Cryptographic Complexity Classes and Computational Intractability Assumptions

Hemanta K. Maji[1] Manoj Prabhakaran[1] Mike Rosulek[2]

[1]Department of Computer Science, University of Illinois, Urbana-Champaign. Partially supported by NSF grant CNS 07-47027.

[2]Department of Computer Science, University of Montana

hmaji2@uiuc.edu mmp@uiuc.edu mikero@cs.umt.edu

Abstract:
Which computational complexity assumptions are inherent to cryptography? We present a broad framework to pose and investigate this question.

We first aim to understand the "cryptographic complexity" of various tasks, independent of any computational assumptions. In our framework the cryptographic tasks are modeled as multi-party computation functionalities. We consider a universally composable secure protocol for one task given access to another task as the most natural complexity reduction between the two tasks. Some of these cryptographic complexity reductions are unconditional, others are unconditionally non-existent, but the vast majority appear to depend on computational assumptions; it is this relationship with computational assumptions that we study.

In our detailed investigation of large classes of 2-party functionalities, we find that every reduction we are able to classify turns out to be unconditionally true or false, or else *equivalent* to the existence of one-way functions (OWF) or of semi-honest (equivalently, standalone-secure) oblivious transfer protocols (sh-OT). This leads us to conjecture that there are only a small finite number of distinct (i.e., black-box separable) computational assumptions that are inherent among the infinite number of different cryptographic reductions in our framework.

On moving to 3-party functionalities, there does exist a natural computational assumption, distinct from the OWF and sh-OT assumptions (namely, the existence of a key agreement protocol) that manifests itself as a cryptographic complexity reduction (in our setting, in a private communication task, the eavesdropper is considered a third party). We point out that the cryptographic complexity of functionalities with three or more parties is little understood in general, and may lead us to new distinct computational assumptions.

If indeed only a few computational complexity assumptions manifest in this framework, we propose that they are of an extraordinarily fundamental nature, since the framework contains a large variety of cryptographic tasks, and was formulated without any regard to any of the prevalent computational complexity assumptions.

Keywords: cryptographic complexity; computational complexity; assumptions; multi-party computation

1 Introduction

Cryptographic tasks, in a broad sense, specify *controlled access to information*. Here, "controlled access" stands for a possibly complex combination of access to *learning* information and access to *influencing* information in the system. A cryptographic task involves allowing access in certain senses, while disallowing access in other senses. The question of whether such a task can be realized (by software/computational means) roughly parallels the central questions in the theory of computational complexity, namely whether there exist problems which are easy in some sense (say, in NP) while hard in other senses (say, not in P).

However, cryptographic properties of interest are much more numerous and diverse compared to computational properties of interest (e.g., space, time, amount of randomness, etc. in various computational models). For instance, in Secure Function Evaluation (SFE), the kind of access to information provided by one function could be very different from that provided by another. This motivates a systematic approach to understanding the different qualitative senses of "cryptographic complexity." We layout an emerging map of cryptographic complexity, showcasing the diversity of cryptographic complexity classes (with no reference to computational complexity).

A striking picture emerges on relating these cryptographic complexity classes to questions in computational complexity. The *gap* in cryptographic complexity between two classes can be characterized by the "computational complexity assumption" that one class collapses into the other. We study several such collapses and exactly characterize these gaps in terms of "standard" computational complexity assumptions. Despite the wide variety in cryptographic complexity, the computational complexity assumptions relating the collapse of different pairs of classes appear to come from a very small class of standard assumptions. Following the influential work of Impagliazzo and Rudich [IR89] and further work by Gertner et. al [GKM$^+$00], three "distinct" computational assumptions are well recognized: the existence of one-way functions, key-agreement and (stand-alone) oblivious transfer. Incidentally these are the only assumptions that have appeared in our framework corresponding to cryptographic complexity gaps that we have studied so far. This naturally raises the question whether there are other such fundamental computational assumptions related to cryptographic complexity.[1]

[1]Indeed various other complexity assumptions (like, say the existence of enhanced trapdoor permutations) are widely used as fundamental assumptions, but it seems unlikely that they will be *equivalent* to high-level cryptographic tasks (defined independent of computational complexity aspects). One could argue that while intuitive and extremely useful, such an assumption may not be essential to cryptography.

- We propose a broad (though not necessarily exhaustive) framework for a theory of "cryptographic complexity" (i.e., complexity of cryptographic properties) based on secure multi-party computation [YAO82, GMW87]. Then, we relate the gaps in cryptographic complexity of MPC functionalities to assumptions in computational complexity. That is, we relate each pair of functionalities to the *assumption* that one functionality reduces to the other. This gives a framework for formally identifying computational complexity assumptions as "intrinsic" to cryptography.
- We examine several pairs of cryptographic complexity classes and for each such pair, find standard computational complexity assumptions *equivalent* to the assumption that one class collapses to another. Combined with other recent results, this gives a fairly complete picture for gaps among most of the "familiar" cryptographic tasks. Interestingly, the only complexity assumptions that we encounter here are the three mentioned above (existence of one-way functions, key agreement protocols and stand-alone OT protocols).
- Large tracts in the cryptographic complexity landscape remain unexplored. We do not have much understanding of the complexity gaps between classes in these regions and elsewhere, or among classes within these regions. *We conjecture that some of these gaps correspond to new fundamental complexity assumptions, that have not been discovered yet.*
  However, for the case of 2-party functionalities, we conjecture that despite the fact that there are infinitely many levels of cryptographic complexity, there are only a finite number of computational complexity assumptions corresponding to the gaps among them.

The Framework.

The seminal work of Goldreich, Micali and Wigderson [GMW87] on secure multi-party computation introduced an idealization of cryptographic tasks in terms of a trusted party, or an *ideal functionality*. An ideal functionality is an arbitrary program (possibly stateful, possibly randomized), to be executed privately by an external entity that can be trusted by all parties. This provides an extremely versatile language for capturing a great variety of kinds of controlled access to information, by simply defining various behaviors for the trusted entity — i.e., various functionalities. Further, *this idealization of the cryptographic task is separate from computational complexity considerations* (instead, the security definition involves specifying indistinguishability from this idealization, and all computational aspects are confined to this specification). The later, more refined treatments, like Canetti's Universal

Composition framework [CAN01] follow the same pattern.

To study the cryptographic content of these tasks, then, is to study the complexity of these ideal functionalities. Here, it is *not* the computational complexity of the ideal functionalities (time or space required by the ideal functionality) that is of interest. Rather, as mentioned above, we will be more interested in various qualitatively different types of functionalities. To develop a formal notion of complexity, we define *reductions* that capture the relevant cryptographic aspects. The natural notion of reduction among functionalities (without involving any computational complexity aspects) is the following. A functionality $\mathcal{F}$ is said to reduce to $\mathcal{G}$ (denoted by $\mathcal{F} \sqsubseteq \mathcal{G}$), if there exists a protocol that securely realizes $\mathcal{F}$ using access to $\mathcal{G}$. Here, the stricter the definition of secure realization used, the tighter the notion of reduction will be.[2] We shall use the strong security definition of the Universal Composition framework [CAN01] in a computationally unbounded environment. We shall consider an active adversary, but in a static corruption model. This turns out to be a robust choice which strikes a good balance: while UC security is a powerful lens to expose subtle differences in cryptographic qualities of different functionalities which disappear under weaker security definitions (like, security in a stand-alone setting), it still allows for a rich collection of reductions.[3] In Section 6 we briefly propose extensions to the basic framework.

Questions.

In this work our central concern is with the set of assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ for all pairs of functionalities $(\mathcal{F}, \mathcal{G})$. Some of these "assumptions" are unconditionally true, with the security of the reduction holding even in the statistical setting. Further, some of these assumptions are unconditionally false, no matter what computational assumption is made; the pairs $(\mathcal{F}, \mathcal{G})$ for which this happens have an explicit characterization [CKL03, PR08A]. But most of the assumptions $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are unresolved, their fate depending on computational complexity results.

1) Maximal and Minimal Assumptions.

The first question is whether there is a "maximal" or "minimal" one among these (unresolved) assumptions. That is, among these assumptions, is there:

- (a maximal assumption) an assumption $\mathcal{F}^* \sqsubseteq_{\text{PPT}} \mathcal{G}^*$ such that it implies $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ for all such pairs $(\mathcal{F}, \mathcal{G})$?
- (a minimial assumption) an assumption $\hat{\mathcal{F}} \sqsubseteq_{\text{PPT}} \hat{\mathcal{G}}$ such that $(\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G} \wedge \mathcal{F} \not\sqsubseteq_{\text{STAT}} \mathcal{G}) \implies \hat{\mathcal{F}} \sqsubseteq_{\text{PPT}} \hat{\mathcal{G}}$?

The first of these questions was recently answered in [MPR09A], where it was shown that indeed such pairs do exist: $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{COM}}$ is one such pair, where $\mathcal{F}_{\text{OT}}$ is the oblivious-transfer functionality and $\mathcal{F}_{\text{COM}}$ is the commitment functionality. A more familiar form of this assumption is that there exists a stand-alone secure oblivious-transfer protocol (sh-OT assumption). In other words, the sh-OT assumption is a maximal assumption in our framework.

However the question of the minimal assumption remains open. We conjecture that there indeed is a minimal assumption and that it in fact corresponds the existence of one-way functions. Some of the results below represent support for this conjecture. In particular we show that for several interesting pairs $(\mathcal{F}, \mathcal{G})$, $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is indeed equivalent to the existence of one-way functions (OWF).

2) Intermediate Assumptions.

Assuming OWF assumption is indeed the minimal assumption of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, one can ask if there are "intermediate" assumptions between OWF assumption and sh-OT assumption. Here, by as assumption being intermediate, we mean that there exists a black-box separation in the sense of [IR89, GKM+00], that separates it from both OWF assumption and sh-OT assumption.

Indeed, [GKM+00] shows that the existence of a secret communication protocol is an intermediate assumption. It is easy to see that their result extends to the UC secure reduction of a secret communication functionality to a public communication functionality). In our framework we find it convenient to work with functionalities which do not communicate with the adversary when all the parties are honest (called regular functionalities in [PR08A]). So this particular reduction appears only when we consider multi-party functionalities with three or more parties. For the case of 2-party (regular) functionalities, one could conjecture that all assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are in fact equivalent to either OWF assumption or sh-OT assumption (or are unconditionally true or false). This would be supported by all the results derived in this work. However, since our techniques — as well as our intuition — do not seem to apply to 2-party functionalities of a certain range of cryptographic complexity (i.e., in terminology introduced later, neither "complete," nor "passive-trivial"), we do not put forward such a conjecture. On the contrary, at this point, we consider it quite possible that for certain 2-party functionalities $\mathcal{F}$ and $\mathcal{G}$, the assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ can be black-box separated from both OWF assumption and sh-OT assumption.

