality, we assume that P_3 chooses P_1 to approach. In this case, if P_1 agrees to give the money to P_3, P_3 will send the share to him but not to P_2. The

Inputs:

x_i from P_1 and y_j from P_2. If one of the received input is not in the correct domain, then both the parties are given $\perp$.

Computation:

The dealer does the following:

Insert an intermediate player P_3.

Chooses r according to a geometric distribution $\mathcal{G}(\gamma)$ with parameter γ and sets r as the revelation round, i.e., the round in which the value of f is either $(0,0)$ or $(1,1)$.

Chooses d according to the geometrical distribution $\mathcal{G}(\gamma)$ and sets the total number of iterations as $m = r + d$.

Prepares a list $list_w$ of shares for each party P_w, where $w \in \{1,2,3\}$ such that

P_1 receives the values of $a_1^1, a_2^1, \ldots, a_m^1$, $b_1^1, b_2^1, \ldots, b_m^1$ and $c_1^1, c_2^1, \ldots c_m^1$.

P_2 receives the values of $a_1^2, a_2^2, \ldots, a_m^2$, $b_1^2, b_2^2, \ldots, b_m^2$ and $c_1^2, c_2^2, \ldots c_m^2$.

P_3 receives the values of $c_1^3, c_2^3, \ldots, c_m^3$.

Output:

$a_l = a_l^1 \oplus a_l^2 \oplus a_l^3$, where $a_l^3 = c_l^2 \oplus c_l^3$.

$b_l = b_l^1 \oplus b_l^2 \oplus b_l^3$, where $b_l^3 = c_l^1 \oplus c_l^3$.

$a_r = b_r = f(x_i, y_j)$, where x_i and y_j are parties inputs.

For $l \in \{1, \ldots, m\}, l \neq r$, set $a_l = \perp$.

For $l \in \{1, \ldots, m\}, l \neq r$, set $b_l = \perp$.

$a_1, a_2, \ldots, a_m$ and $b_1, b_2, \ldots, b_m$ correspond to the outputs of P_1 and P_2 respectively for $1 \leq l \leq m$.

Algorithm 3: ShareGen for $\mathbf{\Pi_{fair}^{CMP}}$

Inputs:

P_1 obtains $a_1^1, a_2^1, \ldots, a_m^1$, $b_1^1, b_2^1, \ldots, b_m^1$ and $c_1^1, c_2^1, \ldots, c_m^1$.

P_2 obtains $a_1^2, a_2^2, \ldots, a_m^2$, $b_1^2, b_2^2, \ldots, b_m^2$ and $c_1^2, c_2^2, \ldots, c_m^2$.

P_3 obtains $c_1^3, c_2^3, \ldots, c_m^3$.

Computation:

There are m number of iterations. In each iteration $l \in \{1, 2, \ldots, m\}$ do the following.

P_2 sends a_l^2 to P_1 and P_1 sends b_l^1 to P_2.

After receiving the share fromP_2, P_1 sends c_l^1 to P_3, else halts.

After receiving the share fromP_1, P_2 sends c_l^2 to P_3, else halts.

P_3 computes the values of a_l^3 and b_l^3 and sends a_l^3 to P_1 and then b_l^3 to P_2.

Output:

If P_2 aborts in round l, i.e., does not send its share at that round and $l \leq r$, P_1 outputs $\perp$. If $l > r$, P_1 has already determined the output in some earlier iteration. Thus it outputs that value.

If P_1 aborts in round l, i.e., does not send its share at that round and $l \leq r$, P_2 outputs $\perp$. If $l > r$, P_2 has already determined the output in some earlier iteration. Thus it outputs that value.

If P_1 or P_2 does not send its share to P_3, P_3 outputs $\perp$ to the both of the players.

If P_3 does not send its computed share to any one of the party P_w, $w \in \{1,2\}$, in a round l, P_w chooses to abort from the very next round and the protocol will be terminated.

Algorithm 4: $\mathbf{\Pi_{fair}^{CMP}}$

following result shows that P_1 will not have any incentive to give the money to P_3 in the motivation to get the output by himself only provided certain conditions hold.

Theorem 2. *Provided $\delta > 0$, $0 < \gamma < 1$ and $U_w^{TN} + (1-\gamma)U_w^{NN} < U_w^{TT}$ for all $w \in \{1, 2\}$, P_3 always chooses* **Option** 1 *and plays the game honestly.*

Proof. According to the protocol, to obtain the secret alone with the help of P_3, P_1 has to guess correctly the revelation round. Otherwise, the protocol will be terminated from the very next round and both the players get no information about the output. Suppose, P_1 guesses the l-th round to be the revelation round and gives P_3 the money for that round so that P_3 will not send the corresponding share to P_2 for that round. If the guess is correct, i.e., $l = r$, the probability of which is γ, its utility is $(U_1^{TN} - \delta)$. Otherwise, its utility is $(U_1^{NN} - \delta)$, as in this case P_2 will abort from the next round. So the expected utility of P_1 is given by

$$\gamma(U_1^{TN} - \delta) + (1-\gamma)(U_1^{NN} - \delta) = \gamma U_1^{TN} + (1-\gamma)U_1^{NN} - \delta < U_1^{TT} - \delta < U_1^{TT}.$$

The last inequality follows from our assumptions that δ is positive and $\gamma U_1^{TN} + (1-\gamma)U_1^{NN} < U_1^{TT}$. Thus P_1 has no incentive to offer money to the intermediate player P_3 in the motivation to get the function alone. Similar analysis can be done for P_2.

