Towards a Game Theoretic View of Secure Computation

Gilad Asharov[*] Ran Canetti[†] Carmit Hazay[‡]

July 3, 2015

Abstract

We demonstrate how Game Theoretic concepts and formalism can be used to capture cryptographic notions of security. In the restricted but indicative case of two-party protocols in the face of malicious fail-stop faults, we first show how the traditional notions of secrecy and correctness of protocols can be captured as properties of Nash equilibria in games for rational players. Next, we concentrate on fairness. Here we demonstrate a Game Theoretic notion and two different cryptographic notions that turn out to all be equivalent. In addition, we provide a simulation-based notion that implies the previous three. All four notions are weaker than existing cryptographic notions of fairness. In particular, we show that they can be met in some natural setting where existing notions of fairness are provably impossible to achieve.

[*] Department of Computer Science, Bar-Ilan University, Israel. Supported by the European Research Council as part of the ERC project LAST. asharog@cs.biu.ac.il

[†] Department of Computer Science, Tel-Aviv University, Israel. Supported by the Check Point Institute for Information Security, BSF, ISF, and Marie Curie grants. canetti@tau.ac.il

[‡] Faculty of Engineering, Bar-Ilan University, Israel. carmit.hazay@biu.ac.il

Contents

1 Introduction

Both Game Theory and the discipline of cryptographic protocols are dedicated to understanding the intricacies of collaborative interactions among parties with conflicting interests. Furthermore, the focal point of both disciplines is the same, and is algorithmic at nature: designing and analyzing algorithms for parties in such collaborative situations. However, the two disciplines developed very different sets of goals and formalisms. Cryptography focuses on designing algorithms that allow those who follow them to interact in a way that guarantees some basic concrete properties, such as secrecy, correctness or fairness, in face of adversarial, malicious behavior. Game Theory is more open-ended, concerning itself with understanding algorithmic behaviors of "rational" parties with well-defined goals in a given situation, and on designing rules of interaction that will "naturally" lead to behaviors with desirable properties.

Still, in spite of these differences, some very fruitful cross fertilization between the two disciplines has taken place (see e.g. [28, 9]). One very natural direction is to use cryptographic techniques to solve traditional Game Theoretic problems. In particular, the works of Dodis et al. [8], Ismalkov et al. [27, 26], Abraham et al. [1] and Halpern and Pass [25] take this path and demonstrate how a mutli-party protocol using cryptographic techniques can be used to replace a trusted correlation device or a mediator in mechanism design. Another line of research is to extend the traditional Game Theoretic formalisms to capture, within the context of Game Theory, cryptographic concerns and ideas that take into account the fact that protocol participants are computationally bounded, and that computational resources are costly [8, 25, 22].

Yet another line of work is aimed at using Game Theoretic concepts and approach to amend traditional cryptographic goals such as secure and fair computation. A focal point in this direction has been the concept of rational fair exchange of secrets (also known as *rational secret sharing*) [24, 20, 31, 29, 30, 33, 10, 3]. Here the goal is to design a protocol for exchanging secrets in a way that "rational players" will be "interested" in following the protocol, where it is assumed that players are interested in learning the secret inputs of the other players while preventing others from learning their own secrets. In fact, it is assumed that the participants have specific preferences and some quantitative prior knowledge on these preferences of the participants is known to the protocol designer. Furthermore, such prior knowledge turns out to be essential in order to get around basic impossibility results [3, 6].

These ingenious works demonstrate the benefit in having a joint theory of protocols for collaborative but competing parties; but at the same time they underline the basic incompatibility in the two formalisms. For instance, the (primarily Game Theoretic) formalisms used in the works on rational secret sharing do not seem to naturally capture basic cryptographic concepts, such as semantic security of the secrets. Instead, these works opt for more simplistic notions that are not always compatible with traditional cryptographic formalisms. In particular, existing modeling (that is used both by constructions and by impossibility results) treats the secret as an atomic unit and consider only the case where the parties either learnt or did not learn the secret *entirely.* Unlike traditional cryptographic modeling, the option where *partial information* about the secret is leaked through the execution is disregarded.

This work. We relate the two *formalisms.* In particular we show how Game Theoretic formalism and concepts can be used to capture traditional cryptographic security properties of protocols. We concentrate on the setting of two-party protocols and fail-stop adversaries. While this setting is admittedly limited, it does incorporate the core aspects of secrecy, correctness and fairness in face of malicious (i.e., not necessarily "rational") aborts.

In this setting, we first show Game Theoretic notions of secrecy and correctness that are *equivalent,* respectively, to the standard cryptographic notions of secret and correct evaluation of deterministic functions in the fail-stop setting (see e.g [13]). We then turn to capturing fairness. Here the situation turns out to be more intricate. We formulate a natural Game Theoretic notion of fairness, and observe that it is *strictly weaker* than existing cryptographic notions of fair two-party function evaluation. We then formulate new cryptographic notions of fairness that are *equivalent* to this Game Theoretic notion, and a simulation-based notion of fairness that implies the above three. Furthermore, we show that these new notions can indeed be realized in some potentially meaningful settings where traditional cryptographic notions are provably unrealizable.

Our results in more detail. The basic idea proceeds as follows. We translate a given protocol into a set of games, in such a way that the protocol satisfies the cryptographic property in question *if and only if* a certain pair of strategies (derived from the protocol) are in a (computational) Nash equilibrium in each one of the games. This allows the cryptographic question to be posed (and answered) in Game Theoretic language. More precisely, given a protocol, we consider the (extensive form with incomplete information) game where in each step the relevant party can decide to either continue running the protocol as prescribed, or alteratively abort the execution. We then ask whether the pair of strategies that instruct the players to continue the protocol to completion is in a (computational) Nash equilibrium. Each cryptographic property is then captured by an appropriate set of utilities and input distributions (namely, distributions over the types). In particular:

Secrecy. A given protocol is secret (as in, e.g. [13]) if and only if the strategy that never aborts the protocol is in a computational Nash equilibrium with respect to the following set of utilities and distributions over the types. For each pair of values in the domain, we define a distribution that chooses an input for one party at random from the pair. The party gets low payoff if the two values lead to the same output value and yet the other party managed to guess which of the two inputs was used. It is stressed that this is the first time where a traditional cryptographic notion of secrecy (in the style of [17]) is captured in Game Theoretic terms. In particular, the works on rational secret sharing do not provide this level of secrecy for the secret. (Indeed, the solution approaches taken there need the secret to be taken from a large domain.)

Correctness. A protocol correctly computes a deterministic function if and only if the strategy that never aborts the protocol is in a computational Nash equilibrium with respect to the set of utilities where the parties get high payoff only if they output the correct function value on the given inputs (types), or abort before the protocol starts; in addition, the players get no payoff for incorrect output.

Fairness. Here we make the bulk of our contributions. We first recall the basic setting: Two parties interact by exchanging messages in order to evaluate a function f on their inputs. The only allowed deviation from the protocol is abortion, in which event both parties learn that the protocol was aborted. Consequently, a protocol in this model should specify, in addition to the next message to be sent, also a prediction of the output value in case the execution is aborted. (Although the setting makes sense for any function, it may be helpful to keep in mind the *fair exchange* function, where the output of each party is the input of the other.)

Current notions of fairness for two party protocols in this model (e.g., [19, 21]) require there to be a point in the computation where both parties move from a state of no knowledge of the output to a full knowledge of it. This is a strong notion, which is impossible to realize in many scenarios. Instead, we would like to investigate more relaxed notions of fairness, which allow parties to gradually learn partial information on their desired outputs - but do so in a way that is "fair". Indeed, such an approach seems reasonable both from a Game Theoretic point of view (as a zero-sum game) and from a cryptographic point of view via the paradigm of gradual release (see e.g. [4, 16, 12, 21, 3] and the references within).

A first thing to note about such a notion of fairness is that it is sensitive to the potential prior knowledge that the parties may have on each other's inputs. Indeed, a "gradual release" protocol that is "fair" without prior knowledge may become "unfair" in a situation where one of the parties has far more knowledge about the possible values of the inputs of the second party than vice versa.

We thus explicitly model in our security notions the knowledge that each party has on the input of the other party. That is, we let each party has, in addition to its own input, some additional information on the input of the other party. Furthermore, to simplify matters and put the two parties on equal footing, we assume that the information that the parties have on the input of the other consists of two possible values for that input. That is, each party receives three values: its own input, and two possible values for the input of the other party. Indeed, such information naturally captures situations where the domain of possible inputs is small (say, binary). The formalism can also be naturally extended to deal with domains of small size which is larger than two.

We first sketch our Game Theoretic notion. We consider the following set of distributions over inputs (types): Say that a quadruple of elements (a_0, a_1, b_0, b_1) in the domain of function f is *valid* if for all $i \in \{0,1\}$, $f(a_0, b_i) \neq f(a_1, b_i)$ and $f(a_i, b_0) \neq f(a_i, b_1)$. For each valid quadruple of values in the domain, we define a distribution that chooses an input for one party at random from the first two values, and an input for other party at random from the other two values. The utility function for a party is the following: when the party aborts the protocol, each party predicts its output. If the party predicts correctly and the other one does not, then it gets payoff +1. If it predicts incorrectly and the other party predicts correctly then it gets payoff -1. Else, it gets payoff 0. We say that a protocol is *Game Theoretically fair* if the strategy that never aborts the protocol is in a computational Nash equilibrium with respect to the above utility, applied to both parties, and any distribution from the above family.

This formalism is perhaps the most natural way to define fairness using Game Theoretic tools, yet, it also provides a very different point of view of fairness than the traditional cryptographic one. Specifically, in cryptography fairness is usually captured by requiring that in almost each execution, either both parties learn the secret, or neither of them does. In contrast, our new notion of fairness examines the protocol in a broader sense and requires that no party obtains an advantage. In particular, even if the protocol enables one of the parties to gain an advantage in some execution, it may still be considered "fair" if it balances this advantage and gives an equivalent superiority to the other party in subsequent executions. As a result these cross advantages are eliminated and the parties are in an equilibrium state. This interesting notion of fairness may give hope for a much wider class of functions for which the standard cryptographic notion is impossible to achieve. In this work we explore the feasibility of this new notion of cryptographic fairness, while proposing an appropriate equivalent definition that captures this Game Theory fairness using the conventional cryptographic language. Specifically, we consider three different cryptographic notions of fairness and study their relationships with the above Game Theoretic notion; see Figure 1 for an illustration.

- First, we formulate a simple "game based" notion of fairness that limits the gain of an arbitrary (i.e., not necessarily "rational") fail-stop adversary in a game that closely mimics the above Game Theoretic interaction. The main difference between the notions is that in the cryptographic setting the adversary is arbitrary, rather than rational. Still, we show that the two notions are *equivalent*. Specifically, we consider a test for the protocol in a "fair" environment, where each party has two possible inputs and its effective input is chosen uniformly at random from this set. Moreover, both parties know the input tuple and the distribution over the inputs. This is due to the fact that we, untraditionally, assume that the protocol instructs the honest party to guess the output of the function based on its view and the information it recorded so far. By doing so, we are able to capture scenarios for which both parties learn the same incomplete information (in a computationally indistinguishable sense) regarding their output. In particular, as long as both parties hold the same partial information, with the same probability, then the protocol is fair. Below we further explore the feasibility of our new notion and present a protocol that realizes it under some restrictions.

- Next, we show that this notion in fact corresponds to a natural new concept of *limited gradual release*. That is, say that a protocol satisfies the limited gradual release property if at any round the probability of any party to predict its output increases only by a negligible amount. We show that a protocol is fair (as in the above notions) *if and only if* it satisfies the limited gradual release property. We note that the traditional notion of gradual release (where the predictions are non-negligibly increased, unlike our notion studied here), is in essence the basis of the classic protocols of Beaver and Goldwasser and Levin [4, 16].

- Then, we formulate an ideal-model based notion of fairness that allows for limited gradual release of secrets. In this notion the ideal functionality accepts a "sampling algorithm" M from the ideal-model adversary. The functionality then obtains the inputs from the parties and runs M on these inputs, and obtains from M the outputs that should be given to the two parties. The functionality then makes the respective outputs available to the two parties. (I.e., once the outputs are available, the parties can access them at any time.) The correctness and fairness guarantees of this interaction clearly depend on the properties of M. We thus require that M be both "fair" and "correct" in the sense that both parties get correct output with roughly equal (and substantial) probability. We then show that the new simulation-based definition implies the our limited gradual release notion. (We note that the converse does not necessarily hold with respect to secure computation in the fail-stop model, even disregarding fairness).

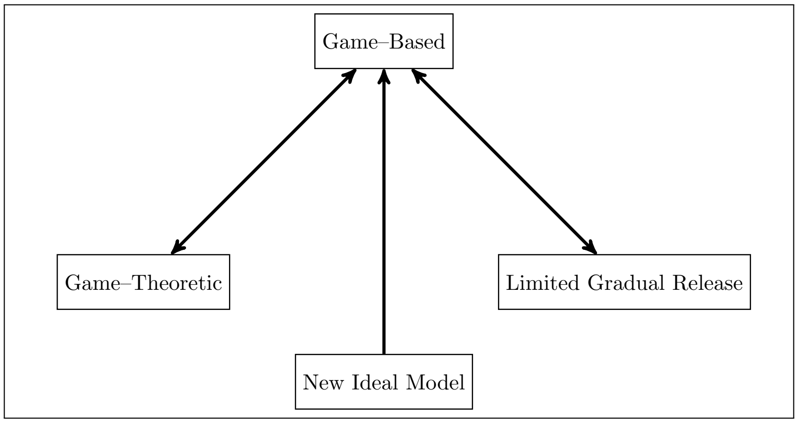

Figure 1: Our four notions of fairness and their relationships

On the feasibility of game-based fairness. Finally, we consider the realizability of our game-based notion. Notably, this notion is *strictly weaker* than the conservative cryptographic notion of fairness, as any protocol that is fair with respect to the latter definition is also fair with respect to our new notion. On the other hand, known impossibility results, such as of Cleve [6] and Asharov-Lindell [3], hold even with respect to this weaker notion, as long as *both parties are required to receive an output.* This demonstrates the meaningfulness of our new concept, as well as the benefits of translating definitions from one field to another. In particular, ruling out the feasibility of our Game Theoretic notion is derived by minor adaptations of *existing* impossibility results in cryptography.

We demonstrate it by designing a protocol where only one of the parties learn the correct output (even when both parties are honest), while guaranteeing that the expected number of times that the honest party learns its correct output is negligibly far from the number of times an arbitrary fail-stop adversary learns its output. This restricted setting is useful, for instance, for realizing the sampling problem where the goal is to toss two correlated coins. Interestingly, in a recent work [2] it was proven that sampling is impossible to achieve in the standard cryptographic setting in the presence of a fail-stop adversary. On the other hand, our protocol can be easily used to realise this task. We note that [2] is an independent work, yet, it provides a good demonstration of the main motivation for our work. That is, by considering reasonable game-theoretic notions of cryptographic primitives and objects (that are relaxations of the standard cryptographic notions), it is feasible to achieve meaningful realizations of tasks that are impossible according to the standard definition.

Consider a concrete application for our protocol of a repeated game where two parties have two shares of a long term secret that are used to jointly compute a session key, such that the first party that presents the session key gets a treasure. Then, the question is how can the parties collaborate fairly in order to get the treasure. This is exactly the type of scenarios captured by our protocol. Specifically, the condition under which our protocol guarantees correlation is that the parties become aware of their gain or loss only after the protocol execution is done. Therefore any strategizing done during the protocol must be done without knowing the outcome of the execution.

To conclude, we view this result as a fruitful cross fertilization between the two disciplines, where the Game Theoretic point of view provides different and new insights in cryptography. We stress that these feasibility results are indeed somewhat restricted and hard to be generalized. Nevertheless, they do point to a surprising feasibility result. It is an interesting open problem to extend this protocol to more general settings and functions.

Related work. We stress that our Game Theoretic approach of fairness implies an entirely different notion than the cryptographic notions of fairness, and is in particular different than "complete fairness" [19] where we do not allow any advantage of one party over the other. To overcome the impossibility of complete fairness, relaxed notions of "partial fairness" have been suggested in the cryptographic literature such as gradual release [5, 7, 4, 16] and optimistic exchange [32]. A notable recent example is the work of [21] that addresses the question of partial fairness using a new approach of the standard real-/ideal-world paradigm. Namely, this definition ensures fairness with probability at least $1 - 1/p$ for some polynomial $p(\cdot)$. Nevertheless, a drawback of this approach is that it allows a total break of security with probability $1/p$. In this work we demonstrate how can our approach capture some notion of partial fairness as well (where according to our approach, one party may have some limited advantage over the other party in *a collection* of executions and not necessarily in a single one). We further discuss these differences below in the main body.

The work of [23]. Following the proceedings version of our work, Groce and Katz [23] showed the following result, that demonstrates the feasibility of rational fair computation in the two-party setting in the presence of fail-stop and malicious adversaries. Specifically, they showed that any mediated game where (1) the mediator is guaranteed to either give the full function output to both parties or give nothing to both, and (2) both parties have positive expected utility from participating in the game at a Nash equilibrium, can be transformed into a protocol without a mediator, while preserving the same incentive structure and Nash equilibria.

Our work and [23] complement each other in the sense that the latter work shows that whenever the parties have a strict incentive to compute the function in the ideal world, there exists a real-world protocol for computing the function fairly. On the other hand, our result shows an example where the parties do *not* have a strict incentive to compute the function in the ideal world, and there does *not* exist a real-world protocol for computing the function fairly. Moreover, our paper shows that a feasibility result can be recovered in that specific case by relaxing correctness.

On the definitional choices. One question that comes to mind when considering our modeling is why use plain Nash equilibria to exhibit correspondence between cryptographic notions and Game Theoretic ones. Why not use, for instance, stronger notions such as Dominant Strategy, Survival Under Iterated Deletions, or Subgame Perfect equilibria. It turns out that in our setting of two party computation with fail-stop faults, Nash equilibria do seem to naturally correspond to cryptographic secure protocols. In particular, in the fail-stop case any Nash equilibrium is sub-game perfect, or in other words empty threats do not hold (see more discussion on this point in the next section).