[2]Studying the landscape of functionalities using a strict reduction can be compared to zooming into a map to distinguish between different elements in the map; but if one zooms in too close – uses too strict a reduction – then virtually each element appears isolated.

[3]The strong impossibility results [CAN01, CKL03, PR08A] for UC security are sometimes misinterpreted as saying that UC secure *reductions* are rare. On the contrary, many powerful reductions — including the completeness of Oblivious Transfer [KIL89] (simplified in [IPS08]) — are UC secure reductions.

Moving to 3-party functionalities already provides us one such intermediate assumption. But our understanding of multi-party functionalities with 3 or more parties is very limited. (Note that our map in Figure 1 is only for 2-party functionalities.) We consider it highly likely that several other intermediate assumptions can be discovered as gaps among functionalities with more than two parties.

Our Results.

While we consider our main contribution in this work to be the framework itself and the questions raised above, we also present several results aimed at resolving these questions. The new results here are of the form that (for various pairs of functionalities $(\mathcal{F}, \mathcal{G})$), $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies sh-OT assumption or OWF assumption. This is complemented by our recent results [MPR09A] showing several reductions (i.e., UC secure protocols) under either sh-OT assumption or OWF assumption. Together, we obtain the *equivalence* between assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ and either sh-OT assumption or OWF assumption.

Firstly, we identify a cryptographic complexity class — namely, the class of "exchange-like" functionalities — which essentially captures the ability of two parties to engage in the simultaneous exchange[4] of their inputs (drawn from finite input domains). We show that if any exchange-like functionality is to be useful in securely realizing more functionalities in the PPT setting than in the computationally unbounded, then the sh-OT assumption must hold. In particular, this implies that if a 1-bit exchange functionality can be used to UC securely realize a 2-bit exchange functionality, then in fact, the 1-bit exchange functionality can be used to UC securely realize every functionality.

We also show that several reductions among "passive-trivial" functionalities (i.e., functionalities that can be securely realized, statistically, againt passive adversaries) of differing cryptographic complexity, are equivalent to the OWF assumption.

While these results go to show that the computational complexity assumptions falling out of our framework (which encompasses a large variety of functionalities) are not as numerous as one may have *a priori* expected, we conjecture that there are "new" computational complexity assumptions that are hiding in this framework. Our main theorems and conjectures are presented in Section 3.

Related Works.

Until recently, most work in secure multi-party computation focused on the extremes of complexity; namely, classifying the functionalities that are trivially realizable (using only a communication channel) and those which are complete. Such classifications have been found for a wide variety of security models (i.e., reduction strengths) and subclasses of functionalities [GMW87, BGW88, KIL88, CCD88, CK89, BEA89, KUS89, KIL91, KIL00, CKL03, PR08C, KMQ08, IPS08].

Beimel et al. [BMM99] address a similar question as here, relating cryptographic complexity of MPC functionalities and computational complexity assumptions. But they consider computational complexity assumptions only of the form that a functionality is standalone-trivial. Restricted to a special class of SFE functionalities, they show that there is only one such assumption (other than being unconditionally true), namely the sh-OT assumption.

Recently [MPR09B] demonstrated infinitely many distinct, intermediate levels of cryptographic complexity under computationally unbounded UC security reductions. On the contrary, assuming sh-OT assumption, the state of affairs is totally different — [MPR09A] show that in this setting, every (deterministic, finite) functionality is either trivial or complete. In independent work, [?] show that under sh-OT assumption, the "common random string" functionality (or in our framework of finite functionalities, the coin-tossing functionality $\mathcal{F}_{\text{COIN}}$) is complete.

Impagliazzo and Luby [IL89] showed that several important cryptographic primitives are equivalent to the OWF assumption. Following the approach there, we also rely on the fact that a weaker primitive called distributionally one-way functions yield OWFs. In [DG03], Damgård and Groth showed that if $\mathcal{F}_{\text{COM}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{COIN}}$ then the sh-OT assumption holds. This is subsumed by our results regarding exchange-like functionalities (Theorem 1), but in fact contains one of the ideas that is used in (the simpler of) our constructions.

In considering the existence of OWF, key-agreement and sh-OT as distinct computational assumptions, we rely on the black-box separation results from [IR89, GKM$^+$00]. We refer to such separations in our *conjectures*, which predict more such distinct computational assumptions coming out of our framework. However, for the formal statement of our results (which merely show that various reductions are equivalent to one of these assumptions), such a separation is not essential.

2 Technical Preliminaries

We write $\mathcal{F} \sqsubseteq \mathcal{G}$ to denote that there exists a UC-secure protocol for $\mathcal{F}$ in the $\mathcal{G}$-hybrid world. More specifically, we use the subscripts $\sqsubseteq_{\text{PPT}}$ and $\sqsubseteq_{\text{STAT}}$ to indicate that the protocol is secure against PPT adversaries and computationally unbounded adversaries, respectively.

Definition 1. *A functionality is a* secure function evaluation (SFE) *functionality if it waits for inputs* $x \in X$

[4]Here simultaneity only refers to the input independence (Alice should not be able to choose her input based on Bob's input, and vice versa), and not to any notion of fairness.

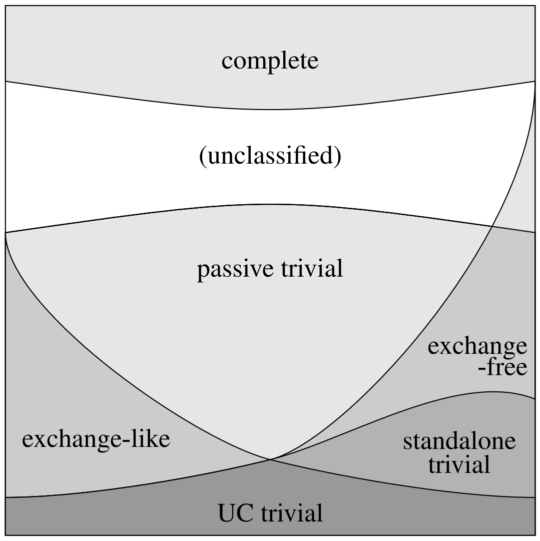

Figure 1: A map of various cryptographic complexity classes (of 2-party SSFE functionalities)

from Alice and $y \in Y$ from Bob (for some finite sets X and Y) and then sends $f_A(x, y)$ to Alice and $f_B(x, y)$ to Bob. If either party is corrupt, the output for that party is first delivered; if it allows, the output to the other party is also delivered.

We say that the functionality is symmetric SFE (SSFE) *if $f_A = f_B$.*

Note that if neither party is corrupt, the functionality does not interact with the adversary. (Such functionalities were called regular functionalities in [PR08A].)

Several specific functionalities have important roles in our results. $\mathcal{F}_{\text{OT}}$, $\mathcal{F}_{\text{COM}}$, and $\mathcal{F}_{\text{COIN}}$ denote the standard 1-out-of-2 oblivious transfer (OT) functionality, bit-commitment functionality, and coin-tossing functionality, respectively. $\mathcal{F}_{\text{CC}}$ denotes a bit "cut-and-choose" functionality, which is an SSFE whose function table is $\boxed{\begin{smallmatrix} 0 & 2 \\ 1 & 2 \end{smallmatrix}}$. That is, Alice provides a bit, and Bob can choose whether or not to learn it, and Alice learns Bob's choice.

We denote by $\mathcal{F}^{i \times j}_{\text{EXCH}}$ the SSFE functionality in which Alice gives input $x \in [i]$, Bob gives input $y \in [j]$, and both parties learn (x, y). We call functionalities of this form *exchange functions*. The cryptographic sophistication of an exchange function is that it enforces inputs to be chosen independently (though we make no requirement for fairness in learning the output). The special case $\mathcal{F}^{2 \times 2}_{\text{EXCH}}$ is isomorphic to the boolean-XOR SSFE functionality.

Definition 2. *We say that two SFE functionalities are* isomorphic *if one can be obtained from the other by repeatedly adding/removing duplicate or redundant inputs, permuting a party's inputs, relabeling a party's outputs for one of its inputs, and reversing the roles of Alice and Bob.*

By redundant input, we mean an input x that always induces the same output for the other party as another input x', but where the output when using x' always uniquely determines the output when using x.

2.1 Passive Security, Previous Work

We will find useful some of the well-known results from the theory of passive security:

Definition 3 (Decomposable [KUS89, BEA89]). *An SSFE functionality $\mathcal{F} : X \times Y \to Z$ is* row decomposable *if there exists a partition $X = X_1 \cup \cdots \cup X_k$ ($X_i \neq \emptyset$), $k \geq 2$, such that the following hold for all $i \leq k$:*

- *for all $y \in Y$, $x \in X_i$, $x' \in (X \setminus X_i)$, we have $\mathcal{F}(x, y) \neq \mathcal{F}(x'y)$; and*
- *$\mathcal{F}\big|_{X_i \times Y}$ is either a constant function or column decomposable, where $\mathcal{F}\big|_{X_i \times Y}$ denotes the restriction of $\mathcal{F}$ to the domain $X_i \times Y$.*

We define being column decomposable *symmetrically with respect to X and Y. We say that $\mathcal{F}$ is simply* decomposable *if it is either constant, row decomposable, or column decomposable.*

We define a *decomposition tree* of f to be the natural representation of the decomposition of f, where each node of the tree is associated with some subdomain $X' \times Y' \subseteq X \times Y$, and the children of a node correspond to the partitioning of the domain induced by the row or column decomposition step.

If $\mathcal{F}$ is decomposable, then a *canonical protocol* for f is a deterministic protocol defined inductively as follows [KUS89]:

- If $\mathcal{F}$ is a constant function, both parties output the output value of $\mathcal{F}$, without interaction.
- If $\mathcal{F}$ is row decomposable as $X = X_1 \cup \cdots \cup X_k$, then party 1 announces the unique i such that its input $x \in X_i$. Then both parties run a canonical protocol for $\mathcal{F}\big|_{X_i \times Y}$.
- If $\mathcal{F}$ is column decomposable as $Y = Y_1 \cup \cdots \cup Y_k$, then party 2 announces the unique i such that its input $y \in Y_i$. Then both parties run a canonical protocol for $\mathcal{F}\big|_{X \times Y_i}$.