As the utility values are public and P_3 knows the condition that $U_w^{TN} + (1-\gamma)U_w^{NN} < U_w^{TT}$ for all $w \in \{1, 2\}$, he always chooses **Option** 1. □

In our mechanism, there are three players, namely P_1, P_2 and P_3. For the condition of achieving correctness and fairness, we have to assume that when one of the players deviates, others are sticking to the protocol. From the above analysis we have seen that P_3 has no incentive to deviate from the protocol. Thus, we have to consider the following two cases.

1. P_1 deviates (P_2 follows the protocol).

2. P_2 deviates (P_1 follows the protocol).

In fail-stop setting, the deviation of P_1 and P_2 is considered as early abort whereas in Byzantine setting the players behave arbitrarily. That means they can abort early as well as can send the arbitrary inputs or can swap the inputs.

We analyze the security notions such as correctness and fairness considering all the above issues. The following theorems show that our proposed mechanism is correct and fair.

In Byzantine setting, the shares given to the players are signed by the dealer so that no player can send a false share to the other player. The signing procedure discussed in Section 3 remains similar in our protocol expect M is replaced by m and with some additional steps.

- For $1 \le l \le m$, P_1 is given $(c_l^1, t_l^{c_1})$, where $t_l^{c_1} = Mac_{k_{c_1}}(l \parallel c_l^1)$.

- For $1 \le l \le m$, P_2 is given $(c_l^2, t_l^{c_2})$, where $t_l^{c_2} = Mac_{k_{c_2}}(l \parallel c_l^2)$.

- P_3 is given MAC key k_{c_1} and MAC key k_{c_2} so that for $1 \le l \le m$, it can verify the shares by algorithm $Vrfy_{k_{c_1}}(l \parallel c_l^1, t_l^{c_1})$ for P_1 and $Vrfy_{k_{c_2}}(l \parallel c_l^2, t_l^{c_2})$ for P_2. If $Vrfy_{k_{c_w}}(l \parallel c_l^w, t_l^{c_w}) = 0$, P_3 halts, else continues, where $w \in \{1, 2\}$.

There is no need to sign the shares given to P_3, as P_3 is fail-stop by nature.

The following result establishes the correctness of the protocol.

Theorem 3. *The protocol $\mathbf{\Pi^{CMP}_{Fair}}$ is U_w^{NF}-independent for $w \in \{1, 2\}$ and hence correct.*

Proof. we should recall that the deviations of P_1 and P_2 are similar. Thus for simplicity, here, we only consider the deviations of P_1.

In fail-stop setting, if P_1 aborts early and the round in which he aborts is less than j, according to Gordon's protocol, P_2 will output 0 and conclude that $i \leq j$. When $i > j$, it is the situation when P_2 is deceived by P_1. However, our protocol is designed in such a way that if P_1 has chosen abort in any round before r, P_2 will output $\perp$ and does not conclude anything. Thus, P_1 can not deceive P_2 by early abort. There is no incentive for P_1 to abort in a round $l > r$, as P_2 has already determined the output in some earlier iteration.

In case of Byzantine setting, P_1 can send arbitrary shares to both P_2 and P_3, so that P_2 will finally compute a wrong function. But since each share is signed by the dealer, no one can send an arbitrary share to the other. Another important deviation of P_1 in this setting is to swap the inputs. By swapping the inputs, P_1 can make P_2 compute a wrong function. As all the inputs came from the same dealer, there is no chance to catch this type of deviation by considering only the signature scheme. However, we consider signature with tagging. P_1 receives $a_1^1, a_2^1, \ldots, a_m^1$ and $(b_1^1, t_1^b), (b_2^1, t_2^b), \ldots, (b_m^1, t_m^b)$ and MAC key k_a. Similarly P_2 is given $(a_1^2, t_1^a), (a_2^2, t_2^a), \ldots, (a_m^2, t_m^a)$ and $b_1^2, b_2^2, \ldots, b_m^2$ and MAC key k_b. After receiving the share in the round l from P_1, if $Vrfy_{k_b}(l \parallel b_l^1, t_l^b) = 0$, then P_2 halts. Similar checking is done by P_3 as well. Thus, by input swapping no one can make the other believe in a wrong function.

Thus, assuming P_1 has $U_1^{NF} > U_1^{TT}$, the mechanism is designed in such a way that it becomes U_1^{NF} independent and hence correct. Proceeding in the same way for P_2, we can prove the U_2^{NF} independence. □

Now we are in a position to establish fairness of $\mathbf{\Pi^{CMP}_{Fair}}$.

Theorem 4. *Provided $\mathcal{R}_1$, the protocol $\mathbf{\Pi^{CMP}_{Fair}}$ achieves fairness.*

Proof. Without loss of generality, let us assume that the player P_1 is deviating. The analysis when P_2 deviates is similar.

In this case, the reason for deviation is to get the function alone. In fail-stop as well as in Byzantine setting P_1 can abort in round l.

P_1 may choose three types of abort in round l.

1. It may not send its share to P_2.
2. It may not send its share to P_3.
3. It may not send its share to both P_2 and P_3.