Future work. Although considered in [23], it is still an interesting challenge to extend the modeling of this work to the Byzantine case. For one, in the Byzantine case there are multiple cryptographic notions of security, including various variants of simulation-based notions. Capturing these notions using Game Theoretic tools might shed light on the differences between these cryptographic notions. In particular, it seems that here the Game Theoretic formalism will have to be extended to capture arbitrary polynomial time strategies at each decision point.In particular, it seems likely that more sophisticated Game Theoretic solution concepts such as sub-game perfect equilibria and computational relaxations thereof [22, 34] will be needed.

Another challenge is to extend the notions of fairness presented here to address also situations where the parties have more general, asymmetric a priori knowledge on each other's inputs, and to find solutions that use minimal trust assumptions on the system. Dealing with the multi-party case is another interesting challenge.

Organization. Section 2 presents the basic model, as well as both the cryptographic and the Game Theoretic "solution concepts". In the cryptographic setting we present both ideal-model based definitions and indistinguishability-based ones. Section 3 presents our results for secrecy and correctness for deterministic functions. Section 4 presents the results regarding fairness, i.e., (**i**) the Game Theoretic notion, (**ii**) the equivalent cryptographic definition, (**iii**) a new simulation-based definition, (**iv**) the limited gradual release property and its relation to fairness and, (**v**) the study of the fairness definition.

2 The Model and Solution Concepts

We review some basic definitions that capture the way we model protocols as well as the solution concepts from Game Theory.

2.1 Cryptographic Definitions

We review some standard cryptographic definitions of security for protocols. The definitions address secrecy and correctness using the indistinguishability approach.

Negligible functions and indistinguishability. A function $\mu(\cdot)$ is negligible if for every polynomial $p(\cdot)$ there exists a value N such that for all $n > N$ it holds that $\mu(n) < \frac{1}{p(n)}$. Let $X = \{X(a,n)\}_{n \in N, a \in \{0,1\}^*}$ and $Y = \{Y(a,n)\}_{n \in N, a \in \{0,1\}^*}$ be distribution ensembles. Then, we say that X and Y are computationally indistinguishable, denoted $X \overset{c}{\equiv} Y$, if for every non-uniform probabilistic polynomial time (PPT) distinguisher D there exists a negligible function $\mu(\cdot)$ such that for all sufficiently long $a \in \{0,1\}^*$,

$$|\Pr[D(X(a,n)) = 1] - \Pr[D(Y(a,n)) = 1]| < \mu(n).$$

Protocols. Our basic object of study is a *two-party protocol,* modeled as a pair of interacting Turing machines as in [18]. We formulate both the cryptographic and the Game Theoretic concepts in terms of two-party protocols. We restrict attention to PPT machines. (To simplify the analysis, we consider machines that are polynomial in a globally known *security parameter,* rather than in the length of their inputs.)

Two-Party functions. In general, a *two-party function* is a probability distribution over functions $f : \{0,1\}^* \times \{0,1\}^* \times \mathbf{N} \to \{0,1\}^* \times \{0,1\}^*$. Here the first (second) input and output represent the input and output of the first (second) party, and the third input is taken to be the security parameter. In this work we consider the restricted model of deterministic functions. We say that a function is *efficiently invertible* if, for $i = 0, 1$, given 1^n, an input value x_i and an output value y, it is possible to compute in PPT a value x_{1-i} such that $f(x_0, x_1) = y$.

The Fail-Stop setting. The setting that we consider in this paper is that of two-party interaction in the presence of *fail-stop* faults. In this setting both parties follow the protocol specification exactly, with the exception that any one of the parties may, at any time during the computation, decide to stop, or *abort* the computation. Specifically, it means that fail-stop adversaries do not change their initial input for the execution, yet, they may arbitrarily decide on their output. As seen momentarily, the Game Theoretic and the cryptographic approaches differ in the specifics of the abortion step. However, in both cases we assume that the abortion operation is explicit and public: As soon as one party decides to abort, the other party receives an explicit notification of this fact and can act accordingly. (This is in contrast to the setting where one party decides to abort while the other party keeps waiting indefinitely to the next incoming message.) This modeling of abortion as a public operation is easily justified in a communication setting with reasonable timeouts on the communication delays. Protocols in this model should specify the output of a party in case of early abortion of the protocol. We assume that this output has a format that distinguishes this output from output that does not result from early abortion. To this end, we denote the identity of the corrupted party by i^*.

2.1.1 Cryptographic Security

We present game based definitions that capture the notions of privacy and correctness. We restrict attention to deterministic functions. By definition [13], the **View** of the ith party ($i \in \{0,1\}$)

during an execution of π on (x_0, x_1) is denoted $\mathbf{View}_{\pi,i}(x_0, x_1, n)$ and equals $(x_i, r^i, m_1^i, ..., m_t^i)$, where r^i equals the contents of the ith party's *internal* random tape, and m_j^i represents the jth message that it received.

Privacy. We begin by introducing a definition of private computation [13]. Intuitively, it means that no party (that follows the protocol) should be able to distinguish any two executions when using the same inputs and seeing the same outputs. This holds even for the case that the other party uses different inputs. More formally:

Definition 2.1 (privacy) *Let f and π be as above. We say that π* privately computes *f if the following holds:*

1. *For every non-uniform* PPT *adversary $\mathcal{A}$ that controls party P_0*

$$\left\{\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1, n)\right\}_{x_0,x_1,x_1',y,z\in\{0,1\}^*,n\in\mathbb{N}} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1', n)\right\}_{x_0,x_1,x_1',z\in\{0,1\}^*,n\in\mathbb{N}}$$

where $|x_0| = |x_1| = |x_1'|$ and $f(x_0, x_1) = f(x_0, x_1')$.

2. *For every non-uniform* PPT *adversary $\mathcal{A}$ that controls party P_1*

$$\left\{\mathbf{View}_{\pi,\mathcal{A}(z),1}(x_0, x_1, n)\right\}_{x_0,x_0',x_1,z\in\{0,1\}^*,n\in\mathbb{N}} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{View}_{\pi,\mathcal{A}(z),1}(x_0', x_1, n)\right\}_{x_0,x_0',x_1,z\in\{0,1\}^*,n\in\mathbb{N}}$$

where $|x_0| = |x_0'| = |x_1|$ and $f(x_0, x_1) = f(x_0', x_1)$.

Correctness. Recall that we distinguish between output that corresponds to successful termination of the protocol, and output generated as a result of an `abort` message. We assume that the two types have distinct formats (e.g., the second output starts with a $\perp$ sign). The correctness requirement only applies to the first type of output. More precisely:

Definition 2.2 (correctness) *Let f and π be as above. We say that π* correctly computes *f if for all sufficiently large inputs x_0 and x_1 such that $|x_0| = |x_1| = n$, we have:*

$$\Pr[\mathbf{Output}_{\pi,i} \in \{\perp \circ \{0,1\}^*, f(x_0, x_1)\}] \geq 1 - \mu(n)$$

where $\mathbf{Output}_{\pi,i} \neq \perp$ denotes the output returned by P_i upon the completion of π whenever the strategy of the parties is `continue`*, and μ is a negligible function.*

Note that, for the fail-stop setting, it holds that privacy and correctness imply simulation-based security with abort. This follows by a simple extension of the proof from [13] that states the same for semi-honest adversaries.

2.2 Game Theoretic Definitions

We review the relevant concepts from Game Theory, and the extensions needed to put these concepts on equal footing as the cryptographic concepts. (Specifically, these extensions include introducing asymptotic, computationally bounded players, and negligible error probabilities.) For simplicity, we restrict attention to the case of two-player games. Traditionally, a 2-player *(normal form, full information)* game $\Gamma = (\{A_0, A_1\}, \{u_0, u_1\})$ is determined by specifying, for each player P_i, a set A_i of possible actions and a utility function $u_i : A_0 \times A_1 \mapsto R$. Letting $A \overset{\text{def}}{=} A_0 \times A_1$, we refer to a tuple of actions $a = (a_0, a_1) \in A$ as an outcome. The utility function u_i of party P_i expresses this player's preferences over outcomes: P_i prefers outcome a to outcome a' if and only if $u_i(a) > u_i(a')$. A *strategy* σ_i for P_i is a distribution on actions in A_i. Given a strategy vector $\sigma = \sigma_0, \sigma_1$, we let $u_i(\sigma)$ be the expected utility of P_i given that all the parties play according to σ. We continue with a definition of Nash equilibria:

Definition 2.3 (Nash equilibria for normal form, complete information games) *Let $\Gamma = (\{A_0, A_1\}, \{u_0, u_1\})$ be as above, and let $\sigma = \sigma_0, \sigma_1$ be a pair of strategies as above. Then σ is in a* Nash equilibrium *if for all i and any strategy σ'_i it holds that $u_i(\sigma''_0, \sigma''_1) \leq u_i(\sigma)$, where $\sigma''_i = \sigma'_i$ and $\sigma''_{1-i} = \sigma_{1-i}$.*

The above formalism is also naturally extended to the case of *extensive form* games, where the parties take turns when taking actions. Here the strategy is a probabilistic function of a sequence of actions taken so far by the players. The execution of a game is thus represented naturally via the *history* which contains the sequence of actions taken. Similarly, the utility function of each player is applied to the history. Another natural extension is games with *incomplete information.* Here each player has an additional piece of information, called *type,* that is known only to itself. That is, the strategy σ_i now takes as input an additional value x_i. We also let the utility function depend on the types, in addition to the history. In fact, we let the utility of each player depend on the types of *both players.* This choice seems natural and standard, and greatly increases the expressibility of the model. We note however that as a result, a party cannot necessarily compute its own utility. To extend the notion of Nash equilibria to deal with this case, it is assumed that an a priori distribution on the inputs (types) is known and fixed. The expected utility of players is computed with respect to this distribution.

Definition 2.4 (Nash equilibria for extensive form, incomplete information games) *Let $\Gamma = (\{A_0, A_1\}, \{u_0, u_1\})$ be as above, and let D be a distribution over $(\{0,1\}^*)^2$. Also, let $\sigma = \sigma_0, \sigma_1$ be a pair of extensive-form strategies as described above. Then σ is in a* Nash equilibrium for *D if for all i and any strategy σ'_i it holds that $u_i(x_0, x_1, \sigma''_0(x_0), \sigma''_1(x_1)) \leq u_i(x_0, x_1, \sigma_0(x_0), \sigma_1(x_1))$, where (x_0, x_1) is taken from distribution D, $\sigma_i(x)$ denotes the strategy of P_i with type x, $\sigma''_i = \sigma'_i$ and $\sigma''_{1-i} = \sigma_{1-i}$.*

Extensions for the cryptographic model. We review the (by now standard) extensions of the above notions to the case of computationally bounded players. See e.g. [8, 28] for more details. The first step is to model a strategy as an (interactive) probabilistic Turing machine that algorithmically generates the next move given the type and a sequence of moves so far. Next, in order to capture computationally bounded behavior (both by the acting party and, more importantly, by the other party), we move to an asymptotic treatment and consider an infinite sequence of games. That is, instead of considering a single game, we consider an infinite sequence of games, one for each value of a *security parameter* $n \in \mathbb{N}$. (Formally, we give the security parameter as an additional input to each set of possible actions, to each utility function, and to the distribution over types.) The only strategies we consider are those whose runtime is polynomial in n.[1] The third and last step is to relax the notion of "greater or equal to" to "not significantly less than". This is intended to compensate for the small inevitable imperfections of cryptographic constructs. That is, we have:

[1] An alternative and more general formalism might measure the runtime of a strategy as a function of the length of its inputs. However, the present formalism is considerably simpler.

Definition 2.5 (Computational Nash equilibria for extensive form, incomplete inf. games) *Let $\Gamma = (\{A_0, A_1\}, \{u_0, u_1\})$ be as above, and let $D = \{D_n\}_{n \in \mathbf{N}}$ be a family of distributions over $(\{0,1\}^*)^2$. Let $\sigma = \sigma_0, \sigma_1$ be a pair of* PPT *extensive-form strategies as described above. Then σ is in a* Nash equilibrium for *D if for all sufficiently large n's, all i and any* PPT *strategy σ'_i it holds that $u_i(n, x_0, x_1, \sigma''_0(n, x_0), \sigma''_1(n, x_1)) \leq u_i(n, x_0, x_1, \sigma_0(n, x_0), \sigma_1(n, x_1)) + \mu(n)$, where (x_0, x_1) is taken from distribution D_n, $\sigma_i(x, n)$ denotes the strategy of P_i with type x, $\sigma''_i = \sigma'_i$ and $\sigma''_{1-i} = \sigma_{1-i}$, and μ is a negligible function.*

We remark that an alternative and viable definitional approach would use a parametric, non asymptotic definition of cryptographic security. Another viable alternative is to use the [25] notion of costly computation. We do not take these paths since our goal is to relate to standard cryptographic notions, as much as possible.

Our setting We consider the following setting: Underlying the interaction there is a two-party protocol $\pi = (\pi_0, \pi_1)$. At each step, the relevant party can make a binary decision: Either *abort* the computation, in which case the other party is notified that an abort action has been taken, or else *continue running the protocol π scrupulously.* That is, follow all instructions of the protocol leading to the generation of the next message, including making all random choices as instructed. Namely, the strategic part of the action is very simple (a mere binary choice), in spite of the fact that complex computational operations may be involved. One may envision a situation in which a human has access to a physical device that runs the protocol, and can only decide whether or not to put the device into action in each round.

The traditional Game Theoretic modeling of games involving such "exogenous" random choices that are not controllable by the players involves, introduces additional players (e.g., "Nature") to the game. In our case, however, the situation is somewhat different, since the random choices may be secret, and in addition each player also has access to local state that is preserved throughout the interaction and may affect the choices. We thus opt to capture these choices in a direct way: We first let each player have *local history.* (Initially, this local history consists only of the type of the player and its internal randomness.) The notion of an "action" is then extended to include also potentially complex algorithmic operations. Specifically, an action may specify a (potentially randomized) algorithm and a configuration. The outcome of taking this action is that an output of running the said algorithm from the said configuration, is appended to the history of the execution, and the new configuration of the algorithm is added to the local history of the player. Formally:

Definition 2.6 *Let $\pi = (P_0, P_1)$ be a two-party protocol (i.e., a pair of Interactive Turing Machines). Then, the* local history *of P_i (for $i \in \{0,1\}$), during an execution of π on input (x_0, x_1) and* internal *random tape r^i, is denoted by $\mathbf{History}_{\pi,i}(x_0, x_1, n)$ and equals $(x_i, r^i, m^i_1, \ldots, m^i_t)$, where m^i_j represents its jth message. The* history *of π during this execution is captured by $(m^0_1, m^1_1), ..., (m^0_t, m^1_t)$ and is denoted by $\mathbf{History}_\pi$. The configuration of π at some point during the interaction consists of the local configurations of P_0, P_1.*

Fail-stop games. We consider games of the form $\Gamma_{\pi,u} = (\{A_0, A_1\}, \{u_0, u_1\})$, where $A_0 = A_1 = \{\texttt{continue}, \texttt{abort}\}$. The decision is taken before the sending of each message. That is, first the program π_i is run from its current configuration, generating an outgoing message. Next, the party makes a strategic decision whether to continue or to abort. A `continue` action by player i means that the outgoing message generated by π_i is added to the history, and the new configuration is added to the local history. An `abort` action means that a special `abort` symbol is added to the configurations of both parties and then both π_0 and π_1 are run to completion, generating local outputs, and the game ends. We call such games *fail-stop games.*

The utility functions in fail-stop games may depend on all the histories: the joint one, as well as the local histories of both players. In the following sections, it will be convenient to define utility functions that consider a special field of the local history, called the *local output* of a player P_i. We denote this field by $\mathbf{Output}_{\pi,i}$. Also, denote by $\sigma^{\texttt{continue}}$ the strategy that always returns `continue`. The basic Game Theoretic property of protocols that we will be investigating is whether the pair of strategies $(\sigma^{\texttt{continue}}, \sigma^{\texttt{continue}})$ is in a (computational) Nash equilibrium in fail-stop games, with respect to a given set of utilities and input distributions. That is:

Definition 2.7 (Nash protocols) *Let $\mathcal{D}$ be a set of distribution ensembles over pairs of strings, and let $\mathcal{U}$ be a set of extensive-form binary utility functions. A two-party protocol π is called* Nash Protocol with respect to $\mathcal{U}, \mathcal{D}$ *if, for any $u \in \mathcal{U}$ and $D \in \mathcal{D}$, the pair of strategies $\sigma = (\sigma^{\texttt{continue}}, \sigma^{\texttt{continue}})$ is in a computational Nash equilibrium for the fail-stop game $\Gamma_{\pi,u}$ and distribution ensemble D.*

On subgame perfect equilibria and related solution concepts. An attractive solution concept for extensive form games (namely, interactive protocols) is subgame perfect equilibria, which allow for analytical treatment which is not encumbered by "empty threats". Furthermore, some variants of this notion that are better suited to our computational setting have been recently proposed (see [22, 34]). However, we note that in our limited case of fail-stop games any Nash equilibrium is subgame perfect. Indeed, once one of the parties aborts the computation there is no chance for the other party to "retaliate", hence empty threats are meaningless. (Recall that the output generation algorithms are not strategic, only the decision whether to abort is.)

3 Privacy and Correctness in Game Theoretic View

In this section we capture the traditional cryptographic privacy and correctness properties of protocols using Game Theoretic notions. We restrict attention to the fail-stop setting and deterministic functions with a single output. (Fairness aside, private computation of functions with two distinct outputs can be reduced to this simpler case; see [13] for more details.)

3.1 Privacy in Game Theoretic view

Our starting point is the notion of private computation. A protocol is private if no (fail-stop) PPT adversary is able to distinguish any two executions where the adversary's inputs and outputs are the same, even when the honest party uses different inputs in the two executions. Our goal, then, is to define a set of utility functions that preserve this property for Nash protocols. We therefore restrict ourselves to input distributions over triples of inputs, where the input given to one of the parties is fixed, whereas the input of the other party is uniformly chosen from the remaining pair. This restriction captures the strength of cryptographic (semantic) security: *even* if a party knows that the input of the other party can only be one out of two possible values, the game does not give it the ability to tell which is the case. We then have a distribution for each such triple.