Note that the canonical protocol could alternately be defined as a traversal from root to leaf of $\mathcal{F}$'s decomposition tree. It is a simple exercise to see that a canonical protocol is a perfectly secure protocol for $\mathcal{F}$ against passive adversaries.

3 The Cryptographic Complexity Landscape

In this section we present our results connecting cryptographic complexity classes and computational assumption. Before doing that, first we need to describe various cryptographic complexity classes, defined independent of any computational complexity concerns. (Some of these classes were identified in [MPR09B].)

Then, in Section 3.2 we list the connections with computational assumptions that we have proven, and that we conjecture.

3.1 Some Cryptographic Complexity Classes

In this work we consider only finite functionalities — namely, functionalities which are finite state (possibly randomized) machines, with finite input and output alphabets. (See Section 6 for a brief discussion.) However, the total space of functionalities – including reactive and randomized functionalities – is still far from being well-understood. As such, we shall go about exploring two-party deterministic, non-reactive functionalities for most part, and describe the extensions we have for more general functionalities, when possible. (However, we do closely consider certain randomized functionalities – like coin-flipping, and certain reactive functionalities – like commitment.)

In this section we identify a few major classes of two-party symmetric secure function evaluation (SSFE) functionalities. SSFE functionalities, starting with the original *Millionaire's Problem* proposed by Yao [YAO82], are perhaps the most well-studied class of two-party functionalities. We present our results on computational assumptions by relating them to these cryptographic complexity classes. (Further details of these classes, and examples, are given in Appendix A.)

The classes listed below, except the first two, are "downward closed" with respect to $\sqsubseteq_{\text{STAT}}$. That is, if $\mathcal{G}$ is a functionality in a class and $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ then $\mathcal{F}$ also falls in the same class. (The first one is "upward closed.") These classes are graphically represented in Figure 1.

Complete Functionalities. These are SSFE functionalities that are statistically "complete." That is, for all $\mathcal{G}$ in this class, and all functionalities $\mathcal{F}$, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$. Kilian [KIL91] gave a combinatorial characterization for these functionalities, as the evaluation of functions containing an "OR-minor" (also called embedded OR).

Unclassified Functionalities. Among incomplete SSFE functionalities, we leave a set of functionalities as "unclassified." These are functionalities which are neither complete, nor passive-trivial (see next class).

Passive-Trivial Functionalities. These are SSFE functionalities securely realizable against a passive (a.k.a honest-but-curious, or semi-honest) adversary in a computationally unbounded environment. For SSFE functionalities, such functions have an explicit combinatorial characterization, namely that they are evaluations of what are called "decomposable" functions [KUS89, BEA89, MPR09B, KMQR09]. An alternate characterization for this class was given in [MPR09B]: $\mathcal{F}$ is passive-trivial iff $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}_{\text{COM}}$, where $\mathcal{F}_{\text{COM}}$ is the commitment functionality.

Exchange-Like Functionalities. This is a important sub-class of passive-trivial functionalities defined to be the class of functionalities that reduce to a simultaneous exchange functionality. That is, $\mathcal{F}$ is in this class iff there exists an integer i such that $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}^{i,i}_{\text{EXCH}}$.

Exchange-Free Functionalities. This is the class of SSFE functionalities to which no non-trivial exchange functionality can be statistically reduced. That is, a functionality $\mathcal{G}$ is in this class iff $\mathcal{F}^{2,2}_{\text{EXCH}} \not\sqsubseteq_{\text{STAT}} \mathcal{G}$.

A useful combinatorial property of exchange-free SSFE functionalities which are passive-trivial is that they are evaluations of functions whose function tables are "uniquely decomposable," as defined in [MPR09B]. (However, being uniquely decomposable does not necessarily mean that the SFE is exchange-free.)

Standalone-Trivial Functionalities. These are SSFE functionalities securely realizable against a stand-alone adversary in a computationally unbounded environment. SSFE functionalities in this class were combinatorially characterized in [KMQR09, MPR09B]. They are passive-trivial and it turns out they are all exchange-free functionalities.

An important example of an SSFE in this class (and indeed the simplest non-trivial one) is $\mathcal{F}_{\text{CC}}$– the evaluation of the "cut-and-choose" function $\boxed{\begin{smallmatrix}0 & 2\\ 1 & 2\end{smallmatrix}}$ – in which Bob can choose to learn Alice's input or not (and Alice learns Bob's choice).

Trivial Functionalities. These are SSFE functionalities securely realizable against a general adversary in a computationally unbounded environment, using protocols which only rely on (private) communication channels.[5] These functionalities are isomorphic to evaluation of functions with a one-dimensional function-table (that is, one of the parties has a fixed input). This also corresponds to the intersection of the classes exchange-like and exchange-free.

3.2 Relating Cryptographic Complexity Gaps to Computational Complexity Assumptions

Recall that our interest is in "assumptions" of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are arbitrary, finite functionalities. Among two-party functionalities, all the reductions we have been able to classify fall into one of four kinds (this leaves out several reductions that we have not been able to classify):

1. Reductions that are unconditionally true (even relative to arbitrary computational oracles).
2. Reductions equivalent to the OWF assumption.
3. Reductions equivalent to the $\mathsf{sh\text{-}OT}$ assumption.
4. Unconditionally false reductions.

If $\mathcal{F}$ is non-trivial and $\mathcal{G}$ is trivial (as defined in Section 3.1), then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is unconditionally false, no matter what computational complexity results may hold. (This is a consequence of the impossibility results in [CKL03, PR08A].) Recent results in [MPR09A]

[5]Recall that in our model functionalities — including communication channels — interact only with the parties. A channel with an eavesdropper is modeled as a 3-party functionality.

show that all other assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are *implied by* the $\mathsf{sh\text{-}OT}$ assumption. Thus the remaining question is to find how strong an assumption is implied by $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ (for pairs for which it is not unconditionally false). Note that the space of finite functionalities is infinite, and *a priori* one might expect a large number of, if not infinitely many, assumptions that can be black-box separated from each other. Indeed, such a phenomenon is not without precedence: in [GKM$^+$00] for instance, an infinite hierarchy of complexity assumptions related to cryptographic protocols is derived. However, the assumptions in such an infinite hierarchy tend to have quantitative differences, and may not relate to conceptual differences. On the other hand, the above assumptions (the OWF assumption and the $\mathsf{sh\text{-}OT}$ assumption) are conceptually different.

In our framework, we hope to discover "all" conceptually different computational assumptions related to cryptographic complexity.

Conjecture 1 (Quantization of Computational Assumptions.)**.** *There are only finitely many assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are two-party functionalities, that can be black-box separated from each other.*

We believe that there are several such new assumptions to be discovered. But we conjecture that we need to look beyond the more familiar functionalities to discover them:

Conjecture 2. *Assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are two-party SSFE functionalities that do not belong to the class of unclassified functionalities (see Section 3.1), fall into one of the four classes above.*

We present significant progress towards proving the above conjecture. Note that we are looking to prove that the assumptions $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are *equivalent* to the OWF assumption or the $\mathsf{sh\text{-}OT}$ assumption, unless they can be shown to be unconditionally true or false. One part of showing the equivalence is to show that the OWF assumption or the $\mathsf{sh\text{-}OT}$ assumption yields such reductions, presumably by giving explicit protocols. This was carried out in [MPR09A].

Proposition 1 (Based on results in [MPR09A].)**.** *The assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is:*

- *unconditionally true (i.e., $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$) if $\mathcal{G}$ is complete or if $\mathcal{F}$ is trivial;*
- *unconditionally false, if $\mathcal{G}$ is trivial and $\mathcal{F}$ is non-trivial;*
- *equivalent to the $\mathsf{sh\text{-}OT}$ assumption if $\mathcal{G}$ is complete and $\mathcal{F}$ is passive-trivial;*
- *implied by the $\mathsf{sh\text{-}OT}$ assumption if $\mathcal{G}$ is not trivial;*
- *implied by the OWF assumption if $\mathcal{G}$ is not exchange-like and $\mathcal{F}$ is passive-trivial.*

In this work, the focus is on showing the converse, that for various pairs $(\mathcal{F}, \mathcal{G})$, $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies the OWF assumption or the $\mathsf{sh\text{-}OT}$ assumption.

Reductions equivalent to the $\mathsf{sh\text{-}OT}$ assumption

We completely characterize pairs $(\mathcal{F}, \mathcal{G})$ among SSFE functionalities that are not unclassified, for which $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the $\mathsf{sh\text{-}OT}$ assumption.

Theorem 1. *Let $\mathcal{G}$ be a non-trivial exchange-like SSFE functionality. For any SSFE functionality $\mathcal{F}$ such that $\mathcal{F} \not\sqsubseteq_{\text{STAT}} \mathcal{G}$, we have $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G} \iff$ $\mathsf{sh\text{-}OT}$ assumption.*

Among SSFE functionalities that are not unclassified, the only pairs which are equivalent to the $\mathsf{sh\text{-}OT}$ assumption are the $(\mathcal{F}, \mathcal{G})$ in the above theorem, and those in Proposition 1 (item 3). For each of the other pairs, either the assumption is unconditionally true or unconditionally false, or is implied by the OWF assumption (by Proposition 1, item 5).

As a first step in proving the above result we consider the question of whether exchanging a single-bit (or equivalently the $\mathcal{F}_{\text{XOR}}$ functionality) reduces to the coin-tossing functionality $\mathcal{F}_{\text{COIN}}$, and show that $\mathcal{F}_{\text{XOR}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{COIN}}$ is equivalent to the $\mathsf{sh\text{-}OT}$ assumption. We construct an $\mathsf{sh\text{-}OT}$ protocol from a protocol $\pi^{\mathcal{F}_{\text{COIN}}}$ for $\mathcal{F}_{\text{XOR}}$, in which the sender and receiver will run π, with the receiver simulating $\mathcal{F}_{\text{COIN}}$; the receiver will either run the simulator for π or the actual protocol π itself, depending on whether or not it wants to learn the bit being input by the sender. The protocol π will be truncated at a random point so that there is a higher chance that the simulator would have extracted but the actual party in the protocol would not have learned the bit yet. This gives a weak oblivious transfer which can be amplified to a full-fledged $\mathsf{sh\text{-}OT}$ protocol.