If P_1 does not send its share to P_2, then P_2 will not send its share to P_3. As a result the protocol will be terminated without producing any result either for P_1 or P_2. Similarly, if P_1 does not send its share to P_3, according to the protocol P_3 will output $\perp$ to both the players. In the third case, the protocol will be terminated from the beginning of the round l. Thus, there is no incentive for P_1 to abort early in the motivation to get the secret alone. □

3.3 Fairness analysis of $\mathbf{\Pi^{CMP}}$ when players have unequal domain size

As discussed in [8, Section 3.2], when the domain sizes of the players are unequal, the analysis in the non-rational setting does not change. It is easy to see from our analysis of Section 3.1 that even in the rational setting, we can carry out an analogous calculation to conclude that the protocol is U^{NF}-dependent and hence not fair.

4 Secure Two-Party Computation involving Embedded XOR with Rational Players

In this section, we first describe the embedded XOR problem or, more precisely, the equality function, proposed by Gordon et al. [7]. We, then, will show how fairness condition is affected in the presence of the rational players having the preferences $\mathcal{R}_1$. Let us denote two players by P_1 and P_2. Player P_1 is given an ordered list $\{x_1, x_2, x_3\}$ and P_2 is given an ordered list $\{y_1, y_2\}$. P_1 randomly chooses the input from the ordered list and sends to the dealer. P_2 also randomly chooses the input from his list and delivers to the dealer. Dealer calculates the function. For convenience, we here recall the table for f given in [7].

	y_1	y_2
x_1	0	1
x_2	1	0
x_3	1	1

The function can be described as

$$f(x_i, y_j) = \begin{cases} 1 & \text{if } i \neq j; \\ 0 & \text{if } i = j. \end{cases} \tag{5}$$

The protocol proceeds in a series of M iterations, where $M = \omega(\log \lambda)$, λ is the security parameter. Let x and y denote the inputs from P_1 and P_2 respectively. The dealer chooses the revelation round l^* according to geometric distribution with parameter γ. The dealer then creates two sequences $\{a_l\}$ and $\{b_l\}$, $l = 1, 2, \ldots, M$, as follows.

$$\text{For } l \geq l^*, \qquad a_l = b_l = f(x, y).$$

$$\text{For } l < l^*, \quad a_l = f(x, \hat{y}), \quad b_l = f(\hat{x}, y),$$

where $\hat{x}$ (or $\hat{y}$) is a random value of x (or y) chosen by the dealer.

Inputs:
x from P_1 and y from P_2. If one of the received input is not in the correct domain, then both the parties are given $\perp$.
Computation:
The dealer does the following:
Chooses the l^* according to a geometric distribution $\mathcal{G}(\gamma)$ with parameter γ. Here l^* is the revelation round, i.e., the round in which the value of f is either $(0, 0)$ or $(1, 1)$ and $M = \omega(\log \lambda)$
Prepares a list $list_w$ of shares for each party P_w, where $w \in \{1, 2\}$ such that
  P_1 receives the values of $a_1^1, a_2^1, \ldots, a_M^1$ and $b_1^1, b_2^1, \ldots, b_M^1$.
  P_2 receives the values of $a_1^2, a_2^2, \ldots, a_M^2$ and $b_1^2, b_2^2, \ldots, b_M^2$.
Output:
$a_l = a_l^1 \oplus a_l^2$.
$b_l = b_l^1 \oplus b_l^2$.
For $l < l^*$, set $a_l = f(x, \hat{y})$.
For $l < l^*$, set $b_l = f(\hat{x}, y)$.
For $l \geq l^*$, set $a_l = a_{l^*}$.
For $l \geq l^*$, set $b_l = b_{l^*}$.
$a_1, a_2, \ldots, a_M$ and $b_1, b_2, \ldots, b_M$ correspond to the outputs of P_1 and P_2 respectively for $1 \leq l \leq M$.

Algorithm 5: ShareGen2

Next, the dealer splits the secret a_l into the shares a_l^1 and a_l^2, and the secret b_l into the shares b_l^1 and b_l^2, so that $a_l = a_l^1 \oplus a_l^2$ and $b_l = b_l^1 \oplus b_l^2$, and gives the shares $\{(a_l^1, b_l^1)\}$ to P_1 and the shares $\{(a_l^2, b_l^2)\}$ to P_2. In each round l, P_2 sends a_l^2 to P_1, who, in turn sends b_l^1 to P_2. P_1 and P_2 both learns the output value $f(x, y)$ in iteration l^*, unlike the Millionaire's problem. The algorithm for the functionality share generation in fail-stop setting is revisited in Algorithm 5. Here we assume that the dealer who will distribute the shares is honest and can compute the function described in Equation (5).

The algorithms in the Byzantine setting are the same as those in the fail-stop setting except some additional steps. In Byzantine setting, the shares are signed by the dealer. The signing message distribution procedure is same as Section 3.

The protocol for computing f is described in Algorithm 6.

Inputs:
P_1 obtains $a_1^1, a_2^1, \ldots, a_M^1$ and $b_1^1, b_2^1, \ldots, b_M^1$.
P_2 obtains $a_1^2, a_2^2, \ldots, a_M^2$ and $b_1^2, b_2^2, \ldots, b_M^2$.
Computation:
There are M number of iterations. In each iteration $l \in \{1, 2, \ldots, M\}$ do:
P_2 sends a_l^2 to P_1 and P_1 computes $a_l = a_l^1 \oplus a_l^2$.
P_1 sends b_l^1 to P_2 and P_2 computes $b_l = b_l^1 \oplus b_l^2$.
Output:
If P_2 aborts in round l, i.e., does not send its share at that round and $l \leq l^*$, P_1 outputs $a_{l-1} = f(x, \hat{y})$. If $l > l^*$, P_1 has already determined the output in some earlier iteration. Thus it outputs that value.
If P_1 aborts in round l, i.e., P_1 computes its output and does not send its share at that round and $l \leq l^*$, P_2 outputs $b_l = f(\hat{x}, y)$. If $l > l^*$, P_2 has already determined the output in some earlier iteration. Thus it outputs that value.