We turn to defining the utility functions. At first glance it may seem that one should define privacy by having each party *gain* whenever it learns something meaningful on the other party's private input. Nevertheless, it seems that it is better to make a party *lose* if the other party learns anything about its secret information. Intuitively, the reason is that it must be worthwhile for the party who holds the data to maintain it a secret. In other words, having the other party gain any profit when breaking secrecy is irrelevant, since it does not introduce any incentive for the former party to prevent this leakage. (Note however that here the utility of a party depends on events that are not visible to it during the execution.) Formally, for each n we consider a sequence of input distributions for generating inputs of this length. Each distribution is denoted by D and indexed by triples (a_0, a_1, b), and is defined by picking $x \leftarrow (a_0, a_1)$ and returning (x, b). More concretely,

Definition 3.1 (Distribution ensembles for privacy) *The* distribution ensemble for privacy for *P_0 for a two-party function f is the ensemble $\mathcal{D}^{\mathsf{p}}_f = \{D^{\mathsf{p}}_{f,n}\}_{n\in\mathbf{N}}$ where*
$D^{\mathsf{p}}_{f,n} = \{D_{a_0,a_1,b}\}_{a_0,a_1,b\in\{0,1\}^n, f(a_0,b)=f(a_1,b)}$, and $D_{a_0,a_1,b}$ outputs (x, b), where $x \overset{R}{\leftarrow} (a_0, a_1)$.

Distribution ensembles for privacy for P_1 are defined analogously.

Let π be a two-party protocol computing a function f. Then, for every (n, a_0, a_1, b) as above and for every PPT algorithm $\mathcal{B}$, let the *augmented protocol for privacy* for π, with *guess algorithm* $\mathcal{B}$, be the protocol that first runs π and then runs $\mathcal{B}$ on the local state of π and two additional auxiliary values. We assume that $\mathcal{B}$ outputs a binary value, denoted by $\mathsf{guess}_{\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}},i}$ where P_i is the identity of the attacker. This value is interpreted as a guess for which one of the two auxiliary values is the input value of the other party.

Definition 3.2 (Utility function for privacy) *Let π be a two-party protocol and f be a two party function. Then, for every a_0, a_1, b such that $f(a_0, b) = f(a_1, b)$, and for every guessing algorithm $\mathcal{B}$, the* utility function for privacy *for party P_0, on input $x \in \{a_0, a_1\}$, is defined by:*

$$u^{\mathsf{p}}_0(\mathbf{History}_{\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}},1}(x, b, n), a_0, a_1, b) \mapsto \begin{cases} -1 & \text{if } \mathsf{guess}_{\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}},1} = g \text{ and } x = a_g \\ 0 & \text{otherwise} \end{cases}$$

The utility function for party P_1 is defined analogously. Note that if the history of the execution is empty, i.e., no message has been exchanged between the parties, and the inputs of the parties are taken from a distribution ensemble for privacy, then u^{p}_0 equals at least $-1/2$. This is due to the fact that P_1 can only guess x with probability at most $1/2$. Therefore, intuitively, it will be rational for P_0 to participate in the protocol (rather than to abort at the beginning) only if (and only if) the other party cannot guess the input of P_0 with probability significantly greater than $1/2$. The definition of Game Theoretic privately is as follows:

Definition 3.3 (Game Theoretic private protocols) *Let f and π be as above. Then, we say that π is* Game Theoretic private for party *P_0 if $\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}}$ is a Nash protocol with respect to $u^{\mathsf{p}}_0, u^{\mathsf{p}}_1$ and $\mathcal{D}^{\mathsf{p}}_f$ and all valid* PPT *$\mathcal{B}$.*

Game Theoretic private protocol for P_1 is defined analogously. A protocol is Game Theoretic private if it is Game Theoretic private both for P_0 and for P_1.

Theorem 3.4 *Let f be a deterministic two-party function, and let π be a two-party protocol that computes f correctly (cf. Definition 2.2). Then, π is Game Theoretic private if and only if π privately computes f in the presence of fail-stop adversaries.*

Proof: We begin with the proof that a Game Theoretic private protocol implies privacy by indistinguishability. Assume by contradiction that π does not compute f privately (in the sense of Definition 3.1) with respect to party P_0 (w.l.o.g.). This implies that there exists a PPT adversary

$\mathcal{A}$ that corrupts party P_0, a PPT distinguisher D, a non-negligible function ϵ and infinitely many tuples (x_0, x_1^0, x_1^1, n) where $|x_0| = |x_1^0| = |x_1^1| = n$ and $f(x_0, x_1^0) = f(x_0, x_1^1)$ such that,

$$\Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^0, n)) = 1] - \Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^1, n)) = 1] \geq \epsilon(n).$$

We assume, without loss of generality, that $\mathcal{A}$ never aborts prematurely, or in fact, it plays honestly. This is because we can construct an equivalent distinguisher that ignores all the messages sent after the round in which $\mathcal{A}$ would have originally aborted, and then applies D on the remaining view.

We prove that there exist a valid PPT $\mathcal{B}$ in which $(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}})$ is not a computational Nash Equilibrium in the game $\Gamma_{\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}},u^{\mathsf{p}}}$. That is, there exists an alternative strategy σ_1' for P_1, a non-negligible function $\epsilon'(\cdot)$ and infinitely many distributions $D_{x_0,x_1^0,x_1^1} \in \mathcal{D}_f^{\mathsf{p}}$ such that

$$u_1(\sigma_0^{\mathtt{continue}}, \sigma_1') > u_1(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}}) + \epsilon'(n)$$

contradicting the assumption that $(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}})$ is Nash Equilibrium in the game $\Gamma_{\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}},u^{\mathsf{p}}}$.

Let $\sigma_1^{\mathtt{abort}}$ be the strategy where P_1 aborts the protocol before it starts (the initial abort strategy). Then, for any $D_{x_0,x_1^0,x_1^1} \in D_f^{\mathsf{p}}$, and any PPT valid algorithm $\mathcal{B}$ it holds that:

$$u_1(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{abort}}) = -1/2$$

Consider the following $\mathcal{B}$ algorithm: On input $\mathbf{History}_{\pi,1}(x, c, n), x_0, x_1^0, x_1^1$, $\mathcal{B}$ invokes the distinguisher D and outputs x_1^0 if D outputs 1, and outputs x_1^1 otherwise.

We now consider the expected utility of P_1 in the game $\Gamma_{\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}},u^{\mathsf{p}}}$, when both parties run according to the prescribed strategy:

$$\begin{aligned}
&u_1(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}})\\
&\quad= \; -1 \cdot \Pr[\mathsf{guess}_{\pi^{p}_{\mathsf{Aug},\mathcal{B}},0} = x_1^b]\\
&\quad= \; -\Big(\Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^b, n)) = 1 \wedge b = 0] - \Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^b, n)) = 0 \wedge b = 1]\Big)\\
&\quad= \; -1/2 \cdot \Big(\Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^b, n)) = 1|\ b = 0] + (1 - \Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^b, n)) = 1|\ b = 1])\Big)\\
&\quad\leq \; -1/2 - 1/2 \cdot \epsilon(n)
\end{aligned}$$

Then it holds that,

$$u_1(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{abort}}) - u_1(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}}) \geq -1/2 + 1/2 + 1/2 \cdot \epsilon(n) \geq 1/2 \cdot \epsilon(n)$$

contradicting the assumption that $(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}})$ is in Nash Equilibrium. The above implies that π is not a Nash protocol.

We now turn to the proof in which privacy implies Nash with respect to $u_0^{\mathsf{p}}, u_1^{\mathsf{p}}$ and $\mathcal{D}_f^{\mathsf{p}}$ and all valid $\mathcal{B}$. Assume by contradiction that there exists a PPT $\mathcal{B}$, such that the augmented protocol $\pi^{\mathsf{p}}_{\mathsf{Aug},\mathcal{B}}$ is not a Nash protocol with respect to these parameters and party P_1 (w.l.o.g.). This means that there exist an alternative strategy σ_1', infinitely many distributions $D_{x_0,x_1^0,x_1^1,n} \in \mathcal{D}_f^{\mathsf{p}}$ and a non-negligible function ϵ such that

$$u_1(\sigma_0^{\mathtt{continue}}, \sigma_1') \geq u_1(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}}) + \epsilon(n)$$

without loss of generality, we can assume that σ_1' is the initial abort strategy; this is because in any other strategy, some information regarding P_1's input may be leaked (since it participates in the protocol), and lowers the utility. Moreover, for any valid PPT algorithm $\mathcal{B}$ it holds that

$$u_1(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{abort}}) = -1/2$$

From the contradiction assumption, it holds that

$$\begin{aligned} u_1(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{abort}}) &\geq u_1(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) + \epsilon(n) \\ -1/2 &\geq -\Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^b] + \epsilon(n) \\ \Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^b] &\geq 1/2 + \epsilon(n) \end{aligned}$$

Next we show that this implies that there exists a PPT adversary $\mathcal{A}$, a PPT distinguisher D, a non-negligible function ϵ' and infinitely many tuples of equal length inputs (x_0, x_1^0, x_1^1) with $f(x_0, x_1^0) = f(x_0, x_1^1)$ such that,

$$\Big| \Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^0, n)) = 1] - \Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^1, n)) = 1] \Big| \geq \epsilon'(n)$$

Fix (x_0, x_1^0, x_1^1, n) and consider an adversary $\mathcal{A}$ that simply runs as an honest party. Then, define a distinguisher D that invokes the algorithm $\mathcal{B}$ and outputs 1 if and only if $\mathcal{B}$ outputs x_1^0. Then formally,

$$\begin{aligned} &\Big| \Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^0, n)) = 1] - \Pr[D(\mathbf{View}_{\pi,\mathcal{A}(z),0}(x_0, x_1^1, n)) = 1] \Big| \\ &= \Big| \Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^0 \mid b = 0] - \Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^0 \mid b = 1] \Big| \\ &= \Big| \Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^0 \wedge b = 0] / \Pr[b = 0] - \Big(1 - \Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^1 \wedge b = 1] / \Pr[b = 1]\Big) \Big| \\ &= \Big| 2\Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^0 \wedge b = 0] + 2\Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^1 \wedge b = 1] - 1 \Big| \\ &= \Big| 2\Pr[\mathsf{guess}_{\pi^p_{\mathsf{Aug},\mathcal{B}},0} = x_1^b] - 1 \Big| \geq 2 \cdot (1/2 + \epsilon(n)) - 1 = 2\epsilon(n) \end{aligned}$$

which is a non-negligible probability. This concludes the proof. ■

3.2 Correctness in Game Theoretic view

We continue with a formulation of a utility function that captures the notion of correctness as formalized in Definition 3.5. That is, we show that a protocol correctly computes a deterministic function if and only if the strategy that never aborts the protocol is in a computational Nash equilibrium with respect to the set of utilities specified as follows. The parties get high payoff only if they output the correct function value on the given inputs (types), or abort before the protocol starts; in addition, the players get no payoff for incorrect output. More formally, we introduce the set of distributions for which we will prove the Nash theorem. The distribution ensemble for correctness is simply the collection of all point distributions on pairs of inputs:

Definition 3.5 (Distribution ensemble for correctness) *Let f be a deterministic two-party function. Then, the* distribution ensemble for correctness *is the ensemble $\mathcal{D}^{\mathsf{c}}_f = \{D^{\mathsf{c}}_n\}_{n\in\mathbf{N}}$ where $D^{\mathsf{c}}_n = \{D_{a,b}\}_{a,b\in\{0,1\}^n}$, and $D_{a,b}$ outputs (a, b) w.p. 1.*

Note that a fail-stop adversary cannot affect the correctness of the protocol as it plays honestly with the exception that it may abort. Then, upon receiving an abort message we have the following: (i) either the honest party already learnt its output and so, correctness should be guaranteed, or, (ii) the honest party did not learn the output yet, for which it outputs $\perp$ together with its guess for the output (which corresponds to a legal output by Definition 2.2). Note that this guess is different than the guess appended in Definition 3.2 of utility definition for privacy, as here, we assume that the protocol instructs the honest party how to behave in case of an abort. Furthermore, an incorrect protocol in the presence of fail-stop adversary implies that the protocol is incorrect regardless of the parties' actions (where the actions are continue or abort).

This suggests the following natural way of modeling a utility function for correctness: The parties gain a higher utility if they output the correct output, and lose if they output an incorrect output. Therefore, the `continue` strategy would not induce a Nash Equilibrium in case of an incorrect protocol, as the parties gain a higher utility by not participating in the execution. More formally:

Definition 3.6 (Utility function for correctness) *Let π be a two-party fail-stop game as above. Then, for every a, b as above the* utility function for correctness *for party P_0, denoted u_0^{c}, is defined by:*

- $u_0^{\mathsf{c}}(\mathbf{History}_{\pi,0}^{\phi}) = 1$.
- $u_0^{\mathsf{c}}(\mathbf{Output}_{\pi,0}, a, b) \mapsto \begin{cases} 1 & \text{if } \mathbf{Output}_{\pi,0} = f(a,b) \\ 0 & \text{otherwise} \end{cases}$

where $\mathbf{History}_{\pi,0}^{\phi}$ denotes the case that the local history of P_0 is empty. (Namely, P_0 does not participate in the protocol).

Intuitively, this implies that the protocol is a fail-stop game if it is correct and vice versa. A formal statement follows below. u_1^{c} is defined analogously, with respect to P_1.

Theorem 3.7 *Let f be a deterministic two-party function, and let π a two-party protocol. Then, π is a Nash protocol with respect to $u_0^{\mathsf{c}}, u_1^{\mathsf{c}}$ and $\mathcal{D}_f^{\mathsf{c}}$ if and only if π correctly computes f in the presence of fail-stop adversaries.*

Proof: We begin with the proof that a Nash protocol with respect to $u_0^{\mathsf{c}}, u_1^{\mathsf{c}}$ and $\mathcal{D}_f^{\mathsf{c}}$ implies correctness. Assume by contradiction that π does not compute f correctly. Meaning, there exist infinitely many pairs of inputs (x_0, x_1), $i \in \{0,1\}$ and a non-negligible function ϵ such that

$$\Pr[\mathbf{Output}_{\pi,i} \notin \{\perp \circ \{0,1\}^*, f(x_0,x_1)\}] \geq \epsilon(n)$$

We assume that the output of the protocol in case of prematurely abort starts with $\perp$. Thus, our contradiction assumption holds only for the case where the protocol terminates successfully, however, P_i outputs a value different than $f(x_0,x_1)$. Without loss of generality, assume that $i = 0$. We now show that $(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}})$ does not induce a computational Nash Equilibrium in the game $\Gamma_{\pi,u^{\mathsf{c}}}$. Namely, the expected utility for P_0 when both parties play according to the prescribed strategy is,

$$\begin{aligned} u_0(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}}) &= \Pr[\mathbf{Output}_{\pi,0} \notin \{f(x_0,x_1), \perp \circ \{0,1\}^*\}] \\ &= 1 - \Pr[\mathbf{Output}_{\pi,0} \in \{f(x_0,x_1), \perp \circ \{0,1\}^*\}] \leq 1 - \epsilon(n) \end{aligned}$$

Let $\sigma_0^{\mathsf{abort}}$ be the strategy for which P_0 initially aborts (i.e., does not participate in the protocol). Then, we have that

$$u_0(\sigma_0^{\mathtt{abort}}, \sigma_1^{\mathtt{continue}}) = 1$$

Thus, it holds that

$$u_0(\sigma_0^{\texttt{abort}}, \sigma_1^{\texttt{continue}}) \geq u_0(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) + \epsilon(n)$$

contradicting the assumption that $(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}})$ induces a computational Nash Equilibrium in game $\Gamma_{\pi,u^{\mathsf{c}}}$.

We now turn to the proof in which correctness implies Nash with respect to $u_0^{\mathsf{c}}, u_1^{\mathsf{c}}$ and $\mathcal{D}_f^{\mathsf{c}}$. Assume by contradiction that π is not a Nash protocol with respect to u_0^{c} (w.l.o.g.) and $\mathcal{D}_f^{\mathsf{c}}$. This means that there exists an alternative strategy for P_0 (w.l.o.g), infinitely many distributions $D_{x_0,x_1,n} \in \mathcal{D}_f^{\mathsf{c}}$ and a non-negligible function ϵ in which

$$u_0(\sigma_0', \sigma_1^{\texttt{continue}}) \geq u_0(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) + \epsilon(n)$$

When both parties follow the prescribed strategy, the format of the output does not start with $\perp$, and thus we have

$$u_0(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) = \Pr[\mathbf{Output}_{\pi,i} \in \{\perp \circ \{0,1\}^*, f(x_0,x_1)\}].$$

Under the assumption that P_1 plays according to $\sigma_1^{\texttt{continue}}$, once P_0 starts the protocol - its utility can only reduces unless it outputs $f(x_0,x_1)$ exactly. This can happen only when the protocol terminates successfully, i.e., it plays according to $\sigma_0^{\texttt{continue}}$. Therefore, the only alternative strategy that yields a higher utility is $\sigma_0^{\texttt{abort}}$ (i.e, the initial abort). The expected utility for P_0 when it plays according to $\sigma_0^{\texttt{abort}}$ and σ_1 plays according to $\sigma_1^{\texttt{continue}}$ is 1. Thus, we have that

$$\begin{array}{rcl}
u_0(\sigma_0^{\texttt{abort}}, \sigma_1^{\texttt{continue}}) & \geq & u_0(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) + \epsilon(n) \\
1 & \geq & \Pr[\mathbf{Output}_{\pi,i} \in \{\perp \circ \{0,1\}^*, f(x_0,x_1)\}] + \epsilon(n) \\
\Pr[\mathbf{Output}_{\pi,i} \notin \{\perp \circ \{0,1\}^*, f(x_0,x_1)\}] & \geq & \epsilon(n)
\end{array}$$

Yielding that π does not compute f correctly on these inputs. ■

4 Exploring Fairness in the Two-Party Setting

Having established the notions of privacy and correctness using Game Theoretic formalism, our next goal is to capture fairness in this view. However, this turns out to be tricky, mainly due to the highly "reciprocal" and thus delicate nature of this notion. To illustrate, consider the simplistic definition for fairness that requires that one party learns its output if and only if the second party does. However, as natural as it seems, this definition is lacking since it captures each party's output as an atomic unit. As a result, it only considers the cases where the parties either learnt or did not learn their output *entirely*, and disregards the option in which partial information about the output may be gathered through the execution. So, instead, we would like to have a definition that calls a protocol fair if at any point in the execution both parties gather, essentially, the same partial information about their respective outputs.