A similar (though technically more involved) argument works for arbitrary $\mathcal{F}$ and exchange-like $\mathcal{G}$ (instead of $\mathcal{F}_{\text{XOR}}$ and $\mathcal{F}_{\text{COIN}}$). Further, Theorem 1 extends to the case of *reactive* exchange-like functionalities, as well. Analysis of arbitrary reactive functionalities was introduced in [MPR09A], and we build on the analysis there, to characterize exchange-like reactive functionalities.

Reductions equivalent to the OWF assumption

Given Proposition 1 and Theorem 1, Conjecture 2 is equivalent to the following conjecture.

Conjecture 3. *For any two passive-trivial SSFE functionalities $\mathcal{F}$ and $\mathcal{G}$, either $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, or $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies the OWF assumption.*

That is, we would like to prove that the OWF assumption is indeed minimal for conditional reductions, at least among passive-trivial functionalities. While this may sound obvious, proving such a conjecture turns out not to be easy. Essentially, assuming that the OWF assumption does not hold, one must show attacks on any protocol purportedly carrying out such a reduction. We obtain the following results, to partially confirm the above conjecture.

Theorem 2. *For every standalone-trivial functionality $\mathcal{G}$, and every passive-trivial functionality $\mathcal{F}$ that is not standalone-trivial, $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

Theorem 3. *For every passive-trivial SFE functionality $\mathcal{G}$, there are infinitely many standalone-trivial SFE functionalities $\mathcal{F}$ such that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

3.3 Beyond 2-Party Functionalities

The set of two-party functionalities, as we considered above, is quite rich, but still omits some very important and familiar kinds of cryptographic tasks. In particular, the classical task of secret communication, is modeled as a functionality with three parties, Alice, Bob and Eve, and is not captured by any of the two-party functionalities.[6] Let us denote the 3-party private channel functionality (in which the third party learns only that the channel was invoked, when the first party sends a message to the second party, by $\mathcal{F}_{\text{PVT}}$. In contrast, let $\mathcal{F}_{\text{PUB}}$ be the a 3-party public-channel functionality, in which the message from the first party is received by both the other parties (though if the first party is corrupt, it is allowed to send different messages to the two others). The assumption in question is $\mathcal{F}_{\text{PVT}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{PUB}}$.

A key-agreement protocol, as considered in [IR89, GKM$^+$00], yields such a reduction. This assumption, corresponding to secrecy against third-party eavesdroppers, seems to be of a different flavor than any assumption arising out of cryptographic complexity of 2-party functionalities (wherein there is no external adversary).

Conjecture 4. *For any pair of two-party functionalities $\mathcal{F}, \mathcal{G}$, the assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ can be black-box separated (a la [IR89, GKM$^+$00]) from the assumption $\mathcal{F}_{\text{PVT}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{PUB}}$.*

A first step to studying the assumptions arising of 3-party functionalities would be to understand various cryptographic complexity classes (based on statistical reductions). However, even the class of trivial SFE functions is not well-understood in this case. [PR08B, Appendix. B] includes a few "trivial" 3-party functionalities, which are securely realizable using protocols that involve more than a single message. These functionalities are either "aggregated" functionalities (in which the first two parties have an input, and only the third party receives an output) or "disseminated" (in which the third party has an input and the first two parties receive (possibly different) outputs). The examples of realizable functionalities given in [PR08B] are "aggregated XOR," "aggregated OR," "disseminated XOR" and "disseminated OR." (Disseminated XOR is simply a broadcast, in which the output of the two parties receiving outputs is guaranteed to be x and y such that $x = y$. Even more interestingly, in disseminated OR, the output for the two parties are two bits x and y such that it is guaranteed that $x \vee y = 1$.) These protocols are statistically secure in the $\mathcal{F}_{\text{PVT}}$-hybrid model.

Given the variety of cryptographic functionalities that exist in the 3-party scenario, we conjecture that there is at least one "undiscovered assumption" corresponding to a reduction among 3-party functionalities.

Conjecture 5. *There exist 3-party functionalities $\mathcal{F}$, $\mathcal{G}$, such that the assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ can be black-box separated (a la [IR89, GKM$^+$00]) from* OWF *assumption,* $\mathsf{sh\text{-}OT}$ *assumption and $\mathcal{F}_{\text{PVT}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{PUB}}$.*

4 Reductions Equivalent to the OWF Assumption

Our results in this section build on the technique in [MPR09B] that was used to derive the following separation in cryptographic complexity.

Theorem 4 ([MPR09B])**.** *Let $\mathcal{F}$ and $\mathcal{G}$ be SSFE functionalities. If $\mathcal{F}$ has unique decomposition depth n and $\mathcal{G}$ has decomposition depth $m < n$, then $\mathcal{F} \not\sqsubseteq_{\text{STAT}} \mathcal{G}$.*

In [MPR09B], Theorem 4 is proven by attacking any purported protocol π for $\mathcal{F}$ in the $\mathcal{G}$-hybrid world.

First, they show (for plain protocols, not in any hybrid world) that for every adversary $\mathcal{A}$ that attacks the canonical protocol for $\mathcal{F}$, there is a corresponding adversary $\mathcal{A}'$ that attacks π, achieving the same effect in all environments. (Indeed, any functionality whose decomposition depth is at least 2 has a simple attack against its canonical protocol that violates security in the UC sense.) Intuitively, the protocol π must reveal information in the same order as the canonical protocol. More formally, at every point during the canonical protocol (say, a partial transcript t), there is a corresponding "frontier" in π — a maximal set of partial transcripts of π. If two inputs both induce transcript t in the canonical protocol (recall that it is a deterministic protocol), then they also induce statistically indistinguishable distributions on partial transcripts at the frontier. But if the two inputs do not both induce transcript t in the canonical protocol, then at the frontier they induce distributions on partial transcripts that have statistical distance almost 1. Then the adversary $\mathcal{A}'$ runs the protocol π honestly,

[6]Recall that our functionalities do not communicate wiith the adversary when all parties are honest. This convention requires modeling corrupt parties explicitly, in the protocol and in the functionality. Hence secret communication corresponds to a 3-party functionality.

except for occasionally "swapping" its effective input at one of these frontiers. The properties of the frontiers assure that such a swap will only negligibly affect the outcome of the interaction.

Next, to attack a protocol π in the $\mathcal{G}$-hybrid world, they imagine a plain protocol $\widehat{\pi}$ which is π composed with the canonical protocol for $\mathcal{G}$. The plain protocol $\widehat{\pi}$ has frontiers for each step of the canonical protocol (equivalently, step of the decomposition). In our setting, there are more frontiers in $\widehat{\pi}$ than there are rounds in the canonical protocol for $\mathcal{G}$, so not all the frontiers can be contained entirely within the $\mathcal{G}$-subprotocols. Thus an adversary attacking π can behave honestly in all interactions with the ideal $\mathcal{G}$, and still encounter a frontier at which to "swap" its effective input (i.e., outside of the $\mathcal{G}$-subprotocols in $\widehat{\pi}$). Indeed, there is an attack against $\mathcal{F}$ in which an adversary need only encounter one such frontier, so the protocol π is not secure.

Leveraging one-way functions.

While these frontier-based attacks from [MPR09B] are formulated for computationally unbounded adversaries, we show below that they can in fact be carried out under the assumption that one-way functions *do not exist*. In other words, that if a reduction exists between particular functions, then the OWF assumption is true.

These frontier-based attacks require unbounded computation because computing the frontier involves computing global statistical properties about the protocol — namely, the probability that the protocol assigns to various partial transcripts on different inputs. The attacks are otherwise effecient, so given access to an oracle that can compute these probabilities, the attack can be easily effected. In fact, these quantities need not be computed exactly for the attacks to violate security. Thus we will describe how to compute the appropriate quantities given that OWFs do not exist.

In [IL89], it is shown that the OWF assumption is implied by the much weaker assumption that *distributionally one-way* functions exist. Thus if OWFs do not exist, then no function is distributionally one-way: for every efficient function f and polynomial p, there is an efficient algorithm that on input y samples close to uniformly (within $1/p$ statistical difference) from the set $f^{-1}(y)$. We define a function related to the given protocol, and use the ability to sample its preimage to obtain a good estimate of the desired probabilities.

Theorem 5. *If $\mathcal{F}$ has unique decomposition depth n and $\mathcal{G}$ is non-trivial with unique decomposition depth $m < n$, then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

Proof. First, if $\mathcal{G}$ is uniquely decomposable, then $\mathcal{F}_{\text{COM}} \sqsubseteq \mathcal{G}$ under the OWF assumption, by the argument in [MPR09A]. Then, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}_{\text{COM}}$ since $\mathcal{F}$ is passive-trivial [MPR09B]. The non-trivial direction is to show that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies OWF assumption.

As described above, the attack against a protocol π for $\mathcal{F}$ in the $\mathcal{G}$-hybrid world is based on frontiers in the protocol. For a partial transcript u and inputs x for Alice and y for Bob, the probability that the protocol generates u as the prefix of its transcript can be expressed as $\alpha(u, x)\beta(u, y)$, where each of the two terms depends on only one party's input.

The frontiers used in the attack are then all defined in terms of the following quantity:

$$\eta(u, x_0, x_1) = \frac{|\alpha(u, x_0) - \alpha(u, x_1)|}{\alpha(u, x_0) + \alpha(u, x_1)}$$

or the symmetric quantity with respect to the roles of Alice & Bob. Intuitively, $\eta(u, x_0, x_1)$ measures how correlated the transcript u is to Alice's input being x_0 versus x_1. In fact, the entire frontier-based attack can be carried out in polynomial time given an oracle that answers questions of the form "Is $\eta(u, x_0, x_1) \geq 1 - \nu(k)$?", where ν is a certain negligible function in the security parameter. If instead the oracle can answer questions of this form where $\nu(k) = 1/k^c$ for a chosen constant c, then the adversary's attack may fail with at most an extra 1/poly factor. All the attacks from [MPR09B] demonstrate that the real and ideal worlds can be distinguished with constant bias, so they can indeed tolerate this additional 1/poly slack factor. Thus it suffices to show how to implement such an oracle.

We compute $\eta(u, x_0, x_1)$ as follows: First, Consider the function $f(x, r_A, y, r_B, i) = (\tau, x)$, where τ is the first i bits of the transcript produced by the protocol when executed honestly on inputs (x, y), where r_A and r_B are the random tapes of Alice and Bob, respectively. We use the guarantee of no distributionally one-way functions to sample from $f^{-1}(u, x_0)$ and $f^{-1}(u, x_1)$. If both preimages are empty, then the protocol never generates u as a partial transcript on inputs x_0 or x_1. If only one is empty, then $\eta(u, x_0, x_1) = 1$.