Algorithm 6: $\Pi^{\mathbf{CEP2}}$

4.1 $\Pi^{\mathbf{CEP2}}$ is not fair when players are rational

In this subsection, we analyze the fairness condition of the function in rational setting. We assume that the players, P_1 and P_2 have the preferences $\mathcal{R}_1$.

4.1.1 Early abort by P_2

Let us first assume that P_2 be corrupted by a probabilistic polynomial time adversary $\mathcal{A}$ and chooses to abort in the round $l \leq l^*$. Let U_2 be the utility of P_2 when he aborts. We have two cases depending on P_2's choice of y.

4.1.1.1 Case 1: $y = y_1$

Thus, $\Pr(b_{l-1} = 0 | y = y_1) = \Pr(\hat{x} = x_1) = \frac{1}{3}$ and $\Pr(b_{l-1} = 1 | y = y_1) = \Pr(\hat{x} \in \{x_2, x_3\}) = \frac{2}{3}$.

Under this case, three different subcases are possible depending on P_1's choice of x.

Subcase 1.(a): $x = x_1$. Now, $\Pr(a_{l-1} = 0 | x = x_1) = \Pr(\hat{y} = y_1) = \frac{1}{2}$ and $\Pr(a_{l-1} = 1 | x = x_1) = \Pr(\hat{y} = y_2) = \frac{1}{2}$. The following table enumerates the different possibilities for U_2 when $x = x_1$ and $y = y_1$.

(a_{l-1}, b_{l-1})	U_2	Probability
(0,0)	U_2^{TT}	$\frac{1}{2} \cdot \frac{1}{3} = \frac{1}{6}$
(0,1)	U_2^{NT}	$\frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$
(1,0)	U_2^{TN}	$\frac{1}{2} \cdot \frac{1}{3} = \frac{1}{6}$
(1,1)	U_2^{NN}	$\frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$

Thus, $E[U_2 | (x_1, y_1)] = \left[\frac{1}{6}(U_2^{TN} + U_2^{TT}) + \frac{1}{3}(U_2^{NT} + U_2^{NN})\right]$.

Subcase 1.(b): $x = x_2$. Now, $\Pr(a_{l-1} = 0|x = x_2) = \Pr(\hat{y} = y_2) = \frac{1}{2}$ and $\Pr(a_{l-1} = 1|x = x_2) = \Pr(\hat{y} = y_1) = \frac{1}{2}$.

The following table enumerates the different possibilities for U_2 when $x = x_2$ and $y = y_1$.

(a_{l-1}, b_{l-1})	U_2	Probability
(0,0)	U_2^{NN}	$\frac{1}{2} \cdot \frac{1}{3} = \frac{1}{6}$
(0,1)	U_2^{TN}	$\frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$
(1,0)	U_2^{NT}	$\frac{1}{2} \cdot \frac{1}{3} = \frac{1}{6}$
(1,1)	U_2^{TT}	$\frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$

Thus, $E[U_2|(x_1, y_2)] = \Big[\frac{1}{6}(U_2^{NN} + U_2^{NT}) + \frac{1}{3}(U_2^{TN} + U_2^{TT})\Big]$.

Subcase 1.(c): $x = x_3$. In this case, P_1 knows the output with certainty. That means, Now, $\Pr(a_{l-1} = 0|x = x_3) = 0$ and $\Pr(a_{l-1} = 1|x = x_3) = 1$.

The following table enumerates the different possibilities for U_2 when $x = x_3$ and $y = y_1$.

(a_{l-1}, b_{l-1})	U_2	Probability
(0,0)	U_2^{NN}	$0 \cdot \frac{1}{3} = 0$
(0,1)	U_2^{TN}	$0 \cdot \frac{2}{3} = 0$
(1,0)	U_2^{NT}	$1 \cdot \frac{1}{3} = \frac{1}{3}$
(1,1)	U_2^{TT}	$1 \cdot \frac{2}{3} = \frac{2}{3}$

Thus, $E[U_2|(x_3, y_1)] = \frac{1}{3}U_2^{NT} + \frac{2}{3}U_2^{TT}$.

Now, combining all three subcases, we get

$$\begin{aligned}
E[U_2|y_1] &= E[U_2|(x_1,y_1)] \cdot \Pr(x = x_1) + E[U_2|(x_2,y_1)] \cdot \Pr(x = x_2) + E[U_2|(x_3,y_1)] \cdot \Pr(x = x_3) \\
&= \Big[\frac{1}{6}(U_2^{TN} + U_2^{TT}) + \frac{1}{3}(U_2^{NT} + U_2^{NN})\Big] \cdot \frac{1}{3} + \Big[\frac{1}{6}(U_2^{NN} + U_2^{NT}) + \frac{1}{3}(U_2^{TN} + U_2^{TT})\Big] \cdot \frac{1}{3} \\
&\quad + \Big[\frac{1}{3}U_2^{NT} + \frac{2}{3}U_2^{TT}\Big] \cdot \frac{1}{3} \\
&= \frac{1}{18}\Big[3U_2^{TN} + 7U_2^{TT} + 3U_2^{NN} + 5U_2^{NT}\Big].
\end{aligned}$$

If the above expression is greater than U_2^{TT}, P_2 aborts early, otherwise he plays the game.