Motivated by this discussion, we turn to the Game Theoretic setting with the aim to design a meaningful definition for fairness, as we did for privacy and correctness. This would, for instance, allow investigating known impossibility results under a new light. Our starting point is a definition that examines the information the parties gain about their outputs during the game, where each party loses nominatively to the success probability of the other party guessing its output. (This is motivated by the same reasoning as in privacy). In order to obtain this, we first define a new set of utility functions for fairness for which we require that the game would be Nash; see Section 4.1 for the complete details.

Having defined fairness for rational parties, we wish to examine its strength against cryptographic attacks. We therefore introduce a new game-based definition formalizes fairness for two-party protocols and is, in fact, equivalent to the Game Theoretic definition. We then provide a new definition of the gradual release property, that is *equivalent* to the Game Theoretic and game-based definitions. Finally, we present a simulation-based definition and explore the realizability of our notions. Specifically, we include a new definition of the gradual release property (cf. Section 4.3) and demonstrate a correlation between protocols with this property and protocols that are fair according to our notion of fairness. In particular, it shows that at any given round, the parties cannot improve their chances for guessing correctly "too much". Otherwise, the protocol would not be fair. We then introduce in Section 4.4 a new notion of simulation-based definition for capturing security of protocols that follow our game-based notion of fairness, specified above. This new notion is necessary as (gamed-based) fair protocols most likely cannot be simulatable according to the traditional simulation-based definition [13]. We consider the notion of "partial information" in the ideal world alongside preserving some notion of privacy. We then prove that protocols that satisfy this new definition are also fair with respect to game-based definition.

Finally, we consider the realizability of our notion of fairness. We then observe that our notion is meaningful even in the case where parties are not guaranteed to always learn the output when both parties never abort. Somewhat surprisingly, in cases where the parties learn the output with probability one half or smaller, our notion of fairness is in fact *achievable* with no set-up or trusted third parties. We demonstrate two-party protocols that realize the new notion in this settings. We also show that whenever this probability raises above one half, our notion of fairness cannot be realized at all.

4.1 Fairness in Game Theoretic View

In this section we present our first definition for fairness that captures this notion from a Game Theoretic view. As for privacy and correctness, this involves definitions for utility functions, input distributions and a concrete fail-stop game (or the sequence of games). We begin with the description of the input distributions. As specified above, the input of each party is picked from a domain of size two, where all the outputs are made up of distinct outputs. More formally,

Definition 4.1 (Collection of distribution ensembles for fairness) *Let f be a two-party function. Let $(x_0^0, x_0^1, x_1^0, x_1^1, n)$ be an input tuple such that: $|x_0^0| = |x_0^1| = |x_1^0| = |x_1^1| = n$, and for every $b \in \{0,1\}$ it holds that:*

- $f_0(x_0^0, x_1^b) \neq f_0(x_0^1, x_1^b)$ *(in each run there are two possible outputs for P_0).*
- $f_1(x_0^b, x_1^0) \neq f_1(x_0^b, x_1^1)$, *(in each run there are two possible outputs for P_1).*

Then, a collection of distribution ensembles for fairness $\mathcal{D}_f^{\mathsf{f}}$ *is a collection of distributions $\mathcal{D}_f^{\mathsf{f}} = \{D_{x_0^0,x_0^1,x_1^0,x_1^1,n}\}_{x_0^0,x_0^1,x_1^0,x_1^1,n}$ such that for every $(x_0^0, x_0^1, x_1^0, x_1^1, n)$ as above, $D_{x_0^0,x_0^1,x_1^0,x_1^1,n}$ is defined by*

$$(x_0, x_1) \leftarrow D_{x_0^0,x_0^1,x_1^0,x_1^1,n}(1^n), \quad \text{where} \quad x_0 \overset{R}{\leftarrow} (x_0^0, x_0^1) \quad \text{and} \quad x_1 \overset{R}{\leftarrow} (x_1^0, x_1^1).$$

Next, let $\pi_{\mathcal{B}}$ be the protocol, where $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1)$. By this notation, we artificially separate between the protocol and the predicting algorithms in case of prematurely abort. More precisely, in the case that P_0 prematurely aborts, P_1 invokes algorithm $\mathcal{B}_1$ on its input, its auxiliary information and the history of the execution, and outputs whatever $\mathcal{B}_1$ does. $\mathcal{B}_0$ is defined in a similar manner. In fact, we can refer to these two algorithms by the instructions of the parties regarding the values they need to output after each round, capturing the event of an early abort. We stress these algorithms are embedded within the protocol. However, this presentation enables us to capture scenarios where one of the parties follow the guessing algorithm as specified by the protocol, whereas the other party follows an arbitrary algorithm. That is, we can consider protocols $\pi_{\mathcal{B}'}$ (with $\mathcal{B}' = (\widetilde{\mathcal{B}}_0, \mathcal{B}_1)$) that are equivalent to the original protocol $\pi_{\mathcal{B}}$ except for the fact that P_0 guesses its output according to $\widetilde{\mathcal{B}}_0$ instead of $\mathcal{B}_0$. We now describe the fairness game $\Gamma_{\pi_{\mathcal{B}},u^{\mathsf{f}}}$ for some $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1)$. The inputs of the parties, x_0, x_1, are selected according to some distribution ensemble $D_{x_0^0,x_0^1,x_1^0,x_1^1}$ as defined in Definition 4.1. Then, the parties run the fail-stop game, where their strategies instruct them in each step whether to `abort` or `continue`. In case that a party P_i aborts, the outputs of both parties are determined by the algorithms $(\mathcal{B}_0, \mathcal{B}_1)$. Let $\mathbf{Output}_{\pi_{\mathcal{B}'},i}$ denote the output of P_i in game $\pi_{\mathcal{B}'}$, then a utility function for fairness is defined by:

Definition 4.2 (Utility function for fairness) *Let f be a deterministic two-party function, and let π be a two-party protocol. Then, for every $x_0^0, x_0^1, x_1^0, x_1^1, n$ as above (cf. Definition 4.1), for every pair of strategies (σ_0, σ_1) and for every* PPT $\widetilde{\mathcal{B}}_0$*, the* utility function for fairness *for party P_0, denoted by u_0^{f}, is defined by:*

$$u_0^{\mathsf{f}}(\sigma_0,\sigma_1) \mapsto \begin{cases} 1 & \text{if } \mathbf{Output}_{\pi^{\mathsf{f}}_{\mathcal{B}'},0} = f_0(x_0,x_1) \wedge \ \mathbf{Output}_{\pi^{\mathsf{f}}_{\mathcal{B}'},1} \neq f_1(x_0,x_1) \\ -1 & \text{if } \mathbf{Output}_{\pi^{\mathsf{f}}_{\mathcal{B}'},0} \neq f_0(x_0,x_1) \wedge \ \mathbf{Output}_{\pi^{\mathsf{f}}_{\mathcal{B}'},1} = f_1(x_0,x_1) \\ 0 & \text{otherwise} \end{cases}$$

where x_0, x_1 are as in Definition 4.1 and $\mathcal{B}' = (\widetilde{\mathcal{B}}_0, \mathcal{B}_1)$. Moreover, the utility for P_1, $u_1^{\mathsf{f}} = 0$.

Since the utility function of P_1 is fixed, P_1 has no incentive to change its strategy. Moreover, we consider here the sequence of games where P_1 always guesses its output according to $\mathcal{B}_1$, the "original" protocol. This actually means that P_1 always plays honestly, from the cryptographic point of view. We are now ready to define a protocol that is Game Theoretic fair for P_1 as:

Definition 4.3 (Game Theoretic fairness for P_1) *Let f and $\pi_{\mathcal{B}}$ be as above. Then, we say that $\pi_{\mathcal{B}}$ is* Game Theoretic fair for party P_1 *if $\Gamma_{\pi_{\mathcal{B}'},(u_0^{\mathsf{f}},u_1^{\mathsf{f}})}$ is a Nash protocol with respect to $(u_0^{\mathsf{f}}, u_1^{\mathsf{f}})$ and $\mathcal{D}_f^{\mathsf{f}}$ and all* PPT $\widetilde{\mathcal{B}}_0$*, where $\mathcal{B}' = (\widetilde{\mathcal{B}}_0, \mathcal{B}_1)$.*[2]

Namely, if $\pi_{\mathcal{B}}$ is not Game Theoretic fair for P_1, then P_0 (and only P_0) can come up with a better strategy, and some other guessing algorithm $\widetilde{\mathcal{B}}_0$ (where both define an adversary in the cryptographic world). This is due to the fact that the utility for P_1 is fixed, and so it cannot find an alternative strategy that yields a higher utility. In other words, P_1 has no reason to invoke a different guess algorithm, and so this definition formalizes the case where P_1 is honest in the protocol $\pi_{\mathcal{B}}$. This, in particular, implies that a somewhat natural zero-sum game, where the total balance of utilities is zero, does not work here.

Similarly, we define Game Theoretic fair for party P_0, where here we consider all the protocols $\pi_{\mathcal{B}'}$, for all PPT $\widetilde{\mathcal{B}}_1$ and $\mathcal{B}' = (\mathcal{B}_0, \widetilde{\mathcal{B}}_1)$, and the utilities functions are opposite (that is, the utility for P_0 is fixed into zero, whereas the utility of P_1 is modified according to its guess). We conclude with the definition for Game Theoretic protocol:

[2] At first glance it might look as if the utilities from Definition 4.2 do not incentivize the parties to participate in the interaction to begin with. However this concern can be dealt with by giving each party some fixed high payoff in case it participates.

Definition 4.4 (Game Theoretic fair protocol) *Let f and π be as above. Them. we say that π is* Game Theoretic fair protocol *if it is Game Theoretic fair for both P_0 and P_1.*

4.2 A New Indistinguishability-Based Definition of Fairness

Towards introducing our cryptographic notion for fairness, we first consider a basic, one-sided definition that guarantees fairness only for one of the two parties. A protocol is considered fair if the one sided definition is satisfied with respect to both parties. We refer to this game as a test for the protocol in a "fair" environment, where each party has two possible inputs and its effective input is chosen uniformly at random from this set. Moreover, both parties know the input tuple and the distribution over the inputs. This is due to the fact that we, untraditionally, assume that the protocol instructs the honest party to guess the output of the function based on its view and the information it recorded so far. We note that these scenarios are not fair under the traditional simulation-based definition [13], since they do not follow the *all or nothing* method, in which the parties can only learn the entire output at once. In order to illustrate this, consider a protocol that enables both parties to learn the correct output with probability about $3/4$. Then, this protocol is fair according to our definition below although it is not simulatable by the [21] definition. Before introducing the game-based definition, we first introduce non-trivial functionalities, to avoid functionalities that one of the parties may know the correct output without participating.

Definition 4.5 (Non-trivial functionalities.) *Let f be a two-party function. Then, f is non trivial if for all sufficiently large n's, there exists an input tuple $(x_0^0, x_0^1, x_1^0, x_1^1, n)$ such that $|x_0^0| = |x_0^1| = |x_1^0| = |x_1^1| = n$ and $\{f_0(x_0, x_1^b), f_0(x_0, x_1^b)\}_{b\in\{0,1\}}$, $\{f_1(x_0^b, x_1^0), f_1(x_0^b, x_1^1)\}_{b\in\{0,1\}}$ are distinct values.*

We are now ready to introduce our formal definition for fairness:

Definition 4.6 (Game-based definition for fairness.) *Let f be a non-trivial two-party function, and let π be a two-party protocol. Then, for every input tuple (cf. Definition 4.5) and any* PPT *fail-stop adversary $\mathcal{A}$, we define the following game:*

Game $\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n)$:

1. Two bits b_0, b_1 are picked at random.
2. Protocol π is run on inputs $x_0^{b_0}$ for P_0 and $x_1^{b_1}$ for P_1, where $\mathcal{A}$ sees the view of P_{i^*}.
3. Whenever $\mathcal{A}$ outputs a value y, P_{1-i^*} is given an `abort` message. (At this point, P_{1-i^*} would write its guess for $f_{1-i^*}(x_0^{b_0}, x_1^{b_1}, n)$ on its output tape.)
4. The output of the game is:
   - 1 if (i) $y = f_0(x_0^{b_0}, x_1^{b_1}, n)$ and (ii) P_{1-i^*} does not output $f_1(x_0^{b_0}, x_1^{b_1}, n)$.
   - -1 if (i) $y \neq f_0(x_0^{b_0}, x_1^{b_1}, n)$ and (ii) P_{1-i^*} outputs $f_1(x_0^{b_0}, x_1^{b_1}, n)$.
   - 0 otherwise (i.e., either both parties output correct outputs or both output incorrect outputs).

We say that π fairly computes *f if for every* PPT *adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ such that for all sufficiently large inputs it holds that,*

$$\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n)) \leq \mu(n)$$

At first sight it may seem that Definition 4.6 is tailored for the fair exchange function, i.e., when the parties trade their inputs. This is due to the fact that the parties' output completely reveal their inputs. Nevertheless, we note that the definition does not put any restriction on f in this sense and is aimed to capture fairness with respect any nontrivial function. We continue with the following theorem:

Theorem 4.7 *Let f be a two-party function and let π be a protocol that computes f correctly. Then, π is Game Theoretic fair (in the sense of Definition 4.4), if and only if π fairly computes f in the presence of fail-stop adversaries, (in the sense of Definition 4.6).*

Proof: We first write explicitly the guessing algorithm $\mathcal{B} = (\mathcal{B}_0, \mathcal{B}_1)$ and denote the protocol π as $\pi_{\mathcal{B}}$. Assume that $\pi_{\mathcal{B}}$ is Game Theoretic fair for P_0 and P_1; we thus prove that $\pi_{\mathcal{B}}$ is fair in the sense of Definition 4.6. Assume by contradiction that $\pi_{\mathcal{B}}$ is not fair with respect to this latter definition. Thus, there exist infinitely many input tuples $x_0^0, x_0^1, x_1^0, x_1^1, n$ which yield distinct outputs, a PPT adversary $\mathcal{A}$ controlling P_0 (w.l.o.g.) and a non-negligible function ϵ for which

$$\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A},0}(x_0^0, x_0^1, x_1^0, x_1^1, n)) \geq \epsilon(n)$$

This implies that,

$$\left| \Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) = 1\right] - \Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) = -1\right] \right| \geq \epsilon(n)$$

Namely, when P_1 runs according to the protocol specifications and P_0 runs according to $\mathcal{A}'s$ strategy, the expected advantage of P_0 over P_1 in guessing successfully is non-negligibly higher.

We now show that there exists $\mathcal{B}' = (\mathcal{B}_0^{\mathcal{A}}, \mathcal{B}_1)$ for which the game $\Gamma_{\pi_{\mathcal{B}'}, u^{\mathsf{f}}}$ is not a Nash game. That is, consider an alternative strategy $\sigma_0^{\mathcal{A}}$ for P_0: This strategy invokes the adversary $\mathcal{A}$, such that for each message that the strategy receives from the other party, it forwards this message to $\mathcal{A}$. Furthermore, whenever it receives the message and guess for the following round from the game (together with the random coins that were used to generate these values), the strategy invokes the adversary $\mathcal{A}$ with the appropriate random coins. If the adversary chooses to output this message, $\sigma_0^{\mathcal{A}}$ outputs the action `continue`. If the adversary $\mathcal{A}$ chooses to abort, $\sigma_0^{\mathcal{A}}$ outputs `abort`.

Moreover, let $\mathcal{B}_0^{\mathcal{A}}$ be the following guess algorithm: On input $\left(\mathbf{History}_{\pi,0}(x_0^{b_0}, x_1^{b_1}, n), x_0^0, x_0^1, x_1^0, x_1^1\right)$, $\mathcal{B}_0^{\mathcal{A}}$ invokes the adversary $\mathcal{A}$ on the above history. Upon completion, $\mathcal{B}_0^{\mathcal{A}}$ outputs whatever $\mathcal{A}$ does. Let $\vec{x} = (x_0^{b_0}, x_1^{b_1})$. Then, in a run of $(\sigma_0^{\mathcal{A}}, \sigma_1^{\mathtt{continue}})$ within game $\Gamma_{\pi_{\mathcal{B}'}, u^{\mathsf{f}}}$ where $\mathcal{B}' = (\mathcal{B}_0^{\mathcal{A}}, \mathcal{B}_1)$, we have that:

$$\Pr[\mathbf{Output}_{\pi_{\mathcal{B}'},0} = f_0(\vec{x}) \wedge \mathbf{Output}_{\pi_{\mathcal{B}'},1} \neq f_1(\vec{x})] - \Pr[\mathbf{Output}_{\pi_{\mathcal{B}'},0} \neq f_1(\vec{x}) \wedge \mathbf{Output}_{\pi_{\mathcal{B}'},1} = f_1(\vec{x})] \geq \epsilon(n)$$

yielding that the expected utility of P_0 when it follows $\sigma_0^{\mathcal{A}}$ (and assuming that P_1 runs according to the prescribed strategy), is non-negligible. That is,

$$u_0(\sigma_0^{\mathcal{A}}, \sigma_1^{\mathtt{continue}}) \geq \epsilon(n)$$

When both parties play according to the prescribed strategy $\sigma_i^{\mathtt{continue}}$, both parties learn the correct output at the end of the protocol except to some negligible function $\mu(\cdot)$, and thus:

$$u_0(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}}) \leq \mu(n)$$

which is implied by the correctness of the protocol, and thus:

$$u_0(\sigma_0^{\mathcal{A}}, \sigma_1^{\mathtt{continue}}) - u_0(\sigma_0^{\mathtt{continue}}, \sigma_1^{\mathtt{continue}}) \geq \epsilon(n) - \mu(n) \geq \epsilon(n)/2$$

for infinitely many n's, implying that there exists a non-negligible difference between the alternative strategy and the prescribed strategy, and thus, $\pi_{\mathcal{B}}$ is not Game Theoretic fair for P_0.