Otherwise, assume u is indeed a possible partial transcript for both x_0 and x_1 (i.e., the protocol assigns positive probability to u when Alice has inputs x_0 or x_1). Our previous sampling of f^{-1} has yielded an input y^* such that u is a possible partial transcript when executing π on inputs (x_0, y^*). Thus u is also a possible partial transcript on inputs (x_1, y^*). Now define:

$$g(x, r_A, y, r_B, i) = \begin{cases} (\tau, y) & \text{if } x \in \{x_0, x_1\} \\ \bot & \text{otherwise} \end{cases}$$

We now sample n times from $g^{-1}(u, y^*)$. Let n_i be the number of times the sampled preimage included x_i as the first component. Then $|n_0 - n_1|/n$ is an estimate of $\eta(u, x_0, x_1)$. By setting n to be a sufficiently large polynomial in the security parameter, we can ensure that the estimate is within an additive factor $1/k^c$ of the actual value, with high probability. □

Theorem 6. *If $\mathcal{F}$ is passive-trivial but not standalone-trivial and $\mathcal{G}$ is standalone-trivial but not UC-trivial, then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

Proof. The fact that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ under the OWF assumption is by the same argument as in the previous proof.

For the other direction, suppose π is a secure protocol for $\mathcal{F}$ in the $\mathcal{G}$-hybrid world. Standalone secure protocols *for SFE* functionalities are closed under composition. Thus we have a standalone-secure protocol π' for $\mathcal{F}$ without any trusted party.

Being passive-trivial, $\mathcal{F}$ is surely decomposable, and we consider two cases. When $\mathcal{F}$ is uniquely decomposable, then [MPR09B] showed that in the *unbounded* setting, for every adversary $\mathcal{A}$ attacking the canonical protocol, there is an adversary $\mathcal{A}'$ attacking π' such that no environment can distinguish between the two interactions. When $\mathcal{F}$ is uniquely decomposable but not standalone-trivial, there is a simple attack against the canonical protocol for $\mathcal{F}$ that violates standalone security with constant probability. Thus translating this attack into an efficient (assuming that OWF assumption is false) attack on π' using the techniques described in the previous proof, we see that π' is not standalone-secure; a contradiction.

On the other hand, if $\mathcal{F}$ is not uniquely decomposable, then $\mathcal{F}_{\text{XOR}} \sqsubseteq_{\text{STAT}} \mathcal{F}$ via a simple protocol. As such, by composing several protocols, we obtain a standalone-secure protocol π for $\mathcal{F}_{\text{XOR}}$. Consider an interaction using π in which the honest party choses an input at random. We describe an attack that can be carried out assuming that the OWF assumption is false, which biases the honest party's output towards 0 by a noticeable amount:

At each partial transcript u, consider $\eta(u, 0, 1)$ (which measures the transcript's bias towards Alice's input 0 or 1, defined in the previous proof) At the beginning of the protocol, the value of this function is 0, and at the end of the protocol, it is negligibly close to 1 with overwhelming probability since the protocol results in Bob learning Alice's input.

Similarly, define $\eta'(u, 0, 1)$ as a transcript's bias towards Bob's input. By symmetry, with probability at least 1/2, the partial transcript achieves $\eta(u, 0, 1) > 1/2$ before it achieves $\eta'(u, 0, 1) > 1/2$. Thus an attack for Bob is to discover via the sampling procedure described above the first point at which $\eta(u, 0, 1) > 1/2$ but $\eta'(u, 0, 1) \leq 1/2$. At that point, Bob switches his input to match Alice's, in order to bias the output towards 0. Bob reaches such a point with probability at least 1/2, Since $\eta'(u, 0, 1) \leq 1/2$, the correctness of the protocol implies that Bob's output will be 0 with overwhelming probability. Thus this attack successfully biases the output towards 0 with bias 1/4 minus some inverse polynomial in the security parameter. □

5 Reductions Equivalent to the $\mathsf{sh\text{-}OT}$ Assumption

Definition 4. *Let $\mathcal{F}$ be an SFE functionality. We say that $\mathcal{F}$ is* exchange-like *if $\mathcal{F} = \mathcal{F}_{\text{EXCH}}^{i \times j}$ for some i, j.*

Lemma 1 ([MPR09A])**.** *If $\mathcal{F}$ is not exchange-like, then either $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{STAT}} \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq_{\text{STAT}} \mathcal{F}$.*

The proof is a simple combinatorial characterization. If $\mathcal{F}$ is not exchange-like, then it contains one of two kinds of 2×2 minors. One of these minors yields an unconditional $\mathcal{F}_{\text{OT}}$ protocol, due to a result of [KMQ08]. The other kind of minor yields an elementary protocol for $\mathcal{F}_{\text{CC}}$.

Our main classification involving exchange-like functionalities is the following:

Theorem 7. *If $\mathcal{G}$ is exchange-like and non-trivial, then either $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, or $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* $\mathsf{sh\text{-}OT}$ *assumption.*

Proof. From [MPR09A], we have that $\mathcal{G}$ is $\sqsubseteq_{\text{PPT}}$-complete under the $\mathsf{sh\text{-}OT}$ assumption, since it is non-trivial. Thus $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ under the $\mathsf{sh\text{-}OT}$ assumption.

For the other direction, we break the proof into two parts, depending on the status of $\mathcal{F}$. These are carried out in the following two lemmas. □

Lemma 2. *If $\mathcal{F}$ is not exchange-like, and $\mathcal{G}$ is exchange-like and non-trivial, then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies the* $\mathsf{sh\text{-}OT}$ *assumption.*

Proof. Given that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, we directly construct a passive secure protocol for $\mathcal{F}_{\text{OT}}$. From Lemma 1, we have that either $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{STAT}} \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq_{\text{STAT}} \mathcal{F}$. Thus either $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{PPT}} \mathcal{G}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq_{\text{PPT}} \mathcal{G}$ by the universal composition theorem.

In the first case, $\mathcal{F}_{\text{OT}}$ has the property that any UC-secure protocol for $\mathcal{F}_{\text{OT}}$ (even in a hybrid world) is also itself a semi-honest-secure protocol [PR08A]. $\mathcal{G}$ also has a semi-honest-secure protocol (namely, its canonical protocol since it is decomposable). Composing these two protocols yields a semi-honest (plain) protocol for $\mathcal{F}_{\text{OT}}$.

In the other case, suppose π is the secure protocol for $\mathcal{F}_{\text{CC}}$ in the $\mathcal{G}$-hybrid world. Recall that $\mathcal{F}_{\text{CC}}$ has a function table $\boxed{\begin{smallmatrix} 0 & 2 \\ 1 & 2 \end{smallmatrix}}$, which we interpret as Alice sending a bit (top row or bottom row), Bob choosing whether or not to recieve it (left column or right column), and Alice learning Bob's choice (whether or not the output was 2). We directly use π to construct a semi-honest $\mathcal{F}_{\text{OT}}$ protocol as follows, with Alice acting as the OT sender (with inputs x_0, x_1) and Bob the receiver (with input b):

- The parties instantiate two parallel instances of π, with Alice acting as the sender. Since there is no access to an external $\mathcal{G}$, Bob will simulate Alice's interface with instances of $\mathcal{G}$— that is, Alice will send her $\mathcal{G}$ inputs directly to Bob, and he will give simulated responses from instances of $\mathcal{G}$. Alice sends bit x_0 in the first instance, and x_1 in the second instance, running the protocol honestly.

- In protocol instance $(1-b)$, Bob carries out the simulation of $\mathcal{G}$-instances and the π protocol completely honestly. He runs the π protocol on the input that does not reveal Alice's input.
- In protocol instance b, Bob honestly runs the UC simulator for π, treating Alice as the adversary (including simulating Alice's interface with $\mathcal{G}$-instances). At some point, the simulator extracts Alice's bit x_b to send to $\mathcal{F}_{\text{CC}}$. Bob continues running the simulator as if $\mathcal{F}_{\text{CC}}$ responded with output 2. When the interaction completes, Bob outputs x_b.

By the UC security of π, Alice's view is computationally independent of b (i.e., she cannot distinguish an interaction with π's simulator from an interaction in which the receiver and $\mathcal{G}$ are honest). Bob correctly learns x_b, and we must argue that he has no advantage guessing x_{1-b}. If all $\mathcal{G}$-instances were external to the $(1-b)$ interaction as ideal functionalities, then the security of π would imply that Bob has no advantage in guessing x_{1-b} after running the protocol with the input that does not reveal Alice's bit. Being an exchange function, $\mathcal{G}$ has the property that Bob always learns all of Alice's inputs. Thus Alice can send her $\mathcal{G}$-inputs directly to Bob, without any affect on the security of the protocol. This is exactly what happens in the $(1-b)$ interaction. □

For the case where $\mathcal{F}$ is exchange-like, we completely characterize when $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the sh-OT assumption.

Lemma 3. *Let $\mathcal{F}$ and $\mathcal{G}$ be exchange-like, so without loss of generality, $\mathcal{F} = \mathcal{F}_{\text{EXCH}}^{i\times j}$ and $\mathcal{G} = \mathcal{F}_{\text{EXCH}}^{i'\times j'}$. Then if $i \le i'$ and $j \le j'$, or if $i \le j'$ and $j \le i'$, then $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$. Otherwise, $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is implies the* sh-OT *assumption.*

Proof. The protocol to show $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ is elementary. To perform an $i \times j$ exchange using $\mathcal{G}$, simply place $\mathcal{G}$ in the appropriate send inputs directly to $\mathcal{G}$ (with Alice and Bob exchanged if necessary). Each party aborts if the other party provided an input to the $i' \times j'$ exchange which was out of bounds for an $i \times j$ exchange. The security of this protocol is straight-forward.

We sketch here the main ideas behind proving the other direction. The full proof is given in the appendix. For simplicity, suppose that $\mathcal{F} = \mathcal{F}_{\text{EXCH}}^{i\times i}$ and $\mathcal{G} = \mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$.

Suppose we have a protocol π demonstrating $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$. The role of the simulator for π is to first extract the input of a corrupt party, send it to $\mathcal{F}$ in the ideal world, and then continues to simulate π consistently given the output from $\mathcal{F}$.