4.1.1.2 Case 2: $y = y_2$

The analysis is similar and we obtain the same expression for $E[U_2|y_2]$. More specifically, we have the following observation.

Subcase 2.(a): $x = x_1$. The analysis is exactly identical to Subcase 1.(b).

Subcase 2.(b): $x = x_2$. The analysis is exactly identical to Subcase 1.(a).

Subcase 2.(c): $x = x_3$. The analysis is exactly identical to Subcase 1.(c).

4.1.2 Early abort by P_1

Now, we consider the aborting of P_1. We assume that there is a probabilistic polynomial time adversary $\mathcal{A}$ who corrupts P_1 and makes P_1 to choose abort in round l. Let U_1 be the utility of P_1 when he aborts. We have three cases depending on P_1's choice of x.

4.1.2.1 Case 1: $x = x_1$

We have $\Pr(a_l = 0|x = x_1) = \Pr(\hat{y} = y_1) = \frac{1}{2}$ and $\Pr(a_l = 1|x = x_1) = \Pr(\hat{y} = y_2) = \frac{1}{2}$, for $l < l^*$. Note that for $l = l^*$, P_1 will abort after receiving the exact value of y. Hence,

$$\text{in case of } y = y_1, \qquad \Pr(a_{l^*} = 0|(x_1, y_1)) = 1, \qquad \Pr(a_{l^*} = 1|(x_1, y_1)) = 0$$

and

$$\text{in case of } y = y_2, \qquad \Pr(a_{l^*} = 0|(x_1, y_2)) = 0, \qquad \Pr(a_{l^*} = 1|(x_1, y_2)) = 1.$$

Subcase 1.(a): $y = y_1$. Now, we have $\Pr(b_l = 0|y = y_1) = \Pr(\hat{x} = x_1) = \frac{1}{3}$ and $\Pr(b_l = 1|y = y_1) = \Pr(\hat{x} \in \{x_2, x_3\}) = \frac{2}{3}$.

The following table enumerates the different possibilities for U_1 when $x = x_1$ and $y = y_1$.

(a_l, b_l)	U_1	Probability	
		$l < l^*$	$l = l^*$
(0,0)	U_1^{TT}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{1}{3} = (1-\gamma) \cdot \frac{1}{6}$	$\gamma \cdot 1 \cdot \frac{1}{3} = \gamma \cdot \frac{1}{3}$
(0,1)	U_1^{TN}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{2}{3} = (1-\gamma) \cdot \frac{1}{3}$	$\gamma \cdot 1 \cdot \frac{2}{3} = \gamma \cdot \frac{2}{3}$
(1,0)	U_1^{NT}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{1}{3} = (1-\gamma) \cdot \frac{1}{6}$	$\gamma \cdot 0 \cdot \frac{1}{3} = 0$
(1,1)	U_1^{NN}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{2}{3} = (1-\gamma) \cdot \frac{1}{3}$	$\gamma \cdot 0 \cdot \frac{2}{3} = 0$

Thus,

$$\begin{aligned} E[U_1|(x_1, y_1)] &= (1-\gamma)\Big[\frac{1}{3}U_1^{TN} + \frac{1}{6}U_1^{TT} + \frac{1}{3}U_1^{NN} + \frac{1}{6}U_1^{NT}\Big] + \gamma\Big[\frac{2}{3}U_1^{TN} + \frac{1}{3}U_1^{TT}\Big] \\ &= \frac{(1+\gamma)}{6}\Big(2U_1^{TN} + U_1^{TT}\Big) + \frac{(1-\gamma)}{6}\Big(2U_1^{NN} + U_1^{NT}\Big). \end{aligned}$$

Subcase 1.(b): $y = y_2$. Now, we have $\Pr(b_l = 0|y = y_2) = \Pr(\hat{x} = x_2) = \frac{1}{3}$ and $\Pr(b_l = 1|y = y_2) = \Pr(\hat{x} \in \{x_1, x_3\}) = \frac{2}{3}$.

The following table enumerates the different possibilities for U_1 when $x = x_1$ and $y = y_2$.

(a_l, b_l)	U_1	Probability	
		$l < l^*$	$l = l^*$
(0,0)	U_1^{NN}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{1}{3} = (1-\gamma) \cdot \frac{1}{6}$	$\gamma \cdot 0 \cdot \frac{1}{3} = 0$
(0,1)	U_1^{NT}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{2}{3} = (1-\gamma) \cdot \frac{1}{3}$	$\gamma \cdot 0 \cdot \frac{2}{3} = 0$
(1,0)	U_1^{TN}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{1}{3} = (1-\gamma) \cdot \frac{1}{6}$	$\gamma \cdot 1 \cdot \frac{1}{3} = \frac{1}{3}$
(1,1)	U_1^{TT}	$(1-\gamma) \cdot \frac{1}{2} \cdot \frac{2}{3} = (1-\gamma) \cdot \frac{1}{3}$	$\gamma \cdot 1 \cdot \frac{2}{3} = \frac{2}{3}$

$$\begin{aligned} E[U_1|(x_1, y_2)] &= (1-\gamma)\Big(\frac{1}{6}U_1^{TN} + \frac{1}{3}U_1^{TT} + \frac{1}{6}U_1^{NN} + \frac{1}{3}U_1^{NT}\Big) + \gamma\Big(\frac{1}{3}U_1^{TN} + \frac{2}{3}U_1^{TT}\Big) \\ &= \frac{(1+\gamma)}{6}\Big(U_1^{TN} + 2U_1^{TT}\Big) + \frac{(1-\gamma)}{6}\Big(U_1^{NN} + 2U_1^{NT}\Big). \end{aligned}$$