Next, assume that $\pi_{\mathcal{B}}$ meets Definition 4.6 and prove that $\pi_{\mathcal{B}}$ is Game Theoretic fair for P_0 and P_1. Assume by contradiction that $\pi_{\mathcal{B}}$ is not Game Theoretic fair for P_1 (w.l.o.g.), implying that there exists an alternative strategy σ_0', infinitely many distributions $D_{x_0^0,x_0^1,x_1^0,x_1^1,n} \in \mathcal{D}_f^{\mathsf{f}}$, a PPT $\widetilde{\mathcal{B}}_0$ and a non-negligible function ϵ such that $\pi_{\mathcal{B}'}$ is not a Nash protocol, where $\mathcal{B}' = (\widetilde{\mathcal{B}}_0, \mathcal{B}_1)$. Namely,

$$u_0(\sigma_0', \sigma_1^{\texttt{continue}}) \geq u_0(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) + \epsilon(n)$$

Based on the same assumption as above, when both parties follow $\sigma^{\texttt{continue}}$ then both learn the correct output except to some negligible function $\mu(\cdot)$, and thus:

$$u_0(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) \geq -\mu(n)$$

implying that,

$$u_0(\sigma_0', \sigma_1^{\texttt{continue}}) \geq u_0(\sigma_0^{\texttt{continue}}, \sigma_1^{\texttt{continue}}) + \epsilon(n) \geq \epsilon(n) - \mu(n) \geq \epsilon(n)/2$$

Now, consider the following adversary $\mathcal{A}$ which controls the party P_0: $\mathcal{A}$ invokes the strategy σ_0' with the random coins of $\mathcal{A}$. Then, whenever σ_0' outputs `continue`, $\mathcal{A}$ computes the next message (using the same random coins) and outputs it. Whenever $\mathcal{A}$ receives a message from P_1, it passes the message to σ_0'. When σ_0' outputs `abort`, $\mathcal{A}$ aborts, invokes the guess algorithm $\widetilde{\mathcal{B}}_0$, and outputs the corresponding output to $\widetilde{\mathcal{B}}_0$'s guess. We have that an execution of the protocol $\pi_{\mathcal{B}}$ with an honest party P_1 and the adversary $\mathcal{A}$, is equivalent to an execution of the game $\Gamma_{\pi_{\mathcal{B}'},u^{\mathsf{f}}}$ with $\mathcal{B}' = (\widetilde{\mathcal{B}}_0, B_1)$ where the parties follow the $(\sigma_0', \sigma_1^{\texttt{continue}})$ strategies. Therefore, we have that

$$\begin{aligned}\Pr[\mathbf{Output}_{\pi_{\mathcal{B}},0} = f_0(\vec{x}) \wedge \mathbf{Output}_{\pi_{\mathcal{B}},1} \neq f_1(\vec{x})] - \Pr[\mathbf{Output}_{\pi_{\mathcal{B}},0} \neq f_1(\vec{x}) \wedge \mathbf{Output}_{\pi_{\mathcal{B}},1} = f_1(\vec{x})]\\ = u_0(\sigma_0', \sigma_1^{\texttt{continue}}) \geq \epsilon(n)/2\end{aligned}$$

and so,

$$\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n)) \geq \epsilon(n)/2$$

contradicting the assumption that $\pi_{\mathcal{B}}$ is fair. ■

4.3 The Limited Gradual Release Property

In this section we show that the guesses of the parties at any given round, and that every two consecutive guesses of each party, must be statistically close. Namely, we show that once there is a non-negligible difference between these, the protocol cannot be fair. Intuitively, this follows from the fact that any such difference is immediately translated into an attack where the corresponding party aborts before completing this round, while gaining an advantage over the other party in learning the output.

The significance of limited gradual release is in providing a new tool to study fairness in the two-party setting. Specifically, by proving that it implies fairness (in the sense of Definition 4.6; see Theorem 4.11), it enables to gain more insights regarding the characterization of fairness. On the other hand, the fact that limited gradual release is implied by fairness (cf. Theorem 4.10), may shed light on the reason it is impossible to achieve limited gradual release under certain constraints. Notably, our notion of limited gradual release is in contrast to the typical meaning of gradual release specified in prior work (e.g, [7]), which refers to *partially fair* protocols in which the parties learn their output gradually such that the amount of information that is released regarding the output is limited, but non-negligible, and there always exists a party with a non-negligible advantage over the others.

We show a tight correlation between Definition 4.6 for fairness and the limited gradual release property by showing that each notion is implied by the other. Without loss of generality, assume that P_0 sends the first message of the protocol and denote by a round a pair of messages sent from P_0 to P_1 and back from P_1 to P_0. We recall first that in Definition 4.6, we assume that the protocol instructs the parties how to guess the output in the case of an abort message. In a broader sense, the protocol can instruct each party to maintain a tape where it updates its guess after each round. This corresponds to having each party computing its guess, conditioned on the event that the other party aborts in the following round. More formally, denote by a_{i+1} the guess of party P_0 when P_1 aborts after sending its message in round i; likewise, b_{i+1} denotes the guess of P_1 when P_0 quits after sending its message at round i. We call these values the "default outputs"; see figure 2 for an illustration. Furthermore, if the execution is terminated in some round i (either due to an early abort or to a successful completion), the parties define by $a_j = a_i$ and $b_j = b_i$ for all $j > i$.

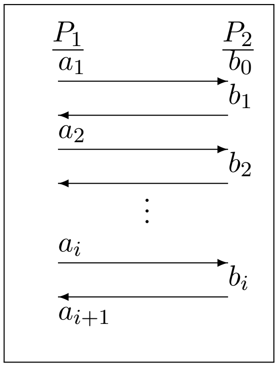

Figure 2: Default output notations (non-simultaneous)

We generalize the default output notations to deal with an arbitrary guess algorithm. Namely, a fail stop adversary $\mathcal{A}$ may prematurely abort and guess its output according to a guess algorithm, different than the one specified by the protocol. Formally, for every adversary $\mathcal{A}$ denote by $\mathsf{guess}_{\mathcal{A},i}$ the adversary's guess in round i. We note that an adversary may not write its guess in every round, however, without loss of generality, we can assume that it does so. Note that $\mathsf{guess}_{\mathcal{A},i}$ is a random variable that is well defined even if the protocol has been terminated before round i. Then, we define limited gradual release to deal with every adversary. Formally:

Definition 4.8 (m-limited gradual release) *Let f be a non-trivial two-party function, and let π be a protocol. We say that π maintains m-*limited gradual release*, if for every* PPT *fail-stop adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$, such that for every $i < m$ and every $x_0^0, x_0^1, x_1^0, x_1^1, n$ as above (cf. Definition 4.5) it holds that:*

1. In case $\mathcal{A}$ controls party P_0:

$$\Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1)\right] \leq \Pr\left[b_i = f_1(x_0, x_1)\right] + \mu(n)$$

2. In case $\mathcal{A}$ controls party P_1:

$$\Pr\left[\mathsf{guess}_{\mathcal{A},i} = f_1(x_0, x_1)\right] \leq \Pr\left[a_i = f_0(x_0, x_1)\right] + \mu(n),$$

where $x_0 \overset{R}{\leftarrow} \{x_0^0, x_0^1\}$ *and,* $x_1 \overset{R}{\leftarrow} \{x_1^0, x_1^1\}$.

Having defined m-limited gradual release, we say that a protocol maintains limited gradual release if it is poly-limited gradual release for some polynomial poly which indicates an upper bound its number of rounds. This captures the fact that the protocol must be terminated within a strict polynomial number of rounds. Formally,

Definition 4.9 (Gradual release) *Let f and π be as above. Then, we say that π maintains limited gradual release if it is m-limited gradual release for $m = \mathsf{poly}(n)$ for some polynomial poly that bounds the number of rounds of π.*

Fairness implies limited gradual release. We complete this section by establishing a correlation between the two notions of fairness and limited gradual release. In particular, we prove that each is implied by the other, beginning with the proof that fairness implies gradual release:

Theorem 4.10 *Let f and π be as above. Let poly be a polynomial that bounds the number of rounds of the protocol π and assume that π is fair (in the sense of Definition 4.6). Then, π maintains limited gradual release (cf. Definition 4.9).*

Proof: The proof follows by contradiction. Namely, assume that π does not maintain the limited gradual release property. Then, there exists an adversary $\mathcal{A}$ that controls (w.l.o.g) party P_0, a non-negligible function $\epsilon(\cdot)$, a round $\mathsf{poly}(n) \geq i > 0$ and infinitely many tuples $x_0^0, x_0^1, x_1^0, x_1^1, n$, such that,

$$\Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1)\right] \geq \Pr\left[b_i = f_1(x_0, x_1)\right] + \epsilon(n)$$

Then observe that,

$$\begin{aligned}\Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1)\right] &= \Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1) \wedge b_i = f_1(x_0, x_1)\right] \\ &+ \quad \Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1) \wedge b_i \neq f_1(x_0, x_1)\right]\end{aligned}$$

and that,

$$\begin{aligned}\Pr\left[b_i = f_1(x_0, x_1)\right] &= \Pr\left[b_i = f_1(x_0, x_1) \wedge \mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1)\right] \\ &+ \quad \Pr\left[b_i = f_1(x_0, x_1) \wedge \mathsf{guess}_{\mathcal{A},i+1} \neq f_0(x_0, x_1)\right].\end{aligned}$$

Combing these and based on the contradiction assumption we conclude that,

$$\Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1) \wedge b_i \neq f_1(x_0, x_1)\right] > \Pr\left[b_i = f_1(x_0, x_1) \wedge \mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1)\right] + \epsilon(n). \quad (1)$$

Next, we describe a PPT adversary $\mathcal{A}_i$ for which the expected output game $\mathsf{Fair}_{\pi,\mathcal{A}_i}(x_0^0, x_0^1, x_1^0, x_1^1, n)$ is non-negligibly greater than zero. Consider the adversary $\mathcal{A}_i$ that controls P_0 and does the following: it invokes the adversary $\mathcal{A}$, plays according to its strategy until round i, aborts in round $i+1$ (right before sending its message in this round), and outputs $\mathcal{A}$'s default output in this round. Note first, that $\mathcal{A}_i$ runs in polynomial time since $i < \mathsf{poly}(n)$, and the honest party's instructions can be followed in polynomial time as well. Moreover, it holds that,

$$\Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}_i}(x_0^0, x_0^1, x_1^0, x_1^1, n) = 1\right] = \Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1) \wedge b_i \neq f_1(x_0, x_1)\right] \quad (2)$$

where x_0, x_1 are chosen according to game $\mathsf{Fair}_{\pi,\mathcal{A}_i}(x_0^0, x_0^1, x_1^0, x_1^1, n)$, (which is exactly as they were chosen according to Definition 4.9). Then, Eq. (2) holds since the output of game $\mathsf{Fair}_{\pi,\mathcal{A}_i}$ is 1 only when $\mathcal{A}_i$ outputs the correct output $f_0(x_0, x_1)$, whereas P_1 outputs an incorrect output, i.e., different than $f_1(x_0, x_1)$. We stress that the probability above already embeds the event that the parties do not reach round $i+1$ (namely, when π is terminated before round $i+1$). This is due to the fact that in case π is terminated in round j, it holds that $\mathsf{guess}_{\mathcal{A},i+1} = \mathsf{guess}_{\mathcal{A},j}$, and $b_i = b_j$ for all $j < i$.

Similarly, we have that,

$$\Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}_i}(x_0^0, x_0^1, x_1^0, x_1^1, n) = -1\right] = \Pr\left[\mathsf{guess}_{\mathcal{A},i+1} \neq f_0(x_0, x_1) \wedge b_i = f_1(x_0, x_1)\right].$$

Finally, combining the above we get that,

$$\begin{aligned}
&\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}_i}(x_0^0, x_0^1, x_1^0, x_1^1, n)) \\
&\quad = \Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}_i}(x_0^0, x_0^1, x_1^0, x_1^1, n) = 1\right] - \Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}_i}(x_0^0, x_0^1, x_1^0, x_1^1, n) = -1\right] \\
&\quad = \Pr\left[\mathsf{guess}_{\mathcal{A},i+1} = f_0(x_0, x_1) \wedge b_i \neq f_1(x_0, x_1)\right] - \Pr\left[\mathsf{guess}_{\mathcal{A},i+1} \neq f_0(x_0, x_1) \wedge b_i = f_1(x_0, x_1)\right] \\
&\quad > \epsilon(n).
\end{aligned}$$

In contradiction to the assumption that π is fair with respect to Definition 4.6. ■

Gradual release implies fairness. Next, we prove that fairness is implied by limited gradual release:

Theorem 4.11 *Let f and π be as above and assume that π maintains the limited gradual release property (cf. Definition 4.9). Then, π is a fair protocol (in the sense of Definition 4.6).*

Proof: Let $\mathcal{A}$ be any adversary. We show that for there exists a negligible function $\mu(\cdot)$, such that for all sufficiently large inputs $(x_0^0, x_0^1, x_1^0, x_1^1, n)$ it holds that:

$$\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n)) \leq \mu(n)$$

Fix $(x_0^0, x_0^1, x_1^0, x_1^1, n)$ to be the input tuple and assume that $i^* = 0$ (i.e., party P_0 is corrupted). Moreover, denote by $\mathsf{abort}_i^{\mathcal{A}}$ the event that $\mathcal{A}$ aborts in round i. We compute the expected value of the game $\mathsf{Fair}_{\pi,\mathcal{A}}$, conditioned on the event that $\mathcal{A}$ aborts in round i. Recall that whenever $\mathcal{A}$ aborts in round i it outputs $\mathsf{guess}_{\mathcal{A},i}$, whereas P_1 outputs b_{i-1}. Then, for every $r(n) > i > 0$ (where $r(n)$ denotes a polynomial bound on the number of rounds of π), we have that,

$$\begin{aligned}
&\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) \mid \mathsf{abort}_i^{\mathcal{A}}) \\
&\quad = \Pr[\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) = 1 \mid \mathsf{abort}_i^{\mathcal{A}}] - \Pr[\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) = -1 \mid \mathsf{abort}_i^{\mathcal{A}}] \\
&\quad = \Pr[\mathsf{guess}_{\mathcal{A},i} = f_0(x_0, x_1) \wedge b_{i-1} \neq f_1(x_0, x_1)] - \Pr[\mathsf{guess}_{\mathcal{A},i} \neq f_0(x_0, x_1) \wedge b_{i-1} = f_1(x_0, x_1)] \\
&\quad = \Pr[\mathsf{guess}_{\mathcal{A},i} = f_0(x_0, x_1) \wedge b_{i-1} \neq f_1(x_0, x_1)] + \Pr[\mathsf{guess}_{\mathcal{A},i} = f_0(x_0, x_1) \wedge b_{i-1} = f_1(x_0, x_1)] \\
&\qquad - \Pr[\mathsf{guess}_{\mathcal{A},i} = f_0(x_0, x_1) \wedge b_{i-1} = f_1(x_0, x_1)] - \Pr[\mathsf{guess}_{\mathcal{A},i} \neq f_0(x_0, x_1) \wedge b_{i-1} = f_1(x_0, x_1)] \\
&\quad = \Pr[\mathsf{guess}_{\mathcal{A},i} = f_0(x_0, x_1)] - \Pr[b_i = f_1(x_0, x_1)] \leq \mu(n)
\end{aligned}$$

for some negligible function $\mu(\cdot)$, where the last inequality holds due to the fact that π maintians the limited gradual release property. Moreover, when $i = 0$, there is no communication and thus both parties guess the correct output with probability 1/2. We therefore conclude that,

$$\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}} \mid \mathsf{abort}_0^{\mathcal{A}}) = 0 \leq \mu(n)$$

This implies that,

$$\begin{aligned}
\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n)) &= \sum_{i=0}^{r(n)} \Pr\left[\mathsf{abort}_i^{\mathcal{A}}\right] \cdot \mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) \mid \mathsf{abort}_i^{\mathcal{A}}) \\
&\leq \mu(n) \cdot \sum_{i=0}^{r(n)} \Pr\left[\mathsf{abort}_i^{\mathcal{A}}\right] \leq \mu(n)
\end{aligned}$$

where the above is true for any adversary, and thus the protocol is fair. ■

For conclusion, we note that the above theorems also hold for the setting of simultaneous channel, where the parties send their messages at the same time. We further stress that our notion of fairness is also meaningful in settings where the parties are not guaranteed to always learn the correct output, even when playing honestly. As hinted above, we consider a setting where a single protocol execution is completed such that only *one* of the parties is guaranteed to learn its correct output instead of both. Yet, it is guaranteed that overall, after multiple executions, both parties learn their correct output with roughly the same probability. This can be viewed as a way of capturing the parties' *expected* utilities. Surprisingly, our notion of fairness *can be achieved* in such limited and restricted settings as demonstrated next.

4.4 A New Notion of Simulation-Based Security

In this section, we modify the classical definition of secure computation with fairness [13] following the ideal/real model paradigm. Our goal is to present an alternative definition that captures the idea of "limited gradual release". Loosely speaking, a protocol meets security with fairness in the classic sense if both parties learn their output *concurrently.* Although very desirable, this notion of fairness cannot be realized *in general* in the two-party setting (not even when malicious behavior is restricted to early abortion). A potential reason for this failure may be due to the fact that realizing this notion requires a point in the computation where both parties move from a state of no knowledge of the output to a full knowledge of it. Instead, our definition allows the parties to gradually learn partial information on their desired outputs - but do so in a way that is "fair". More specifically, our definition for fairness captures the scenario where both parties gain an equivalent partial information in each step of the protocol.

Two challenges with formulating such a definition are: (1) How to capture the notion of guessing in the ideal setting (recall that the parties are instructed to guess their output in case of a prematurely abort). Due to this guessing, the parties may return incorrect values, implying that the trusted party should output incorrect values with the same probability as well. (2) How to define fairness with respect to partial information.