Again for simplicity, suppose that the simulator for a *passively* corrupt Alice always extracts during round r_A.[7] Then through $r_A - 1$ rounds of the simulation, Alice's view is independent of Bob's input. If Bob's input is random (uniform in $[i]$), then after round r_A, Alice cannot guess Bob's input with probability greater than $\zeta = (i-1)/i$, since there are only $i-1$ possible responses from the simulated $\mathcal{G}$ that the simulator can give to complete the round. By the soundness of the simulation, an honest Alice cannot predict Bob's input with probability greater than $\zeta + \text{negl}(k)$ after r_A rounds of an *honest* interaction with Bob. Similarly, if the simulator for a passively corrupt Bob always extracts during round r_B, then an honest Bob cannot predict Alice's random input with probability greater than $\zeta + \text{negl}(k)$ after r_B rounds of an honest interaction with Alice.

By symmetry, suppose that $r_A \le r_B$. Then a semi-honest protocol for a weak variant of OT is as follows:

- Alice chooses two random elements $x_0, x_1 \in [i]$ and runs two instances of the protocol π with these respective inputs, for r_A rounds.
- Bob's input is a choice bit $b \in \{0,1\}$, and in the bth interaction with π, Bob runs the simulator for π against Alice (including simulating her interface with instances of $\mathcal{G}$). In the $(1-b)$ interaction, Bob runs the π protocol honestly on a fixed input, and also honestly simulates all instances of $\mathcal{G}$. After r_A rounds, the b-interaction successfully extracts x_b, which Bob outputs.

By the security of the protocol π, Alice cannot distinguish between the b and $(1-b)$ instances. In the $(1-b)$ instance, Bob runs the protocol honestly against Alice for $r_A \le r_B$ rounds, and as such, cannot predict x_{1-b} with probability greater than $\zeta + \text{negl}(k)$. Using a standard amplification technique (Appendix D), we can obtain a full-fledged OT protocol in which Bob has no advantage in predicting x_{1-b}.

The main proof is more involved in several ways. First, the case where the dimensions of $\mathcal{F}$ and $\mathcal{G}$ are incomparable requires a more careful analysis. Second, r_A and r_B need not be fixed rounds, but may be random variables. In this case, the parties must essentially guess $\min\{r_A, r_B\}$. Still, we can obtain a weak OT protocol in which Bob has noticeable uncertainty about x_{1-b}, and which is therefore amenable to amplification. □

Using an analogous approach, we also show the following in the appendix:

[7]If a round *begins* with a call to the external functionality $\mathcal{G}$, then the round concludes when the parties receive their output from this external functionality. Extracting *during* round r means that the simulator extracts after seeing the adversary's input to the external functionality, and before delivering the corresponding output.

Lemma 4. *$\mathcal{F}^{2\times 2}_{\text{EXCH}} \sqsubseteq \mathcal{F}_{\text{COIN}}$ is equivalent to the* sh-OT *assumption.*

5.1 Extension to Reactive Functionalities

In Appendix C, we extend all of the results in this section to a large class of *reactive* functionalities — namely, those functionalities which can be modelled as a finite state machine. We note that this class of functionalities also includes those which do not give symmetric output to the two parties.

6 Expanding the Framework

The framework that we have presented is quite general thanks to the general nature of functionalities. However, there are several dimensions in which this framework can be extended, leading to possibly further computational complexity assumptions to appear. We mention a few such extensions in Appendix E.

Briefly, one could consider a larger variety of functionalities — "fair" functionalities, randomized functionalities — etc. Also, one could consider alternate notions of reductions, both weaker and stronger than what we have employed. It will indeed be interesting if a computational complexity assumption manifests under one of these extensions, but not within the specific instantiation of the framework we have considered. Another possible extension is to consider inifinite functionalities (as opposed to finite state functionalities with finite input/output alphabet); in this case it is likely that a large continuum of complexity assumptions are generated in this framework. It is an interesting challenge to carry out this extension to infinite functionalities without sacrificing the conceptual clarity of the set of complexity assumptions produced by the framework.

References

[BEA89] Donald Beaver. Perfect privacy for two-party protocols. In Joan Feigenbaum and Michael Merritt, editors, *Proceedings of DIMACS Workshop on Distributed Computing and Cryptography*, volume 2, pages 65–77. American Mathematical Society, 1989.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc.* 20*th STOC*, pages 1–10. ACM, 1988.

[BMM99] Amos Beimel, Tal Malkin, and Silvio Micali. The all-or-nothing nature of two-party secure computation. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 1999.

[CAN01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Electronic Colloquium on Computational Complexity (ECCC) TR01-016, 2001. Previous version "A unified framework for analyzing security of protocols" availabe at the ECCC archive TR01-016. Extended abstract in FOCS 2001.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In *Proc.* 20*th STOC*, pages 11–19. ACM, 1988.

[CK89] Benny Chor and Eyal Kushilevitz. A zero-one law for boolean privacy (extended abstract). In *STOC*, pages 62–72. ACM, 1989.

[CKL03] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*. Springer, 2003.

[DG03] Ivan Damgård and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. In *Proc.* 35*th STOC*, pages 426–437. ACM, 2003.

[DKS99] Ivan Damgård, Joe Kilian, and Louis Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In Jacques Stern, editor, *EUROCRYPT*, volume 1592 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 1999.

[GKM$^+$00] Yael Gertner, Sampath Kannan, Tal Malkin, Omer Reingold, and Mahesh Viswanathan. The relationship between public key encryption and oblivious transfer. In *FOCS*, pages 325–335, 2000.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play ANY mental game. In ACM, editor, *Proc.* 19*th STOC*, pages 218–229. ACM, 1987. See [GOL04, Chap. 7] for more details.

[GOL04] Oded Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.

[HNRR06] Danny Harnik, Moni Naor, Omer Reingold, and Alon Rosen. Completeness in two-party secure computation: A computational view. *J. Cryptology*, 19(4):521–552, 2006.

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *Proc.* 30*th FOCS*, pages 230–235. IEEE, 1989.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591. Springer, 2008.

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *STOC*, pages 44–61. ACM, 1989.

[KIL88] Joe Kilian. Founding cryptography on oblivious transfer. In *STOC*, pages 20–31. ACM, 1988.

[KIL89] Joe Kilian. *Uses of Randomness in Algorithms and Protocols.* PhD thesis, Department of Electrical Engineering and Computer Science, Mas sachusetts Institute of Technology, 1989.

[KIL91] Joe Kilian. A general completeness theorem for two-party games. In *STOC*, pages 553–560. ACM, 1991.

[KIL00] Joe Kilian. More general completeness theorems for secure two-party computation. In *Proc. 32th STOC*, pages 316–324. ACM, 2000.

[KKMO00] Joe Kilian, Eyal Kushilevitz, Silvio Micali, and Rafail Ostrovsky. Reducibility and completeness in private computations. *SIAM J. Comput.*, 29(4):1189–1208, 2000.

[KMQ08] Daniel Kraschewski and Jörn Müller-Quade. Completeness theorems with constructive proofs for symmetric, asymmetric and general 2-party-functions, 2008. Unpublished Manuscript, 2008. http://iks.ira.uka.de/eiss/completeness.

[KMQR09] Robin Künzler, Jörn Müller-Quade, and Dominik Raub. Secure computability of functions in the it setting with dishonest majority and applications to long-term security, 2009.

[KN08] Gillat Kol and Moni Naor. Cryptography and game theory: Designing protocols for exchanging information. In Ran Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 320–339. Springer, 2008.

[KUS89] Eyal Kushilevitz. Privacy and communication complexity. In *FOCS*, pages 416–421. IEEE, 1989.

[MPR09A] Hemanta Maji, Manoj Prabhakaran, and Mike Rosulek. A zero-one law for deterministic 2-party secure computation. Manuscript. Results from preliminary version appear in [ROS09], 2009.

[MPR09B] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Complexity of multi-party computation problems: The case of 2-party symmetric secure function evaluation. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 256–273. Springer, 2009.

[PR08A] Manoj Prabhakaran and Mike Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2008.

[PR08B] Manoj Prabhakaran and Mike Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. *Electronic Colloquium on Computational Complexity (ECCC)*, 15(50), 2008.

[PR08C] Manoj Prabhakaran and Mike Rosulek. Towards robust computation on encrypted data. In Josef Pieprzyk, editor, *ASIACRYPT*, volume 5350 of *Lecture Notes in Computer Science*, pages 216–233. Springer, 2008.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer, 2008.

[ROS09] Mike Rosulek. *The Structure of Secure Multi-Party Computation.* PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2009.

[YAO82] Andrew Chi-Chih Yao. Protocols for secure computation. In *Proc. 23rd FOCS*, pages 160–164. IEEE, 1982.

A Some Cryptographic Complexity Classes

In this section we identify a few major classes of two-party symmetric secure function evaluation (SSFE) functionalities. SSFE functionalities, starting with the original *Millionaire's Problem* proposed by Yao [YAO82], are perhaps the most well-studied class of two-party functionalities. Several questions regarding the cryptographic complexity of 2-party SSFE functionalities (with respect to $\sqsubseteq_{\text{STAT}}$) were addressed in [MPR09B] based on which the following classifications can be carried out. (But some of the classes below, like "exchange-like" functionalities are introduced here, as they have special significance when connections with computational complexity assumptions are considered.)

The classes listed below, except the first two, are "downward closed" with respect to $\sqsubseteq_{\text{STAT}}$. That is, if $\mathcal{G}$ is a functionality in a class and $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ then $\mathcal{F}$ also falls in the same class. (The first one is "upward closed.") These classes are graphically represented in Figure 1.

Complete Functionalities. These are functionalities that are statistically "complete." That is, for all $\mathcal{G}$ in this class, and all functionalities $\mathcal{F}$, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$. Based on the completeness of the oblivious transfer functionality [KIL88] (which remains true with respect to the reduction $\sqsubseteq_{\text{STAT}}$ as well [KIL89, IPS08]), Kilian [KIL91] gave a combinatorial characterization for complete SSFE functionalities, as the evaluation of functions containing an "OR-minor" (also called embedded OR). Indeed, an example of an SSFE functionality in this class is SFE_{OR} where OR is the function $\boxed{\begin{smallmatrix}0&1\\1&1\end{smallmatrix}}$.

Unclassified Functionalities. Among incomplete functionalities, we leave a set of functionalities as "unclassified." These are functionalities which are neither complete, nor passive-trivial (see next class). We know that this class is not empty: the SSFE functionality corresponding to the evaluation of the function $\boxed{\begin{smallmatrix}1&1&2\\4&0&2\\4&3&3\end{smallmatrix}}$ is known to fall into this category [BEA89, KUS89, KKMO00]. These functions do have a combinatorial characterization in terms of minors — they contain an "undecomposable" minor [BEA89, KUS89], but no OR minor. However, we call these functionalities unclassified as we have little insight into their cryptographic properties or different sub-classes.