Now, combining all two subcases, we get

$$\begin{aligned} E[U_1|x_1] &= E[U_1|(x_1, y_1)] \cdot \Pr(y = y_1) + E[U_1|(x_1, y_2)] \cdot \Pr(y = y_2) \\ &= \Big[\frac{(1+\gamma)}{6}\Big(2U_1^{TN} + U_1^{TT}\Big) + \frac{(1-\gamma)}{6}\Big(2U_1^{NN} + U_1^{NT}\Big)\Big] \cdot \frac{1}{2} \\ &\quad + \Big[\frac{(1+\gamma)}{6}\Big(U_1^{TN} + 2U_1^{TT}\Big) + \frac{(1-\gamma)}{6}\Big(U_1^{NN} + 2U_1^{NT}\Big)\Big] \cdot \frac{1}{2} \\ &= \frac{1+\gamma}{4}\Big(U_1^{TN} + U_1^{TT}\Big) + \frac{1-\gamma}{4}\Big(U_1^{NN} + U_1^{NT}\Big). \end{aligned}$$

If the above expression is greater than U_1^{TT}, P_1 chooses abort.

4.1.2.2 Case 2: $x = x_2$

The analysis is similar and we obtain the same expression for $E[U_1|x_2]$. More specifically, we have the following observation.
Subcase 2.(a): $y = y_1$. The analysis is exactly identical to Subcase 1.(b).
Subcase 2.(b): $y = y_2$. The analysis is exactly identical to Subcase 1.(a).

4.1.2.3 Case 3: $x = x_3$

In this case, P_1 has no incentive to play as he knows in certainty that the output should be 1. For any $l \leq l^*$, P_1 always has expected utility $\left[\frac{2}{3}U_1^{TT} + \frac{1}{3}U_1^{TN}\right]$, which is always greater than U_1^{TT}. Thus, if P_1 chooses x_3, he always aborts early and fairness can not be achieved.

4.1.3 Summary of the analysis

From the above analysis, it is clear that aborting of P_2 does not affect fairness. If P_2 aborts and $l \leq l^*$, then no one obtains the output. However, if $l > l^*$, then both obtain the output. Contrary to this, aborting of P_1 affects fairness, as he computes the output first from the input received from P_2. When $x = x_3$, P_1 should have no incentive to continue the game as he knows the output with certainty. Thus, we have the following result.

Theorem 5. . *The protocol* $\mathbf{\Pi^{CEP2}}$ *cannot achieve fairness with rational players.*

4.2 How to make $\mathbf{\Pi^{CEP2}}$ fair when players are rational

In this subsection we suggest a variant of Gordon's protocol with fairness in the presence of a rational adversary. Here, we only modify the step 6 of Algorithm 6: $\mathbf{\Pi^{CEP2}}$, and call the resulting protocol $\mathbf{\Pi^{CEP2}_{Fair}}$. When P_1 aborts in any round l, instead of $f(\hat{x}, y)$, P_2 outputs 1. Every other steps are remain same. We now prove the fairness of the protocol.

4.2.1 Early abort by P_2

The analysis in this case is exactly identical to Section 4.1.1. Thus, for fairness, we need to ensure that

$$\frac{1}{18}\left[3U_2^{TN} + 7U_2^{TT} + 3U_2^{NN} + 5U_2^{NT}\right] \leq U_2^{TT},$$

i.e.,

$$U_2^{TT} \geq \frac{1}{11}\left[3U_2^{TN} + 3U_2^{NN} + 5U_2^{NT}\right]. \tag{6}$$

4.2.2 Early abort by P_1

Now, we discuss each case one by one.

4.2.2.1 Case 1: $x = x_1$

We have $\Pr(a_l = 0|x = x_1) = \Pr(\hat{y} = y_1) = \frac{1}{2}$ and $\Pr(a_l = 1|x = x_1) = \Pr(\hat{y} = y_2) = \frac{1}{2}$, for $l < l^*$. Note that for $l = l^*$, P_1 will abort after receiving the exact value of y. Hence,

$$\text{in case of } y = y_1, \qquad \Pr(a_{l^*} = 0|(x_1, y_1)) = 1, \qquad \Pr(a_{l^*} = 1|(x_1, y_1)) = 0$$

and

$$\text{in case of } y = y_2, \qquad \Pr(a_{l^*} = 0|(x_1, y_2)) = 0, \qquad \Pr(a_{l^*} = 1|(x_1, y_2)) = 1.$$

Subcase 1.(a): $y = y_1$. Now, we have $\Pr(b_l = 0|y = y_1) = 0$ and $\Pr(b_l = 1|y = y_1) = 1$.

The following table enumerates the different possibilities for U_1 when $x = x_1$ and $y = y_1$.