The new definition. Our new definition supports the following change with respect to [13]. In the ideal model, in addition to having the parties send their inputs to the trusted party, the ideal adversary (i.e., the simulator), sends a sampling PPT machine M for which the trusted party invokes on the parties' inputs, and sends back this outcome. Namely, the trusted party uses M to determine the parties' outputs. Finally, the honest party returns the output received from the trusted party, whereas the simulator outputs an arbitrary value (w.l.o.g., it outputs the adversary's view). In order for our definition to make sense in the fair setting, we require that M should be "fair" in the sense that both parties learn the same amount of information about their outputs, so that the guessing algorithms of both parties could be simulated. Note that the notion of fairness that M keeps may differ and depends on the fairness notion obtained by the overall definition. Below, we introduce an instantiation for M that matches our specific needs.

We assume that the simulator in the ideal setting always sends the trusted party machines that support the specified properties. Alternatively, we could have asked the trusted party to run M super polynomially many times in order to verify that these properties indeed hold. In case they are not, the trusted party could have output a special symbol indicating that and thus, forcing the simulator to always send such a machine. The disadvantage with this approach is that the trusted party runs in super polynomial time implying that composition theorems fail to apply. We therefore choose to parameterize our definition with simulators that always send machines as specified. Therefore, proving that a given protocol satisfies this definition should be followed by a proof that the simulator indeed outputs machines as required. See section 4.5.2 for a proof of a protocol that meets this definition.

The input distribution. In the traditional setting, the input is taken to be arbitrary, and security is expected to hold with respect to *any* input. In contrast, here we only require security to hold with respect to some collection of distribution ensembles, denoted by $\mathcal{D}^{\mathsf{f}}_{f} = \{D_z\}_z$ (see Definition 4.1 for one example). Therefore, we parameterize the random variable for the adversary's view with such a collection as well. This is done in order to make the intuitive concept of "comparable information gain" rigorously meaningful.

The auxiliary information. In the traditional setting, only the adversary is allowed to hold auxiliary information. In contrast, in our setting we consider the case where both parties may hold such information. It is stressed that the function value does not depend on the auxiliary input. Still, the protocol may depend on the auxiliary information the parties hold (see for instance a protocol in Section 4.5.2). This approach allows capturing the auxiliary information that the parties hold on each other's input. The security of the protocol is measured with respect to the certain type of the auxiliary information. More formally, let D_z be as above and let $z = (X_0, X_1)$ where the input x_i of the ith party is chosen from the set X_i for all $i \in \{0,1\}$. Then, additionally, we consider families of auxiliary functions $\mathsf{aux}^z_f = \{f_z(x_0, x_1)\}_{z\in\{0,1\}^n}$ where $f_z(x_0, x_1) = (z_0, z_1)$ and P_0 is given z_0, whereas P_1 is given z_1 for auxiliary inputs.

Execution in the ideal model. An ideal execution for the computation of f, involving parties P_0 and P_1 and ideal adversary S, proceeds as follows:

Inputs. P_0, P_1 are given x_0, x_1, sampled according to input distribution D_z. Moreover, each party holds auxiliary information z_i, computed according to some auxiliary input function $f_z(x_0, x_1) = (z_0, z_1)$.

Send inputs to the trusted party. Both parties send their inputs and auxiliary inputs to the trusted party.

The simulator sends a sampling machine. In addition to the adversary's input, the simulator $\mathcal{S}$ sends the trusted party a description of a PPT machine M that takes for input $(x'_0, x'_1, z'_0, z'_1, r')$, for x'_0, x'_1 the inputs of the parties that were sent to the trusted party, z'_0, z'_1 the auxiliary information of the parties that were sent to the trusted party, and r' the random string chosen by the trusted party. Denote by (y'_0, y'_1) the machine's outputs.

Trusted parties send outputs. The trusted party picks $r \in \{0,1\}^n$ uniformly at random, computes $M(x_0, x_1, z_0, z_1, r)$ and gets back two values (y_0, y_1). It then sends the simulator $\mathcal{S}$ the value y_{i^*} and the value y_{1-i^*} to P_{1-i^*}.

Outputs. The honest party outputs whatever the trusted party sent it, whereas $\mathcal{S}$ outputs the view of the corrupted party. We note that the simulator may output any value and it is not restricted to generate an execution that yields y_{i^*}.

Let $\mathbf{NIdeal}_{f,\mathcal{S},\mathcal{D}^{\mathsf{f}}_f,\mathsf{aux}^z_f}(\vec{x})$ denote the random variable consisting of the output of the simulator and the honest party following an execution in the ideal model as described above, were the inputs were chosen according to some input distribution from $\mathcal{D}^{\mathsf{f}}_f$, and the auxiliary inputs were chosen according the appropriate auxiliary input function from aux^z_f.

Execution in the real model. We next consider the real model in which a two-party protocol π is executed by parties P_0, P_1 (and there is no trusted party). The inputs and auxiliary inputs of the parties in the real execution are determined exactly the same as the inputs and auxiliary inputs in the ideal model. In the real execution, the adversary $\mathcal{A}$ controls party P_{i^*} and thus sees the input and the auxiliary information of that party. The adversary sends all messages on behalf of this party, following an arbitrary polynomial-time strategy. The honest party follows the instructions of π. Let f be as above and let π be a two-party protocol computing f. Let $\mathcal{A}$ be a non-uniform probabilistic polynomial-time machine. We let $\mathbf{Real}_{\mathcal{A},\pi,\mathcal{D}^{\mathsf{f}}_f,\mathsf{aux}^z_f}(\vec{x})$ denote the random variable consisting of the view of the adversary (corrupted party) and the honest party when following an execution of π.

Comparing executions of both models. As usual for the ideal/real paradigm [13], security is obtained by comparing the execution in the ideal world to the execution in the real world.

Definition 4.12 *Protocol π is said to* securely compute *f if for every* PPT *adversary $\mathcal{A}$ in the real model, there exists a* PPT *simulator $\mathcal{S}$ in the ideal model, such that:*

$$\left\{\mathbf{NIdeal}_{f,\mathcal{S},\mathcal{D}^{\mathsf{f}}_f,\mathsf{aux}^z_f}(\vec{x})\right\}_{\vec{x},z\in\{0,1\}^*} \overset{\mathrm{c}}{\equiv} \left\{\mathbf{Real}_{\pi,\mathcal{A},\mathcal{D}^{\mathsf{f}}_f,\mathsf{aux}^z_f}(\vec{x})\right\}_{\vec{x},z\in\{0,1\}^*}$$

where the ideal execution uses any (adversatively chosen) sampling machine M that satisfies Definitions 4.13, 4.14.

Partial fairness. Finally, we note that Definition 4.12 can be generalized to capture the notion of "partial fairness". In this case, both parties learn some partial information about the correct output, such that the adversary is allowed to learn some limited additional information. In this case, there is a (restricted) imbalance between the parties. One way to extend Definition 4.12 to capture this scenario is by relaxing the fairness requirement from the sampling algorithm M. This definition has an advantage over the [21] definition, in that the other security aspects of the computation (such as correctness or secrecy) are guaranteed with only negligible error. Nevertheless, it is tailored to this specific setting. More precisely, we say that M is ϵ-partial fair if there exists a negligible function $\mu(\cdot)$ such that for all sufficiently large n's it holds that:

$$\left|\Pr\left[y_{i^*} = f_{i^*}(x_0, x_1)\right] - \frac{1}{2}\right| \leq \Pr\left[y_{1-i^*} = f_{1-i^*}(x_0, x_1)\right] - \frac{1}{2} + \epsilon(n) + \mu(n).$$

Then, we say that a protocol is secure and ϵ-fair if it satisfies Definition 4.12, when $\mathcal{S}$ is restricted to send machines that are ϵ-partial fair. Note that all our other fairness notions discussed in this section can be extended to handle partial fairness respectively. For instance, our game-based definition (cf. Definition 4.6) can be extended by requiring that $\mathrm{E}(\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n)) \leq \mu(n) + \epsilon(n)$.

Even though partial fairness has been extensively studied in the literature [4, 7, 16, 12, 21] (just to state a few), only two works [12, 21] formalized this notion in the ideal/real paradigm. We note that the later work is more general than our work, since it captures a large class of functionalities over various domains and ranges and does not assume a particular input or auxiliary information distributions. Specifically, as noted earlier this definition compares a real execution of the protocol with a fair ideal execution and requires that no efficient distinguisher can distinguish between the two executions with probability non-negligibly better than $1/p$. Nevertheless, a drawback of this approach is that it allows a total break of security with probability $1/p$. In contrast, our definition below is much less general and very limited (in the sense that it considers concrete input distributions and auxiliary information), yet it captures the *exact* notion of fairness. An interesting open question is whether the two definitions can be incorporated in order to obtain a general definition that only allows a limited breach of fairness.

4.4.1 Simulation-Based Security with Fairness

By itself, Definition 4.12 does not ensure any level of security, unless the restrictions on M are explicitly specified. In this section, we restrict attention to the input distribution described in Definition 4.1 and auxiliary input that contains the set of two possible inputs of the other party, and specify the properties M should maintain for this case. In particular, given $z = (x_0^0, x_0^1, x_1^0, x_1^1)$, the input of P_0 is uniformly chosen from x_0^0, x_0^1, whereas the input of P_1 is uniformly chosen out of x_1^0, x_1^1. Therefore, the auxiliary information is the function $f_{x_0^0,x_0^1,x_1^0,x_1^1}(x_0, x_1) = ((x_1^0, x_1^1), (x_0^0, x_0^1))$. We note that it is possible to write the conditions on M in terms of general auxiliary input, but we prefer not to do that since we wish to avoid the additional complexity (and since our solution below works only with respect to this specific auxiliary input anyhow). Moreover, for the specific case discussed here, we obtain simulation-based security with fairness, where security is derived by the properties we require below from M. We begin with a definition for correct sampling machine, followed with a definition of fair sampling machine.

Definition 4.13 (Correct sampling machine) *Let P_{i^*} be the corrupted party, let $x_0^0, x_0^1, x_1^0, x_1^1$ be the input tuple, let x_{1-i^*} be the input of P_{1-i^*}, and let (y_0, y_1) be the outputs of M. We say that M is a* correct sampling machine *if $y_{1-i^*} \in \{f_{1-i^*}(x_{1-i^*}, x_{i^*}^0), f_{1-i^*}(x_{1-i^*}, x_{i^*}^1)\}$.*

This property is required to ensure the correctness of the examined protocol.

Definition 4.14 (Fair sampling machine) *Let $x_0^0, x_0^1, x_1^0, x_1^1$ be the input tuple, let x_0, x_1 be the inputs of the parties, and let (y_0, y_1) be the outputs of M. We say that M is* fair sampling machine *if there exists a negligible function $\mu(\cdot)$ such that for all sufficiently large n's it holds that:*

$$\left| \Pr\left[y_{i^*} = f_{i^*}(x_0, x_1)\right] - \frac{1}{2} \right| \leq \Pr\left[y_{1-i^*} = f_{1-i^*}(x_0, x_1)\right] - \frac{1}{2} + \mu(n) \tag{3}$$

where $(y_0, y_1) = M(x_0, x_1, z_0, z_1, r)$, the adversary controls party i^* and the probability is taken over the random coins of M.

Naturally, fairness can be defined by restricting M to give the adversary the correct output with the same probability as giving it to the honest party. However, since the adversary may output an arbitrary value after receiving the output of the corrupted party this requirement is insufficient. In fact, the sampling machine M may be designed such that it would always return an incorrect output for both parties, enabling the adversary to view this value as an "advice" and modify it into the correct outcome (based on its auxiliary information). In this case, the honest party always outputs an incorrect output, whereas the adversary always outputs the correct output and so, the fairness property is breached.

We avoid these scenarios by always letting the honest party learn the correct output with probability greater than $1/2$. In particular, if M gives the adversary the correct output with probability $p > 1/2$, then the honest party receives the correct output with the same probability. In contrast, if the adversary learns the correct output with probability $p < 1/2$, then the honest party learns the correct value with probability $1 - p$. That is, the absolute value applied on the left term is only meaningful in case the adversary M returns the correct output of the adversary with probability p that is smaller than $1/2$. In this case, it holds that the probability in which the honest party learns its correct output is at least $1-p$. Therefore, any manipulation on M's output will not gain the adversary any advantage.

In Claim 4.15 we formalize this intuition and show that for any strategy that the adversary may follow, the probability that it outputs the correct value is bounded by the probability that the honest party outputs the correct value, implying that fairness is guaranteed. More formally, we show that any (computationally unbounded) machine $\mathcal{G}$ cannot guess the adversary's correct output with probability, that is greater than the probability the honest party learns its correct output. Namely, let $\mathcal{G}$ denotes a machine that gets the input of the corrupted party; $x_{i^*}^{b_{i^*}}$, the tuple $(x_0^0, x_0^1, x_1^0, x_1^1)$ and the output of the corrupted party y_{i^*}, as received from M, and outputs a guess for the output $f_{i^*}(x_0^{b_0}, x_1^{b_1})$, denoted by $\mathbf{Output}_{\mathcal{G}}$, then

Claim 4.15 *For any* PPT *machine M satisfying Eq. (3) and any machine $\mathcal{G}$ it holds that*

$$\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_{i^*}(x_0^{b_0}, x_1^{b_1})\right] \leq \Pr\left[y_{1-i^*} = f_{1-i^*}(x_0^{b_0}, x_1^{b_1})\right] + \mu(n)$$

where b_0, b_1 are chosen uniformly at random from $\{0,1\}$, and (y_0, y_1) are the output of M.

Proof: Without loss of generality, assume that $i^* = 0$ and fix the adversary's input to $x_0^{b_0}$. Moreover, recall that y_{i^*} is the value that $\mathcal{G}$ receives from M. Then, define by:

- $\alpha \overset{\text{def}}{=} \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^0) \mid y_0 = f_0(x_0^{b_0}, x_1^0)\right]$,
- $\beta \overset{\text{def}}{=} \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^1) \mid y_0 = f_0(x_0^{b_0}, x_1^1)\right]$.

That is, in case that the machine $\mathcal{G}$ always outputs the value that was given as an output from M (y_0). Recall that $f_0(x_0^{b_0}, x_1^0) \neq f_0(x_0^{b_0}, x_1^1)$ due to the non-triviality of f. This implies that,

- $1 - \alpha = \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^1) \mid y_0 = f_0(x_0^{b_0}, x_1^0)\right]$,
- $1 - \beta = \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^0) \mid y_0 = f_0(x_0^{b_0}, x_1^1)\right]$.

Finally, let $p \overset{\text{def}}{=} \Pr[y_0 = f_0(x_0^{b_0}, x_1^{b_1})]$ denotes the probability that the machine M gave $\mathcal{G}$ the correct output. Then, we compute the probability that the adversary outputs the correct output. We have that

$$\begin{aligned}\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] &= \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1}) \mid b_1 = 0\right] \cdot \Pr[b_1 = 0] \\ &\quad + \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1}) \mid b_1 = 1\right] \cdot \Pr[b_1 = 1].\end{aligned}$$

Consider the probability $\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1}) \mid b_1 = 0\right]$. In this case the correct output is $f_0(x_0^{b_0}, x_1^0)$. Then,

$$\begin{aligned}\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^0)\right] &= \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^0) \mid y_0 = f_0(x_0^{b_0}, x_1^0)\right] \cdot \Pr\left[y_0 = f_0(x_0^{b_0}, x_1^0)\right] \\ &\quad + \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^0) \mid y_0 = f_0(x_0^{b_0}, x_1^1)\right] \cdot \Pr\left[y_0 = f_0(x_0^{b_0}, x_1^1)\right] \\ &= \alpha \cdot p + (1 - \beta) \cdot (1 - p).\end{aligned}$$

On the other hand, when $b_1 = 1$, the correct output is $f_0(x_0^{b_0}, x_1^1)$ and so, the probability that the adversary succeeds in this case is

$$\begin{aligned}\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^1)\right] &= \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^1) \mid y_0 = f_0(x_0^{b_0}, x_1^1)\right] \cdot \Pr\left[y_0 = f_0(x_0^{b_0}, x_1^1)\right] \\ &\quad + \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^1) \mid y_0 = f_0(x_0^{b_0}, x_1^0)\right] \cdot \Pr\left[y_0 = f_0(x_0^{b_0}, x_1^0)\right] \\ &= \beta \cdot p + (1 - \alpha) \cdot (1 - p).\end{aligned}$$

Therefore, the probability that the adversary outputs the correct output is computed by

$$\begin{aligned}\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] &= \frac{1}{2} \cdot (\alpha \cdot p + (1 - \beta) \cdot (1 - p) + \beta \cdot p + (1 - \alpha) \cdot (1 - p)) \\ &= \frac{1}{2} \cdot (\alpha p + 1 - \beta - p + p\beta + \beta p + 1 - \alpha - p + p\alpha) \\ &= \frac{1}{2} \cdot (1 + 2\alpha p + 2\beta p - 2p + 1 - \alpha - \beta) \\ &= \frac{1}{2} + \frac{1}{2} \cdot (2p \cdot (\alpha + \beta - 1) + (1 - \alpha - \beta)) = \frac{1}{2} + (p - \frac{1}{2}) \cdot (\alpha + \beta - 1)\,.\end{aligned}$$

Relying on the fact that $0 \leq \alpha \leq 1,\ 0 \leq \beta \leq 1$ we get that $-1 \leq \alpha + \beta - 1 \leq 1$ and so, $|\alpha + \beta - 1| \leq 1$. Using the fact that $|A \cdot B| = |A| \cdot |B|$, we get that:

$$\begin{aligned}\left|\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] - 1/2\right| &= |(p - 1/2) \cdot (\alpha + \beta - 1)| \\ &= |p - 1/2| \cdot |\alpha + \beta - 1| \\ &\leq |p - 1/2|\,.\end{aligned}$$

Using the fact that M satisfies Eq. (3), we have that:

$$\left|\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] - 1/2\right| \leq \Pr\left[y_1 = f_1(x_0^{b_0}, x_1^{b_1})\right] - 1/2 + \mu(n)$$

and so, in case that $\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] \geq 1/2$, it holds that:

$$\begin{aligned}\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] - 1/2 &\leq \Pr\left[y_1 = f_1(x_0^{b_0}, x_1^{b_1})\right] - 1/2 + \mu(n) \\ \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] &\leq \Pr\left[y_1 = f_1(x_0^{b_0}, x_1^{b_1})\right] + \mu(n).\end{aligned}$$