Passive-Trivial Functionalities. These are functionalities securely realizable against a passive (a.k.a honest-but-curious, or semi-honest) adversary in a computationally unbounded environment. For SSFE functionalities, such functions have an explicit combinatorial characterization, namely that they are evaluations of what are called "decomposable" functions [KUS89, BEA89, MPR09B, KMQR09].

An important example in this class (though not an SFE functionality) is the commitment functionality, denoted by $\mathcal{F}_{\text{COM}}$. [MPR09B] show that $\mathcal{F}_{\text{COM}}$ gives a *complete characterization* for the SFE functionalities in this class, in the sense for an SFE functionality $\mathcal{F}$, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}_{\text{COM}}$ iff $\mathcal{F}$ is passive-trivial.

Exchange-Like Functionalities. This is a important sub-class of Passive-Trivial functionalities defined to be the class of functionalities that reduce to a simultaneous exchange functionality. That is, $\mathcal{F}$ is in this class iff there exists an integer i such that $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}^{i,i}_{\text{EXCH}}$. It turns out that among SSFE functionalities, exchange-like functionalities are "isomorphic" to $\mathcal{F}^{i,j}_{\text{EXCH}}$ for some integers i and j. As it turns out, the cryptographic complexity gaps within this class and across this class are equivalent to sh-OT. Further, among passive-trivial functionalities, all the results we derive point to the conjecture that these are the only reductions that require sh-OT. This class can be naturally extended to reactive or randomized functionalities, with the same definition, and our results regarding sh-OT assumption extend to them as well.

Exchange-Free Functionalities. This is the class of functionalities to which no non-trivial exchange functionality can be statistically reduced. That is, a functionality $\mathcal{G}$ is in this class iff $\mathcal{F}^{2,2}_{\text{EXCH}} \not\sqsubseteq_{\text{STAT}} \mathcal{G}$. All standalone-trivial functionalities (see the next class) are of this kind. But there are other functionalities too: for instance, $\boxed{\begin{smallmatrix}1&1&2\\3&4&3\end{smallmatrix}}$ (not standalone-trivial) and $\boxed{\begin{smallmatrix}1&1&2\\4&0&2\\4&3&3\end{smallmatrix}}$ (an unclassified functionality) can be shown to be exchange-free.

A useful combinatorial property of exchange-free SSFE functionalities which are passive-trivial is that they are evaluations of functions whose function tables are "uniquely decomposable," as defined in [MPR09B]. (However, being uniquely decomposable does not necessarily mean that the SFE is exchange-free: an example is the function $\boxed{\begin{smallmatrix}1&1&2\\5&0&2\\4&3&3\end{smallmatrix}}$.)

Standalone-Trivial Functionalities. These are functionalities securely realizable against a stand-alone adversary in a computationally unbounded environment. SSFE functionalities in this class were combinatorially characterized in [KMQR09, MPR09B] They are passive-trivial and it turns out they are all exchange-free functionalities. An important example of an SSFE in this class – and indeed the simplest non-trivial one – is the evaluation of the "cut-and-choose" function $\boxed{\begin{smallmatrix}0&1\\0&2\end{smallmatrix}}$, in which Bob can choose to learn Alice's input or not (and Alice learns Bob's choice). There is a simple standalone-secure protocol for this functionality (in which Bob first sends his input to Alice, and if necessary Alice sends her input to Bob); however there is no UC secure protocol for this functionality. Another example of a standalone-trivial functionality is $\boxed{\begin{smallmatrix}1&1&2&2\\3&4&3&4\end{smallmatrix}}$. (In contrast, as mentioned above, $\boxed{\begin{smallmatrix}1&1&2\\3&4&3\end{smallmatrix}}$ is not standalone-trivial, though it is passive-trivial and exchange-free.)

Trivial Functionalities. These are functionalities securely realizable against a general adversary in a computationally unbounded environment, using protocols which only rely on (private) communication channels.[8] Essentially, these functionalities have a simple protocol in which one party sends a single message to the other party. Among SSFE functionalities, these are isomorphic to evaluation of functions with a one-dimensional function-table (that is, one of the parties has a fixed input). This also corresponds to the intersection of the classes exchange-like and exchange-free. (An instance of a non-trivial *randomized* function which is exchange-like and exchange-free is $\mathcal{F}_{\text{COIN}}$, the coin-flipping functionality.)

B Details for the Proof of Theorem 7

In this section we complete the proof of Theorem 7 outlined in the main body. What remains to be shown are the details of Lemma 3 — namely, that if the dimensions of one exchange function $\mathcal{F}$ are not "smaller" than the dimensions of another exchange function $\mathcal{G}$, then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies the sh-OT assumption.

We develop the proof in several parts. First, to introduce our approach to proving separations involving exchange functions, we show that $\mathcal{F}^{2\times 2}_{\text{EXCH}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{COIN}}$ implies sh-OT assumption. Then we show that $\mathcal{F}^{i\times j}_{\text{EXCH}} \sqsubseteq_{\text{PPT}} \mathcal{F}^{(i-1)\times(j-1)}_{\text{EXCH}}$ implies the sh-OT assumption, and finally that $\mathcal{F}^{i\times j}_{\text{EXCH}} \sqsubseteq_{\text{PPT}} \mathcal{F}^{i'\times j'}_{\text{EXCH}}$ implies the sh-OT assumption, where $\min\{i', j'\} < i, j \leq \max\{i', j'\}$. These two cases suffice to prove the desired characterization.

B.1 Reduction to $\mathcal{F}_{\text{COIN}}$.

Lemma 5. *$\mathcal{F}^{2\times 2}_{\text{EXCH}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{COIN}}$ implies the sh-OT assumption.*

Proof. Let π be a secure protocol for $\mathcal{F}^{2\times 2}_{\text{EXCH}}$ in the $\mathcal{F}_{\text{COIN}}$-hybrid world. We will transform π to obtain a secure protocol for $\mathcal{F}_{\text{OT}}$ against semi-honest adversaries.

Let s_B be the random variable denoting the round in which the simulator extracts from a passively corrupt Alice and sends her input to $\mathcal{F}^{2\times 2}_{\text{EXCH}}$. Fix any passive adversarial strategy for Alice which outputs a guess of Bob's input at each step of the protocol, and define t_A as the random variable denoting the round when this guess

[8]Recall that in our model functionalities — including communication channels — interact only with the parties. A channel with an eavesdropper is modeled as a 3-party functionality.

Protocol for a weak variant of $\mathcal{F}_{\text{OT}}$. Alice has inputs $x_0, x_1 \in \{0,1\}$, and Bob has input $b \in \{0,1\}$.

1. Alice runs two instances of the protocol π with Bob, using inputs x_0 and x_1, respectively.
2. Bob picks a random $r \in [r(\kappa)]$, where $r(\kappa)$ is a polynomial bound on the number of rounds in π.
3. In the bth instance of π, Bob runs the simulator for π against Alice (including simulating her interface with instances of $\mathcal{F}_{\text{COIN}}$), and halts the interaction after the rth round of π.
4. In the $(1-b)$ instance of π, Bob runs the π protocol honestly with Alice on a fixed input (say, 0), and also honestly simulates all instances of $\mathcal{F}_{\text{COIN}}$ for Alice. Bob halts the interaction after the rth round of π.
5. If the simulator has extracted x_b, then Bob outputs it. Otherwise, he asks Alice for (x_0, x_1), and she sends it to him.

Figure 2: Weak oblivious transfer protocol, using any secure protocol π for $\mathcal{F}_{\text{EXCH}}^{2\times 2}$ in the $\mathcal{F}_{\text{COIN}}$-hybrid world.

is correct with probability at least $\zeta = 3/4$ (where the probability is over the randomness independent of Alice's view), *when interacting with the simulator*. By the definition of the simulation, Alice's view is completely independent of Bob's input through the first s_B rounds (even in the presence of an ideal $\mathcal{F}_{\text{COIN}}$). Thus $t_A \geq s_B + 1$, and in particular, $E[t_A] \geq E[s_B] + 1$.

Now consider running this passive adversarial strategy for Alice against an honest Bob in the actual protocol execution, instead of against the simulator. We define u_A to be the random variable denoting the first round in which Alice's guess is correct with probability at least ζ. By the security of π, these two interactions must be indistinguishable to this Alice strategy, thus $|E[u_A] - E[t_A]| < \epsilon/\zeta = \epsilon'$, where ϵ is the negligible simulation error of the protocol. Thus $E[u_A] \geq E[s_B] + 1 - \epsilon'$.

Similarly we can define u_B and s_A and conclude that $E[u_B] \geq E[s_A]+1-\epsilon'$. Then, either $E[u_A] \geq E[s_A]+1-\epsilon'$, or $E[u_B] \geq E[s_B]+1-\epsilon'$; otherwise we would get that $E[u_A] < E[s_A]+1-\epsilon' \leq E[u_B] < E[s_B]+1-\epsilon'$.

By symmetry, we assume that $E[u_B] \geq E[s_B]+1-\epsilon'$. In other words, in an interaction with an honest Alice, the simulator will, on average, extract Alice's input earlier than any passive Bob could guess Alice's input with probability at least ζ.

Now consider the protocol given in Figure 2. First, since Alice cannot distinguish a simulated instance of π from an honest execution of π, Alice has no advanatage in predicting Bob's bit b. Thus the protocol gives complete privacy for Bob.

Then a passively corrupt Bob in this protocol can guess Alice's input x_{1-b} correctly with probability at most

$$\begin{aligned}
&\Pr[s_B \leq r < u_B]\zeta + (1 - \Pr[s_B \leq r < u_B]) \\
&\quad = 1 - (1-\zeta)\Pr[s_B \leq r < u_B] \\
&\quad \leq 1 - (1-\zeta)E[u_B - s_B]/r(\kappa) \\
&\quad \leq 1 - (1-\zeta)(1-\epsilon')/r(\kappa)
\end{aligned}$$

by the definition of u_B. Or, in other words, Bob's guess is incorrect with probability at least $(1-\zeta)(1-\epsilon')/r(\kappa)$, which is an inverse polynomial in the security parameter. In Appendix D, we show how a weak $\mathcal{F}_{\text{OT}}$ protocol with this security property can be amplified to give a full-fledged (semi-honest) $\mathcal{F}_{\text{OT}}$ protocol. □

B.2 Reductions Between Exchange Functions

We first establish a convenient technical lemma:

Lemma 6. *For each $j \in [i]$, let $\mathcal{D}_j$ be a probability distribution over the elements $\{m_1, \ldots, m_{i-1}\}$. Now consider the following experiment: Choose $j \in [i]$ in random, and then output a sample according to $\mathcal{D}_j$.*

The probability of correctly predicting j given only the output of this procedure is at most $(i-1)/i$.