(a_l, b_l)	U_1	Probability	
		$l < l^*$	$l = l^*$
(0,0)	U_1^{TT}	$(1-\gamma) \cdot \frac{1}{2} \cdot 0 = 0$	$\gamma \cdot 1 \cdot 0 = 0$
(0,1)	U_1^{TN}	$(1-\gamma) \cdot \frac{1}{2} \cdot 1 = (1-\gamma) \cdot \frac{1}{2}$	$\gamma \cdot 1 \cdot 1 = \gamma \cdot 1$
(1,0)	U_1^{NT}	$(1-\gamma) \cdot \frac{1}{2} \cdot 0 = (1-\gamma) \cdot 0$	$\gamma \cdot 0 \cdot 0 = 0$
(1,1)	U_1^{NN}	$(1-\gamma) \cdot \frac{1}{2} \cdot 1 = (1-\gamma) \cdot \frac{1}{2}$	$\gamma \cdot 0 \cdot 1 = 0$

Thus,

$$\begin{aligned} E[U_1|(x_1, y_1)] &= (1-\gamma)\Big[\frac{1}{2}U_1^{TN} + \frac{1}{2}U_1^{NN}\Big] + \gamma\Big[U_1^{TN}\Big] \\ &= \frac{(1+\gamma)}{2}\Big(U_1^{TN}\Big) + \frac{(1-\gamma)}{2}\Big(U_1^{NN}\Big). \end{aligned}$$

Subcase 1.(b): $y = y_2$. Now, we have $\Pr(b_l = 0|y = y_2) = 0$ and $\Pr(b_l = 1|y = y_2) = 1$.

The following table enumerates the different possibilities for U_1 when $x = x_1$ and $y = y_2$.

(a_l, b_l)	U_1	Probability	
		$l < l^*$	$l = l^*$
(0,0)	U_1^{NN}	$(1-\gamma) \cdot \frac{1}{2} \cdot 0 = (1-\gamma) \cdot 0$	$\gamma \cdot 0 \cdot 0 = 0$
(0,1)	U_1^{NT}	$(1-\gamma) \cdot \frac{1}{2} \cdot 1 = (1-\gamma) \cdot \frac{1}{2}$	$\gamma \cdot 0 \cdot 1 = 0$
(1,0)	U_1^{TN}	$(1-\gamma) \cdot \frac{1}{2} \cdot 0 = (1-\gamma) \cdot 0$	$\gamma \cdot 1 \cdot 0 = 0$
(1,1)	U_1^{TT}	$(1-\gamma) \cdot \frac{1}{2} \cdot 1 = (1-\gamma) \cdot \frac{1}{2}$	$\gamma \cdot 1 \cdot 1 = \gamma$

$$\begin{aligned} E[U_1|(x_1, y_2)] &= (1-\gamma)\Big(\frac{1}{2}U_1^{TT} + \frac{1}{2}U_1^{NT}\Big) + \gamma\Big(U_1^{TT}\Big) \\ &= \frac{(1+\gamma)}{2}\Big(U_1^{TT}\Big) + \frac{(1-\gamma)}{2}\Big(U_1^{NT}\Big). \end{aligned}$$

Now, combining all two subcases, we get

$$\begin{aligned} E[U_1|x_1] &= E[U_1|(x_1, y_1)] \cdot \Pr(y = y_1) + E[U_1|(x_1, y_2)] \cdot \Pr(y = y_2) \\ &= \Big[\frac{(1+\gamma)}{2}\Big(U_1^{TN}\Big) + \frac{(1-\gamma)}{2}\Big(U_1^{NN}\Big)\Big] \cdot \frac{1}{2} + \Big[\frac{(1+\gamma)}{2}\Big(U_1^{TT}\Big) + \frac{(1-\gamma)}{2}\Big(U_1^{NT}\Big)\Big] \cdot \frac{1}{2} \\ &= \frac{(1+\gamma)}{4}\Big(U_1^{TN} + U_1^{TT}\Big) + \frac{(1-\gamma)}{4}\Big(U_1^{NN} + U_1^{NT}\Big). \end{aligned}$$

If the above expression is greater than U_1^{TT}, P_1 chooses abort. Thus, for fairness, we need to ensure that $U_1^{TT} \geq \frac{(1+\gamma)}{4}\Big(U_1^{TN} + U_1^{TT}\Big) + \frac{(1-\gamma)}{4}\Big(U_1^{NN} + U_1^{NT}\Big)$, i.e.,

$$\gamma \leq \frac{3U_1^{TT} - U_1^{TN} - U_1^{NN} - U_1^{NT}}{U_1^{TN} + U_1^{TT} - U_1^{NN} - U_1^{NT}}. \tag{7}$$

4.2.2.2 Case 2: $x = x_2$

The analysis is similar and we obtain the same expression for $E[U_1|x_2]$. More specifically, we have the following observation.
Subcase 2.(a): $y = y_1$. The analysis is exactly identical to Subcase 1.(b).
Subcase 2.(b): $y = y_2$. The analysis is exactly identical to Subcase 1.(a).

4.2.2.3 Case 3: $x = x_3$

When $x = x_3$, P_1 will abort as he knows the output with certainty. In this case, he needs no help from P_2 to compute the function. However, when P_1 chooses to abort, P_2 outputs 1. Thus, for $x = x_3$, both get the correct output of the function. The utility for both the player is U_w^{TT}, $w \in \{1, 2\}$. Hence, the fairness condition in rational setting is always maintained.

4.2.3 Fairness condition

From the above analysis, we can state the following result.

Theorem 6. *Provided $\mathcal{R}_1$, $U_2^{TT} \geq \frac{1}{11}\left[3U_2^{TN} + 3U_2^{NN} + 5U_2^{NT}\right]$, $(U_1^{TT} - U_1^{NN}) + (U_1^{TT} - U_1^{NT}) > (U_1^{TN} - U_1^{TT})$, and*

$$0 < \gamma \leq \frac{3U_1^{TT} - U_1^{TN} - U_1^{NN} - U_1^{NT}}{U_1^{TN} + U_1^{TT} - U_1^{NN} - U_1^{NT}},$$

the protocol $\mathbf{\Pi}_{\mathbf{Fair}}^{\mathbf{CEP2}}$ achieves fairness.