In case that $\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] \leq 1/2$, we can use the fact that:

$$\begin{aligned}
\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] - 1/2 &\geq -\Pr\left[y_1 = f_1(x_0^{b_0}, x_1^{b_1})\right] + 1/2 - \mu(n) \\
\Pr\left[y_1 = f_1(x_0^{b_0}, x_1^{b_1})\right] + \mu(n) &\geq 1 - \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right]
\end{aligned}$$

However, since $\Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] \leq 1/2$ we have that $1 - \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] \geq \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right]$, and so:

$$\Pr\left[y_1 = f_1(x_0^{b_0}, x_1^{b_1})\right] + \mu(n) \geq 1 - \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right] \geq \Pr\left[\mathbf{Output}_{\mathcal{G}} = f_0(x_0^{b_0}, x_1^{b_1})\right].$$

■

4.4.2 Simulation-Based Definition Implies Game-Based Fairness

In this section we show that Definition 4.12, when instantiated with Definitions 4.13-4.14, implies Definition 4.6. Formally,

Theorem 4.16 *Let f, π be as above. Then, if π is simulatable (in the sense of Definition 4.12, instantiated with Definitions 4.13-4.14), then π is fair with respect to the Game-Based (in the sense of Definition 4.6).*

Proof: Generally, we show that if there exists an adversary $\mathcal{A}$ that succeeds in the game-based definition, then there does not exists a simulator for $\mathcal{A}$. That is, assume by contradiction that there exists an adversary $\mathcal{A}$, controlling party P_0 (w.l.o.g.), for which Definition 4.6 is not met. Namely, it holds that

$$\Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) = 1\right] - \Pr\left[\mathsf{Fair}_{\pi,\mathcal{A}}(x_0^0, x_0^1, x_1^0, x_1^1, n) = -1\right] \geq \epsilon(n).$$

Let $\vec{x} = (x_0^0, x_0^1, x_1^0, x_1^1, n, x_0^{b_0}, x_1^{b_1})$, and denote by $\mathbf{Output}_{\pi,i}(\vec{x})$ output of P_i in an execution where P_0 has input $x_0^{b_0}$, and P_1 is invoked with input $x_1^{b_1}$. Then, by the contradiction assumption it holds that,

$$\begin{aligned}
&\Pr\left[\mathbf{Output}_{\pi,0}(\vec{x}) = f_0(\vec{x}) \wedge \mathbf{Output}_{\pi,1} \neq f_1(\vec{x})\right] \\
&\quad - \quad \Pr\left[\mathbf{Output}_{\pi,1}(\vec{x}) = f_1(\vec{x}) \wedge \mathbf{Output}_{\pi,0}(\vec{x}) \neq f_0(\vec{x})\right] \geq \epsilon(n)
\end{aligned}$$

Adding and subtracting $\Pr[\mathbf{Output}_{\pi,0}(\vec{x}) = f_0(\vec{x}) \wedge \mathbf{Output}_{\pi,1}(\vec{x}) = f_1(\vec{x})]$ to the left, we get that

$$\Pr\left[\mathbf{Output}_{\pi,0}(\vec{x}) = f_0(\vec{x})\right] - \Pr\left[\mathbf{Output}_{\pi,1}(\vec{x}) = f_1(\vec{x})\right] \geq \epsilon(n) \tag{4}$$

In other words, P_0 (when controlled by $\mathcal{A}$), learns the correct output with a non-negligible advantage over party P_1. In order to conclude the proof, we present two distinguishers, D_C and D_H. Namely, D_C checks weather the output of party P_0 is correct (i.e., checks the corrupted party's output), whereas D_H checks whether the output of P_1 is correct (i.e., checks the honest party's output). Now, since there is a non-negligible difference between the probabilities that the parties return the correct output in the real world, whereas for any simulator in the ideal model, both parties learn the correct output with almost the same probability, it must hold that one of the following distinguishers distinguishes the executions with a non-negligible difference. More formally:

The distinguisher D_C.

- **Input:** The index for the ensemble: $x_0^{b_0}, x_1^{b_1}$, the view of the corrupted party $r; m_1, \ldots, m_\ell$, the output of the corrupted party y_0, the output of the honest party y_1.
- **The distinguisher:**
  - If $y_0 = f_0(x_0^{b_0}, x_1^{b_1})$, output 1. Otherwise, output 0.

The distinguisher D_H.

- **Input:** The index for the ensemble: $x_0^{b_0}, x_1^{b_1}$, the view of the corrupted party $r; m_1, \ldots, m_\ell$, the output of the corrupted party y_0, the output of the honest party y_1.
- **The distinguisher:**
  - If $y_1 = f_1(x_0^{b_0}, x_1^{b_1})$, output 1. Otherwise, output 0.

The real world. We compute the probability that D_C and D_H output 1 in the real world. That is,

$$\Pr[D_C(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] = \Pr[y_0 = f_0(\vec{x})] = \Pr[\mathbf{Output}_{\pi,0} = f_0(\vec{x})]$$

Similarly,

$$\Pr[D_H(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] = \Pr[y_1 = f_1(\vec{x})] = \Pr[\mathbf{Output}_{\pi,1} = f_1(\vec{x})]$$

Using Eq. (4), we conclude that:

$$\Pr[D_C(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] > \Pr[D_H(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] + \epsilon(n) \tag{5}$$

The ideal world. By Definition 4.12, $\mathcal{S}$ always sends the trusted party a fair sampling machine M. Therefore, using Claim 4.15 there exists a negligible function $\mu(\cdot)$ such that for all sufficiently large n's it holds that:

$$\Pr\left[y_0 = f_0(x_0, x_1)\right] \leq \Pr\left[y_1 = f_1(x_0, x_1)\right] + \mu(n)$$

(note that here (y_0, y_1) denote the outputs of the parties in the ideal executions and not the output of the sampling machine M). Using the descriptions of D_C, D_H, we have that

$$\Pr[D_C(\mathbf{NIdeal}_{f,\mathcal{S}}(\vec{x})) = 1] \leq \Pr[D_H(\mathbf{NIdeal}_{f,\mathcal{S}}(\vec{x})) = 1] + \mu(n)$$

Concluding the proof. We showed above the following statements:

- There exists a negligible function $\mu(\cdot)$ for which:

$$\Pr[D_C(\mathbf{NIdeal}_{f,\mathcal{S}}(\vec{x})) = 1] \leq \Pr[D_H(\mathbf{NIdeal}_{f,\mathcal{S}}(\vec{x})) = 1] + \mu(n) \tag{6}$$

- From Eq. (5), there exists a non-negligible function $\epsilon(\cdot)$ such that:

$$\Pr[D_C(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] > \Pr[D_H(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] + \epsilon(n) \tag{7}$$

We therefore have two cases:

1. The distinguisher D_H distinguishes successfully. That is, there exists a non-negligible function $\epsilon'(\cdot)$ such that for infinitely many n's it holds that:

$$|\Pr[D_H(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] - \Pr[D_H(\mathbf{NIdeal}_{\pi,\mathcal{A}}(\vec{x})) = 1]| > \epsilon'(n)$$

2. The distinguisher D_H does not distinguish successfully, implying that there exists a negligible function $\mu'(\cdot)$ such that for all sufficiently large n's it holds that:

$$|\Pr[D_H(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] - \Pr[D_H(\mathbf{NIdeal}_{\pi,\mathcal{A}}(\vec{x})) = 1]| < \mu'(n).$$

   Thus, we have that:

$$\Pr[D_H(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] > \Pr[D_H(\mathbf{NIdeal}_{f,\mathcal{A}}(\vec{x})) = 1] - \mu'(n).$$

   Combining it with Eq. (7), we have that:

$$\Pr[D_C(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] > \Pr[D_H(\mathbf{NIdeal}_{\pi,\mathcal{A}}(\vec{x})) = 1] + \epsilon(n) - \mu'(n).$$

   Finally, using Eq. (6), we have:

$$\Pr[D_C(\mathbf{Real}_{\pi,\mathcal{A}}(\vec{x})) = 1] > \Pr[D_C(\mathbf{NIdeal}_{\pi,\mathcal{A}}(\vec{x})) = 1] + \epsilon(n) - \mu(n) - \mu'(n) > \epsilon(n)/2$$

   for all sufficiently large n's, implying that D_C distinguishes successfully.

We showed that for any simulator, there exists a distinguisher that distinguishes successfully between the real and the ideal executions with a non-negligible probability, contradicting the assumption that the protocol is simulatable. ■

4.5 The Feasibility of Our Definition

In this section, we study our new game-based cryptographic definition of fairness in a cryptographic context. Our starting point is any correct protocol, where both parties learn their output if playing honestly. We then show, that by relaxing the (negligibly close to) perfect completeness requirement, which implies that the parties should (almost) always learn their output if playing honestly, we can fully characterize the set of two-party protocols according partial correctness. Informally,

1. In case correctness holds with probability that is non-negligibly greater than 1/2, we present an impossibility result, saying that there does not exists a fair protocol with this probability of correctness. This implies that the difficulties in designing fair protocols are already embedded within the fail-stop setting. Stating differently, these difficulties already emerge whenever early abort is permitted.

2. On the positive side, in case correctness holds with probability that is smaller equals to 1/2, we show how to design a fair protocol that meets our notion of fairness. Specifically, we present a family of such protocols, parameterized by this probability of correctness. The implications of this is that there may be still hope for the fail-stop setting with respect to designing fair protocols.

4.5.1 An Impossibility Result

In this section we demonstrate that our game-based definition for fairness cannot be achieved for protocols that guarantee correctness with probability greater than 1/2. Before turning to our main theorem we present a definition of an α-correct protocol.

Definition 4.17 *Let f be a non-trivial two-party function, and let π be a two-party protocol. We say that the protocol π is a α-*$\mathsf{correct}$ *for f if there exists a negligible function $\mu(\cdot)$ such that for all sufficiently large x_0, x_1, n such that $|x_0| = |x_1| = n$,*

$$\left|\Pr[\mathbf{Output}_{\pi,0}(x_0,x_1) = f_0(x_0,x_1) \wedge \mathbf{Output}_{\pi,1}(x_0,x_1) = f_1(x_0,x_1)] - \alpha\right| \leq \mu(n)$$

where $\mathbf{Output}_{\pi,i}(x_0,x_1)$ denote the output of party P_i when invoked on input x_i, while P_{1-i} is invoked on x_{1-i}, and both parties are honest.

Note that, for the impossibility result below, it is sufficient to restrict ourselves to fail-stop adversaries that guess their output according to the instructions of the protocol. That is, their default output are the same default outputs as the honest parties, and thus we use the notation of a_i, b_i. Our theorem of impossibility:

Theorem 4.18 *Let f be a non-trivial two-party function. Then, for every non-negligible function $\epsilon(\cdot)$ and every $\alpha > 1/2 + \epsilon(n)$, there does not exist an α-*$\mathsf{correct}$ *protocol which is also fair (in the sense of Definition 4.6), with a polynomial round complexity.*

Proof: Let f be a function as above, let $\epsilon(\cdot)$ be a non-negligible function and let π be a fair α-correct protocol for some $\alpha > 1/2 + \epsilon(n)$. Furthermore, let $r(n)$ be a polynomial upper bound on the number of rounds of the protocol. Then, there exists a negligible function $\mu'(\cdot)$ such that for all sufficiently large x_0, x_1, n it holds that,

$$\Pr[\mathbf{Output}_{\pi,0}(x_0,x_1) = f_0(x_0,x_1) \wedge \mathbf{Output}_{\pi,1}(x_0,x_1) = f_1(x_0,x_1)] \geq \alpha - \mu'(n).$$

We next show that π must have an exponential number of rounds on the average. We consider an execution of π within game $\mathsf{Fair}(x_0^0, x_0^1, x_1^0, x_1^1, n)$. Then, from the facts that the function is non-trivial and the output values are distinct, it must hold that

$$\Pr\left[a_0 = f_0(x_0,x_1)\right] = \frac{1}{2} \qquad \text{and} \qquad \Pr\left[b_0 = f_1(x_0,x_1)\right] = \frac{1}{2}. \tag{8}$$

From Theorem 4.10, it is implied that π maintains the limited gradual release property which means that there exists a negligible function $\mu(\cdot)$, such that for all $i < r(n)$ and every $x_0^0, x_0^1, x_1^0, x_1^1, n$ as above, it holds that

- $\Pr\left[a_i = f_0(x_0,x_1)\right] \leq \Pr\left[b_{i-1} = f_1(x_0,x_1)\right] + \mu(n)$, and
- $\Pr\left[b_i = f_1(x_0,x_1)\right] \leq \Pr\left[a_i = f_0(x_0,x_1)\right] + \mu(n)$.

where x_0, x_1 are as in Definition 4.1.

In other words, for every $i < r(n)$ it holds that,

$$\Pr\left[a_i = f_0(x_0,x_1)\right] \leq \Pr\left[b_{i-1} = f_1(x_0,x_1)\right] + \mu(n) \leq \Pr\left[a_{i-1} = f_0(x_0,x_1)\right] + 2\mu(n)$$

Repeating inductively this argument, and relying Eq. (8) we have that,

$$\Pr\left[a_i = f_0(x_0,x_1)\right] \leq \frac{1}{2} + 2i \cdot \mu(n) \qquad \text{and} \qquad \Pr\left[b_{i-1} = f_1(x_0,x_1)\right] \leq \frac{1}{2} + 2i \cdot \mu(n)$$

Let both denote the event that both parties learn their output (when both play honestly). Then, based on the fact that π is α-correct protocol, we know that,

$$\Pr[\neg\mathsf{both}] = 1 - \Pr[\mathsf{both}] \leq 1 - (\alpha - \mu'(n)) < \frac{1}{2} - \epsilon(n) + \mu'(n) < \frac{1}{2} - \frac{\epsilon(n)}{2}$$

where the latter is true since $\alpha > 1/2 + \epsilon(n)$ and for all sufficiently large n's, $\mu'(n) < \epsilon(n)/2$.

Next, we consider the probability that the number of rounds is less than $k \cdot r(n)$ for some $k > 1$, where $r(n)$ is some polynomial that upper bounds the number of rounds in the protocol. We have

$$\begin{aligned}
&\Pr[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n)] \\
&\quad = \Pr\left[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n) \mid \mathsf{both}\right] \cdot \Pr[\mathsf{both}] + \Pr[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n) \mid \neg\mathsf{both}] \Pr[\neg\mathsf{both}] \\
&\quad \leq \Pr\left[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n) \mid \mathsf{both}\right] + \frac{1}{2} - \frac{\epsilon(n)}{2}
\end{aligned}$$

where $\mathsf{Rounds}_\pi(x_0, x_1)$ denote the number of rounds of π when invoked on (x_0, x_1). In case that the number of rounds is less than $k \cdot r(n)$ and both occurs, then both $a_{kr(n)}$ and $b_{kr(n)-1}$ equal to the correct output. Therefore, we have that

$$\begin{aligned}
&\Pr\left[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n) \mid \mathsf{both}\right] \\
&\quad \leq \Pr[a_{kr(n)} = f_0(x_0, x_1) \wedge b_{kr(n)-1} = f_1(x_0, x_1)] \leq \Pr[a_{kr(n)} = f_0(x_0, x_1)] \\
&\quad \leq \frac{1}{2} + 2k \cdot r(n) \cdot \mu(n).
\end{aligned}$$

We conclude that,

$$\begin{aligned}
\Pr[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n)] &\leq \frac{1}{2} + 2k \cdot r(n) \cdot \mu(n) + \frac{1}{2} - \epsilon(n) + \mu'(n) \\
&\leq 1 + 2k \cdot r(n)\mu(n) - \frac{\epsilon(n)}{2}. \qquad (9)
\end{aligned}$$

On the other hand, there exists a negligible function $\mu''(n)$ such that for any $k > 1$

$$\Pr[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n)] > 1 - \mu''(n) > 1 - \frac{\epsilon(n)}{4}$$

where the latter is true for all sufficiently large n's. This is due to the fact that the protocol is completed after $r(n)$ rounds. Therefore,

$$1 - \frac{\epsilon(n)}{4} < \Pr[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n)] \leq 1 + 2k \cdot r(n)\mu(n) - \frac{\epsilon(n)}{2}$$

and thus,

$$\begin{aligned}
1 + 2k \cdot r(n)\mu(n) - \frac{\epsilon(n)}{2} &> 1 - \frac{\epsilon(n)}{4} \\
r(n) &> \frac{\epsilon(n)}{8k\mu(n)}
\end{aligned}$$

in contradiction to the fact that $r(n)$ is a polynomial. ■

We note that the above can be generalized easily for the simultaneous setting, and for expected polynomial time protocols; see Appendix A for the specific modifications of the proof for the later.

4.5.2 A Positive Result

Recall that the limited gradual release property implies that the probabilities of guessing the correct output at any given round, for polynomial-time protocols, cannot be increased "too much" during the execution when compared to the initial probability (that is computed based on an empty view).

This implies that the probability of guessing the correct output at any given round is negligibly close to 1/2, for any fair protocol (in the sense of Definition 4.6). Specifically, this protocol does not leak any information about the correct output during its execution (or otherwise, it would not be fair). Therefore, it may seem that this definition cannot be met by protocols that generate correct output, and indeed, we followed this intuition when proved the impossibility result in Section 4.5.1. Interestingly, we show that for relaxed correctness (i.e., lower equal than 1/2), *there exist* non-trivial functionalities that can be computed fairly in this setting. In the following, we present a fair protocol in which either both parties learn the correct output together, or alternatively neither party obtains a correct result. The case where in each execution exactly one party learns its correct output can also be achieved with fairness. More generally, denote by α the probability in which both parties should learn their outputs. Then, we show that for every $\alpha \leq 1/2$, there exists an α-correct protocol that is also fair, even in the non-simultaneous channel model. This relaxation is necessary to obtain fairness, as higher α values set a threshold for achieving this property (as shown in Section 4.5.1). Intuitively, the fact that each party does not know whether it has the correct output implies that a corrupted party would not have any incentive to abort after learning its output, since it does not give the honest party any new information anyway.