Proof. The proof follows from Equations (6) and (7). The additional condition

$$(U_1^{TT} - U_1^{NN}) + (U_1^{TT} - U_1^{NT}) > (U_1^{TN} - U_1^{TT}) \tag{8}$$

follows from the fact that for γ to be meaningful, the numerator $3U_1^{TT} - U_1^{TN} - U_1^{NN} - U_1^{NT}$ must be ≥ 0. Further, from the condition $\gamma \leq \frac{3U_1^{TT} - U_1^{TN} - U_1^{NN} - U_1^{NT}}{U_1^{TN} + U_1^{TT} - U_1^{NN} - U_1^{NT}}$, it is easy to see that the natural restriction $\gamma \leq 1$ always holds. □

In Equation (8), all the three terms within the parentheses are non-negative according to $\mathcal{R}_1$. Again, the condition $U_2^{TT} \geq \frac{1}{11}\left[3U_2^{TN} + 3U_2^{NN} + 5U_2^{NT}\right]$ can be re-written as

$$3(U_2^{TT} - U_2^{NN}) + 5(U_2^{TT} - U_2^{NT}) \geq 3(U_2^{TN} - U_2^{TT}). \tag{9}$$

If the utilities are symmetric, i.e., if $U_1^{xy} = U_2^{xy}$, then Equation (8) implies Equation (9), and hence we need one less condition. The following corollary is immediate.

Corollary 1. . *Provided $\mathcal{R}_1$, $(U^{TT} - U^{NN}) + (U^{TT} - U^{NT}) > (U^{TN} - U^{TT})$, and*

$$0 < \gamma \leq \frac{3U^{TT} - U^{TN} - U^{NN} - U^{NT}}{U^{TN} + U^{TT} - U^{NN} - U^{NT}},$$

the protocol $\mathbf{\Pi}_{\mathbf{Fair}}^{\mathbf{CEP2}}$ with symmetric utilities achieves fairness.

4.3 Fairness analysis of $\mathbf{\Pi^{CEP2}}$ when players have equal domain sizes

In rational setting, the analysis of the original $\mathbf{\Pi^{CEP2}}$ protocol [7, 8], is exactly the same as in Section 4.1 except that the cases corresponding to x_3 would not be there. In this situation, the protocol need not be modified. In order to maintain fairness, we keep the original steps of [7, 8], as in Algorithm 6, and Theorem 6 guarantees fairness. Note that the fairness condition is the same for unequal as well as equal domain sizes.

5 Conclusion

In this paper, we revisit the 'greater than' function proposed by Gordon et al. [7, 8] which serves the goal of millionaires' problem. We observed that the protocol for computing the function suggested by Gordon et al. no longer remains fair in the presence of the rational players having some specific set of utilities. We proposed a variant of this protocol that can compute the function and hence any function without embedded XOR with fairness and correctness.

We also revisit the equality problem of [8], that is an instance of the embedded XOR class. We show that in rational domain it no longer remains fair and then we propose a variant that achieves fairness when the players are rational.

Ours is the first attempt to study the above two problems in rational domain.

References

[1] G. Asharov, Y. Lindell. Utility Dependence in Correct and Fair Rational Secret Sharing. *Journal of Cryptology.* **24**, 157–202, (2010).

[2] G. Asharov, R. Canetti, C. Hazay. Towards a Game Theoretic View of Secure Computation. EUROCRYPT 2011, LNCS **6632**, 426–445, (2011).

[3] M. Ben-Or, S. Goldwasser and A. Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. *Proc. ACM STOC 1988*, (STOC), 1–10, (1988).

[4] D. Chaum, C. Crépeau, I. Damgard. Multi-party unconditionally secure protocols. *Proc. ACM STOC 1988*, (STOC), 11–19, (1988).

[5] R. Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract) *Proceedings of the 18th Annual ACM symposium on Theory of Computing* (STOC), 364–369, ACM Press, (1986).

[6] G. Fuchsbauer, J. Katz, D. Naccache. Efficient Rational Secret Sharing in Standard Communication Networks. *Proceedings of the 7th Conference on Theory of Cryptography*, 419–436, Springer-Verlag, (2010).

[7] S. D. Gordon, C. Hazay, J. Katz, Y. Lindell. Complete Fairness in Secure Two-Party Computation. *Proceedings of 40th Annual ACM symposium on Theory of Computing* (STOC), 413–422, ACM Press, (2008)

[8] S. D. Gordon, C. Hazay, J. Katz, Y. Lindell. Complete Fairness in Secure Two-Party Computation. *Journal of the ACM (JACM)* **58**, Issue 6, December 2011.

[9] A. Groce, J. Katz. Fair computation with rational players. EUROCRYPT 2012, LNCS **7237**, 81–98, Springer (2012).

[10] G. Kol, M. Naor. Games for exchanging information. *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, 423–432, (2008).

[11] Y. Lindell. Composition of Secure Multi-Party Protocols, A Comprehensive study. Springer-Verlag, Berlin, (2003).

[12] John von Neumann, Oskar Morgenstern. Theory of Games and Economic Behavior. Princeton University Press, 1944.

[13] A. C. Yao. Protocols for secure computations. *23rd Annual Symposium on Foundations of Computer Science* (FOCS), 160–164, (1982).