The protocol. The protocol is invoked over tuples of inputs with the distribution of choosing each input randomly out of a known pair. Let $x_0^0, x_0^1, x_1^0, x_1^1$ denote such an input tuple and denote by $x_0^{\mathsf{true}} \stackrel{\mathrm{def}}{=} f_0(x_0^{b_0}, x_1^{b_1}), x_0^{\mathsf{false}} \stackrel{\mathrm{def}}{=} f_0(x_0^{b_0}, x_1^{1-b_1})$, $x_1^{\mathsf{true}} \stackrel{\mathrm{def}}{=} f_1(x_0^{b_0}, x_1^{b_1})$, and $x_1^{\mathsf{false}} \stackrel{\mathrm{def}}{=} f_1(x_0^{1-b_0}, x_1^{b_1})$.

Then function f^α; formally defined below, sets the output of the parties such that both learn the correct output with probability α, as required from an α-correct protocol. Moreover, the parties realize function f^α via protocol $\pi^\alpha_{\mathtt{abort}}$, which is secure-with-abort.

For the special case where $\alpha = 0$ we get that in each execution, either P_0 or P_1 learn their correct output (but never both). This implies that correctness never holds since both parties never learn their correct output together. As for the other extreme case where $\alpha = 1/2$, the functionality ensures that either both parties learn their correct output at the same time (which occurs with probability 1/2), or both learn an incorrect output (with the same probability). For the general case, see the figure below.

The Ideal Functionality f^α

- **Input:** P_0 inputs $b_0, x_0^0, x_0^1, x_1^0, x_1^1$. P_1 inserts $b_1, x_0^0, x_0^1, x_1^0, x_1^1$.
- **The function:**
  - Toss a coin σ that equals 0 with probability 2α, and equals 1 with probability $1 - 2\alpha$.
  - If $\sigma = 0$ (*parties learn same output*) do:
    - Toss another coin τ_0 uniformly at random from $\{0, 1\}$.
    - If $\tau_0 = 0$: set the output of P_0, P_1 to be $(x_0^{\mathsf{true}}, x_1^{\mathsf{true}})$, respectively.
    - If $\tau_0 = 1$: set the output of P_0, P_1 to be $(x_0^{\mathsf{false}}, x_1^{\mathsf{false}})$, respectively.
  - If $\sigma = 1$ (*parties learn true and false outputs*) do:
    - Toss another coin τ_1 uniformly at random from $\{0, 1\}$.
    - Set the output of P_{τ_1} to be $x_{\tau_1}^{\mathsf{true}}$.
    - Set the output of $P_{1-\tau_1}$ to be $x_{1-\tau_1}^{\mathsf{false}}$.

Protocol 1 π^α*: an α-correct and fair protocol for f in the $f^\alpha_{\mathtt{abort}}$-hybrid model*

- Inputs: *P_0 holds input $x_0^{b_0}$, P_1 holds $x_1^{b_1}$.*
- Auxiliary input: *Both parties are given $x_0^0, x_0^1, x_1^0, x_1^1$.*
- The protocol:
  - *Engage in an execution of $f^{\alpha}_{\texttt{abort}}$ on inputs $b_0, x_0^0, x_0^1, x_1^0, x_1^1$ for P_0, and $b_1, x_0^0, x_0^1, x_0^0, x_1^0, x_1^1$ for P_1.*
  - *If P_i receives $\perp$ from the $f^{\alpha}_{\texttt{abort}}$-ideal function, it chooses its output uniformly at random from $\{x_i^{\mathsf{true}}, x_i^{\mathsf{false}}\}$. (Note that P_i knows these values yet it still cannot distinguish a true from a false output).*
  - *Otherwise, it outputs the value returned by $f^{\alpha}_{\texttt{abort}}$, and halts.*

Theorem 4.19 *Let f be a non-trivial two-party function. Then, for every $0 \leq \alpha \leq 1/2$, protocol π^{α} is an α-correct protocol in the f^{α}-hybrid model, and is simulatable (in the sense of Definition 4.12).*

Proof: We first show that for every α in this range, π^{α} is an α-correct protocol. Specifically, in case both parties play honestly, then both receive an output from f^{α}, which implies that both parties learn the correct output with probability α.

We now show that π^{α} satisfies definition 4.12. We first define the machine M that the simulator sends to the trusted party. This machine is hard-wired with the real adversary, together with its random coins. This makes the adversary deterministic within M and moreover, enables the simulator to extract the randomness of the adversary after receiving the output from the trusted party. We therefore describe M with an additional parameter $r_{\mathcal{A}}$ - denote the randomness of the adversary. Specifically, the simulator first selects the random coins for $\mathcal{A}$, and only then creates the machine M to be sent to the trusted party.

The machine $M(\mathcal{A}, r_{\mathcal{A}})$.

- **Input:** *x_0', x_1' and the tuple $(x_0^0, x_0^1, x_1^0, x_1^1)$.*
- **The machine:**
  - *M extracts from x_0', x_1' the bits (b_0, b_1) such that $x_0^{b_0} = x_0'$, $x_1^{b_1} = x_1'$.*
  - *M "invokes" the hardwired adversary $\mathcal{A}$ with random bits $r_{\mathcal{A}}$ and input $(x_0^{b_i^*}, x_0^0, x_0^1, x_1^0, x_1^1)$. The machine $\mathcal{A}$ outputs $(b_{i^*}, x_0^0, x_0^1, x_1^0, x_1^1)$.*
  - *M invokes function f^{α} on input $(b_0, x_0^0, x_0^1, x_1^0, x_1^1)$ for P_0 and $(b_1, x_0^0, x_0^1, x_1^0, x_1^1)$ for P_1. Let (y_0, y_1) be the outputs of this function.*
  - *M gives to the hardwired adversary $\mathcal{A}$ the output y_{i^*}.*
    - *If $\mathcal{A}$ does not abort, M outputs (y_0, y_1).*
    - *If $\mathcal{A}$ aborts, M tosses a coin b uniformly at random, and gives the honest party the evaluation of f on $x_{1-i^*}^{b_{1-i^*}}$ and $x_{i^*}^{b}$ and y_{i^*} to the ideal adversary.*
      (This step emulates the guess of the honest party when it does not receive an output from f^{α}).

We now proceed with the simulator $\mathcal{S}$. Recall that $\mathcal{S}$ needs to create the view of the corrupted party. In the real execution, the view of the corrupted party is the value that it gets from f^{α}. This is exactly the output of M. Therefore, the simulator is straightforward. More formally, let $\mathcal{A}$ be a PPT adversary and let $r(n)$ be a polynomial that bounds the number of random coins that $\mathcal{A}$ uses. Then, simulator $\mathcal{S}$ is defined as follows.

The simulator $\mathcal{S}$.

- *$\mathcal{S}$ chooses uniformly at random $r \in_R \{0,1\}^{r(n)}$.*
- *$\mathcal{S}$ designs $M(\mathcal{A}, r)$ with adversary $\mathcal{A}$ hardwired in it together with the random bits r.*
- *$\mathcal{S}$ receives from the trusted party a value y.*
- *$\mathcal{S}$ invokes $\mathcal{A}$ with input $x_{i^*}^{b_{i^*}}$, auxiliary input $x_0^0, x_0^1, x_1^0, x_1^1$, and random coins r.*
- *The adversary outputs $(b_{i^*}, x_0^0, x_0^1, x_1^0, x_1^1)$ as an input to the ideal function f^α. $\mathcal{S}$ returns to $\mathcal{A}$ the value y, emulating the output given by the trusted party for f^α.*
- *$\mathcal{A}$ outputs* `continue` *or* `abort`*, and outputs a value y'.*
- *$\mathcal{S}$ outputs the view of the adversary: (r, y, y').*

Clearly, the simulator's output is identically distributed to the distribution in the hybrid model. All is left to show is that M is a valid sampling machine. Namely, that M is fair and correct. The correctness requirement is trivially achieved. We show that M is also fair, satisfying Eq. (3). Intuitively, the output of M for the corrupted party is always the output of function f^α. Recall that function f^α hands the corrupted party the correct output in the following cases:

- *Case 1:* When $\sigma = 0$, and $\tau_0 = 0$. (In this case, both parties receive the correct output).
- *Case 2:* When $\sigma = 1$ and $\tau_1 = i^*$. (In this case, only the corrupted party receives the correct output).

Therefore, we have that:

$$\begin{aligned}
\Pr\left[y_{i^*} = f_{i^*}(x_0^{b_0}, x_1^{b_1})\right] &= \Pr[\sigma = 0 \wedge \tau_0 = 0] + \Pr[\sigma = 1 \wedge \tau_1 = i^*] \\
&= \Pr[\sigma = 0] \cdot \Pr[\tau_0 = 0] + \Pr[\sigma = 1] \cdot \Pr[\tau_1 = i^*] \\
&= 2\alpha \cdot \frac{1}{2} + (1 - 2\alpha) \cdot \frac{1}{2} = \frac{1}{2} \; .
\end{aligned}$$

On the other hand, the honest party receives the correct output in the following cases:

- In case that the adversary does not abort, the output of the honest party is the output that it receives from function f^α. As the output of the corrupted party, the honest party receives the correct output from the function f^α with probability 1/2.
- In case that the adversary aborts, the machine M guesses the honest party's correct output with probability 1/2.

We note that the hardwired adversary is deterministic, and so the probability for the output of the honest party is taken over the random coins of f^α and the coins of M (for guessing the correct output in case that $\mathcal{A}$ aborts). In any case, let y_{1-i^*} be the output of the machine M for the honest party. We have that:

$$\Pr\left[y_{1-i^*} = f_{1-i^*}(x_0^{b_0}, x_1^{b_1})\right] = 1/2$$

The above shows that M is fair. This completes the proof. ■

References

[1] I. Abraham, D. Dolev, R. Gonen and J Y. Halpern. Distributed Computing Meets Game Theory: Robust Mechanisms for Rational Secret Sharing and Multiparty Computation. In *PODC*, pages 53–62, 2006.

[2] S. Agrawal and M. Prabhakaran. On Fair Exchange, Fair Coins and Fair Sampling. In *CRYPTO'13*, pages 259-276, 2013.

[3] G. Asharov and Y. Lindell. Utility Dependence in Correct and Fair Rational Secret Sharing In *Journal of Cryptology*, Volume 24, Issue 1 (2011), pages: 157-202.

[4] D. Beaver and S. Goldwasser, Multiparty computation with faulty majority. In *30th FOCS*, pages 468–473, 1989.

[5] Manuel Blum. How to exchange (secret) keys. *ACM Trans. Comput. Syst.*, 1(2):175–193, 1983.

[6] R. Cleve. Limits on the Security of Coin Flips when Half the Processors are Faulty. In 18*th STOC*, pages 364–369, 1986.

[7] R. Cleve. Controlled Gradual Disclosure Schemes for Random Bits and Their Applications In *CRYPTO'89*, pages 573–588, 1989.

[8] Y. Dodis, S. Halevi and Tal Rabin. A Cryptographic Solution to a Game Theoretic Problem. In *CRYPTO'00*, Springer-Verlag (LNCS 1880), pages 112–130, 2000.

[9] Y. Dodis and T. Rabin. Cryptography and Game Theory. In *Algorithmic Game Theory*, Cambridge University Press, 2007.

[10] G. Fuchsbauer, J. Katz and D. Naccache. Efficient Rational Secret Sharing in Standard Communication Networks. In the *7th TCC*, Springer-Verlag (LNCS 5978), pages 419–436, 2010.

[11] D. Fudenberg and J. Tirole. *Game Theory.* The MIT Press, 1991.

[12] J. A. Garay, P. D. MacKenzie, M. Prabhakaran and K. Yang. Resource fairness and composability of cryptographic protocols. In *3rd Theory of Cryptography Conference TCC 2006*, volume 3876 of LNCS, pages 404–428. Springer, 2006.

[13] O. Goldreich. *Foundations of Cryptography: Volume 2 – Basic Applications.* Cambridge University Press, 2004.

[14] O. Goldreich, S. Micali and A. Wigderson. How to Play any Mental Game – A Completeness Theorem for Protocols with Honest Majority. In *19th STOC*, pages 218–229, 1987.

[15] O. Goldreich and A. Kahan. How To Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.

[16] S. Goldwasser and L. A. Levin. Fair computation of general functions in presence of immoral majority. In *CRYPTO 1990*, Springer-Verlag (LCNS), 537, pages 77-93, 1991.

[17] S. Goldwasser and S. Micali. Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information. In *J. Comput. Syst. Sci.*, 28(2): 270-299, 1984.

[18] S. Goldwasser, S. Micali and C. Rachoff. The Knowledge Complexity of Interactive Proof Systems. In *SIAM J. Computing* 18(1): 186–208 1989.

[19] S. D. Gordon, C. Hazay, J. Katz and Y. Lindell. Complete fairness in secure two-party computation In *J. SIGACT News*, 43(1): pages 21-23, 2012.

[20] S. D. Gordon and J. Katz. Rational Secret Sharing, Revisited. In *Security and Cryptography for Networks (SCN)*, Springer-Verlag (LNCS 4116), pages 229–241, 2006.

[21] S. D. Gordon and J. Katz. Partial Fairness in Secure Two-Party Computation. In *J. Cryptology* 25(1): pages 14-40, 2012.

[22] R. Gradwohl, N. Livne and A. Rosen. Sequential Rationality in Cryptographic Protocols. In *FOCS*, pages: 623-632, 2010.

[23] A. Groce and J. Katz. Fair Computation with Rational Players. In *Eurocrypt*, pages 81-98, 2012.

[24] J. Halpern and V. Teague. Efficient Rational Secret Sharing in Standard Communication Networks. In *36th STOC*, pages 623–632, 2004.

[25] J. Halpern and R. Pass. Game Theory with Costly Computation. In *ICS*, pages 120-142, 2010.

[26] S. Izmalkov, M. Lepinski and S. Micali. Verifiably Secure Devices. In *5th TCC*, Springer-Verlag (LNCS 4948), pages 273–301, 2008.

[27] S. Izmalkov, S. Micali and M. Lepinski. Rational Secure Computation and Ideal Mechanism Design. In 46*th FOCS*, pages 585–595, 2005.

[28] J. Katz. Bridging Game Theory and Cryptography: Recent Results and Future Directions. In *5th TCC*, Springer-Verlag (LNCS 4948), pages 251–272, 2008.

[29] G. Kol and M. Naor. Games for exchanging information. In *40th STOC*, pages 423-432, 2008.

[30] G. Kol and M. Naor. Cryptography and Game Theory: Designing Protocols for Exchanging Information. In *5th TCC*, Springer-Verlag (LNCS 4948), pages 320–339, 2008.

[31] A. Lysyanskaya and N. Triandopoulos. Rationality and Adversarial Behavior in Multi-party Computation. In *CRYPTO'06*, Springer-Verlag (LNCS 4117), pages 180–197, 2006.

[32] S. Micali. Certified email with invisible post offices, 1997. Technical report; an invited presentation at the RSA 1997 conference.

[33] S. J. Ong, D. C. Parkes, A. Rosen and S. P. Vadhan. Fairness with an Honest Minority and a Rational Majority. In *6th TCC*, Springer-Verlag (LNCS 5444), pages 36–53, 2009.

[34] Rafael Pass and abhi shelat. Renegotiation-Safe Protocols. In *Innovations in Computer Science* ICS 2011.

A Dealing with Expected Round Complexity

In Theorem 4.18 we required that the round complexity of the considered protocols to be strict. In this section we show how to extend these results for dealing with expected (polynomial) round complexity. Namely, we assume that there exists a polynomial poly such that for any input, the expected number of rounds of the protocol execution on this input is bounded by $\mathsf{poly}(n)$. Moreover, we assume that both the honest party and the adversary are allowed to run in expected polynomial time. We restate and sketch the proof of the impossibility result of Theorem 4.18.

Theorem A.1 *Let f be a non-trivial two-party function. Then, for every non-negligible function $\epsilon(\cdot)$ and every $\alpha > 1/2 + \epsilon(n)$, there does not exist an α-correct protocol which is also fair (in the sense of Definition 4.4), with expected polynomial number of rounds. Not even in the simultaneous channel model.*

Proof Sketch: In this proof, we denote by $r(n) = \mathrm{E}(\mathsf{Rounds}_\pi(x_0, x_1))$ the expected number of rounds of π when invoked on (x_0, x_1). Recall that both denote the event that both parties learn their output (when both play honestly), and that a_{i+1} denotes the guess of party P_0 when P_1 aborts after sending its message in round i, whereas, b_{i+1} denotes the guess of P_1 when P_0 quits after sending its message at round i. Then, in case that the number of rounds is less than $k \cdot r(n)$ and both occurs, both $a_{kr(n)}$ and $b_{kr(n)}$ equal to the correct output. However, there may be cases where $a_{kr(n)}$ and $b_{kr(n)}$ equal to the correct output, but the execution has not terminated yet. In particular, the probability that the number of rounds is less than $k \cdot r(n)$ given that both occurs, is smaller than the probability that both parties know the correct output by round $k \cdot r(n)$. Combining the above with the analysis of Theorem 4.18, we have that,

$$\Pr[\mathsf{Rounds}_\pi(x_0, x_1) < k \cdot r(n)] \leq 1 + 2k \cdot r(n)\mu(n) - \frac{\epsilon(n)}{2}.$$

By the Markov inequality, we know that,

$$\Pr[\mathsf{Rounds}_\pi(x_0, x_1) \geq kr(n)] \leq \frac{\mathrm{E}(\mathsf{Rounds}_\pi(x_0, x_1))}{kr(n)} = \frac{1}{k}$$

and so,

$$\Pr[\mathsf{Rounds}_\pi(x_0, x_1) < kr(n)] \geq 1 - \frac{1}{k}.$$

If we choose k such that $k > 4/\epsilon(n)$, we have that

$$\Pr[\mathsf{Rounds}_\pi(x_0, x_1) < kr(n)] \geq 1 - \frac{1}{k} > 1 - \frac{\epsilon(n)}{4}$$

as in the original proof. From this point, we continue with the calculations as in the original proof and conclude that

$$r(n) > \frac{\epsilon(n)}{8k\mu(n)} > \frac{\epsilon^2(n)}{32\mu(n)}$$

in contradiction to the fact that $r(n)$ is a polynomial. ■