Proof. Let $p_{u,v}$ be the probability of sampling message m_v when using $\mathcal{D}_u$. So, we have:

$$\sum_{v=1}^{i-1} p_{u,v} = 1 \text{ for all } u \in [i]$$

Let $q_{v,u}$ be the probability of outputting u after seeing message v. So, we have:

$$\sum_{u=1}^{i} q_{v,u} = 1 \text{ for all } v \in [i-1]$$

The probability of being correct is:

$$\zeta = \frac{\sum_{u=1}^{i} \sum_{v=1}^{i-1} p_{u,v} q_{v,u}}{i}$$

This is maximized if $q_{v,u} = \alpha_u p_{u,v}$. Therefore, $\zeta \leq \frac{\sum_{u=1}^{i} \alpha_v \sum_{v=1}^{i-1} p_{u,v}^2}{i} \leq \frac{\sum_{u=1}^{i-1} \alpha_u}{i} = \frac{i-1}{i}$ □

Before we prove the general result, let us prove an intermediate result

Lemma 7. *Let $i \geq 3$. Then $\mathcal{F}_{\text{EXCH}}^{i\times i} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$ implies the* sh-OT *assumption.*

Proof. The proof is very similar to that of the previous lemma. However, now that the purported protocol π can use $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$, we must consider information about the parties' inputs that is exchanged via the ideal functionality.

Note that in the proof of the previous lemma, we would obtain a suitable weak OT protocol (i.e., amenable to amplification) even if ζ is at most $1 - \frac{1}{\text{poly}}$ in the security parameter

Consider $\zeta = c + \frac{i-1}{i}$, where $c > 0$ is any constant. As before, we let s_B be the round during which the simulator extracts from a passively corrupt Alice. Thus, Alice may send an input to her interface of $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$, then the simulator will send the extracted input to $\mathcal{F}_{\text{EXCH}}^{i\times i}$, receive the output, and then complete the round by simulate the response of the simulated $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$ functionality to Alice.

The simulator will complete the round by simulating a response from $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$, which will be an element of $[i-1]$. At the start of round s_B, Alice's view is independent of the honest Bob's input $y \in [i]$ to $\mathcal{F}_{\text{EXCH}}^{i\times i}$. There are only $i-1$ possible responses the simulator can provide after receiving y from the ideal $\mathcal{F}_{\text{EXCH}}^{i\times i}$ functionality. So after round s_B is complete, Alice cannot guess y with probability greater than $(i-1)/i < \zeta$, where $\nu(\cdot)$ is negligible. Since t_B is defined as the first point at which Alice can guess Bob's input with probability at least ζ, we have $t_B \geq s_A + 1$.

Similarly we can conclude that $t_A \geq s_B + 1$. Rest of the proof is identical to the proof mentioned above. □

Finally we move to our general result.

Lemma 8. *Let i, j, i', j' be such that ($i > i'$ or $j > j'$) and ($i > j'$ or $j > i'$). Then $\mathcal{F}_{\text{EXCH}}^{i\times j} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{EXCH}}^{i'\times j'}$ implies the* sh-OT *assumption.*

Proof. The proof is very similar to that of the previous lemma. However, now that the purported protocol π can use $\mathcal{F}_{\text{EXCH}}^{i'\times j'}$, we must consider information about the parties' inputs that is exchanged via the ideal functionality.

Note that in the proof of the previous lemma, we would obtain a suitable weak OT protocol (i.e., amenable to amplification) even if ζ is $1 - \frac{1}{\text{poly}}$ in the security parameter, and one of $\{E[t_B - s_B], E[t_A - s_A]\}$ is at least $\frac{1}{\text{poly}}$ in the security parameter.

Case 1: ($\max\{i, j\} > \max\{i', j'\}$): Suppose $i \geq j$ and $i > i' \geq j'$ and Bob feeds input from $[i]$ into the ideal functionality. We define $\zeta = c + \frac{i-1}{i}$. Now we define s_B and t_A as we had done earlier. Similar to the argument in the previous lemma we get that $t_A \geq s_B + 1$ (because $i - 1 \geq i' \geq j'$). It is always the case that $t_B \geq s_A$. So, we get the condition that $t_A \geq s_B + 1$ and $t_B \geq s_A$.

In general we can say that:

$$(t_A \geq s_B \text{ and } t_B \geq s_A + 1), \text{ or}$$
$$(t_B \geq s_A \text{ and } t_A \geq s_B + 1)$$

These conditions imply that:

$$E[u_A] \geq E[s_A] + \left(\frac{1}{2} - \epsilon'\right), \text{ or}$$
$$E[u_B] \geq E[s_B] + \left(\frac{1}{2} - \epsilon'\right)$$

Observe that in our weak OT construction, all we needed was that $E[u_A - s_A]$ or $E[u_B - s_B]$ is $\frac{1}{\text{poly}}$ in the security parameter. So, we can continue with our weak OT construction as we had mentioned earlier.

Case 2: ($\min\{i', j'\} < i, j \leq \max i', j'$): Observe that even if for some polynomial $\lambda(\cdot)$ we have:

$$\left(E[t_A] \geq E[s_B] \text{ and } E[t_B] \geq E[s_A] + \frac{1}{\lambda(\kappa)}\right), \text{ or}$$
$$\left(E[t_B] \geq E[s_A] \text{ and } E[t_A] \geq E[s_B] + \frac{1}{\lambda(\kappa)}\right)$$

we can use the approach mentioned above to get the weak OT protocol. So, we just need to consider the case when $E[t_B] \in \left[E[s_A], E[s_A] + \frac{1}{\lambda(\kappa)}\right)$ and $E[t_A] \in \left[E[s_B], E[s_B] + \frac{1}{\lambda(\kappa)}\right)$, where $\lambda(\cdot)$ is a suitably chosen large polynomial.

In this case, we will prove that:

1. $\Pr(t_B \geq s_B + 1)$ or $\Pr(t_A \geq s_A + 1)$ is $\geq \frac{1}{5}$
2. $|\Pr(u_A = i) - \Pr(t_A = i)|$, $|\Pr(u_B = i) - \Pr(t_B = i)|$ are both $\leq \frac{1}{\rho(\kappa)}$ for any polynomial ρ

These will imply that our weak OT construction will work in this case as well.

Now, we show that the above mentioned properties hold. If $E[t_B] \in \left[E[s_A], E[s_A] + \frac{1}{\lambda(\kappa)}\right)$ and $E[t_A] \in \left[E[s_B], E[s_B] + \frac{1}{\lambda(\kappa)}\right)$, then with probability $\geq 1 - \frac{2}{\lambda(\kappa)n}$ we will have the event that $t_B = s_A$ and $t_A = s_B$. Consider the set of rounds S where $t_A = s_B$. Similarly define T to be the set of rounds where $t_B = s_A$. WLOG, we can assume that Alice uses i' side of $\mathcal{F}_{\text{XOR}(i'\times j')}$ only in even rounds and the j' side of the $\mathcal{F}_{\text{XOR}(i'\times j')}$ only in odd rounds. So, we conclude that the sets S and T are mutually disjoint.

Let $x_S(i)$ be the probability of the event $t_A = s_B \leq i$ happens. Similarly define $x_T(i)$ as the probability of the event $t_B = s_A \leq i$ happens. Initially $x_S(0) = x_T(0) = 0$ and $x_S(n) = x_T(n) = 1 - \frac{2}{\lambda(\kappa)n}$. So look at the first i such that $x_S(i)$ or $x_T(i)$ becomes $\geq \frac{1}{2}\left(1 - \frac{2}{\lambda(\kappa)n}\right)$. Observe that at any given round only $x_S(i)$ or $x_T(i)$ changes. WLOG assume that $x_S(i)$ reaches the threshold first. Then since $y_S(i)$ could not have changed at this round, we get that $y_S(i) \leq \frac{1}{2}\left(1 - \frac{2}{\lambda(\kappa)n}\right)$. Then we see that with probability $\geq \left(\frac{1}{2} - \frac{1}{\lambda(\kappa)n}\right)^2 \geq \frac{1}{4} - \frac{2}{\lambda(\kappa)n} \geq \frac{1}{5}$, we have the event that $s_B \leq t_B - 1$.

Now, all we need to show is that $\Pr(u_B = i)$ and $\Pr(t_B = i)$ are $1/\text{poly}$-close. We pick a suitable polynomial ρ. We run an honest execution of the protocol against a simulator for Alice. We can estimate $\Pr(t_B = i)$ within $\frac{1}{\rho}$ additive error in polynomial time. Similarly, we run an honest execution of the protocol against honest Alice. We can estimate $\Pr(u_B = i)$ within $\frac{1}{\rho}$ additive error in polynomial time.

If $|\Pr(t_B = i) - \Pr(u_B = i)| > \frac{3}{\rho}$, then we can create a polynomial time distinguisher which distinguishes between the real and ideal world. So, for every round $i \in [r(\kappa)]$, $|\Pr(t_B = i) - \Pr(u_B = i)| \le \frac{3}{\rho}$.

Given the guarantee that, for all $i \in [r(\kappa)]$, $|\Pr(t_B = i) - \Pr(u_B = i)| \le \frac{3}{\rho}$ and $\Pr(s_B \le t_B - 1) \ge \frac{1}{5}$, the construction given earlier gives us a weak OT. □

C Reactive Exchange-Like Functionalities

In this section we extend the results of Section 5 to a large class of *reactive* functionalities. Namely, the following:

Definition 5. *A* deterministic finite functionality (DFF) *is a functionality with a finite set of internal states Q, whose behavior is as follows:*

1. *Set the internal state q to the distinguished start state $q_0 \in Q$.*
2. *Wait for inputs $x \in X$ from Alice and $y \in Y$ from Bob, where X, Y are finite input sets.*
3. *If $\delta(q, x, y)$ is defined, then send "delayed outputs" $f_A(q, x, y)$ to Alice and $f_B(q, x, y)$ to Bob, where δ, f_A, f_B are deterministic functions.*
4. *Set $q \leftarrow \delta(q, x, y)$ and repeat from step 2.*

To reason about the behavior of reactive functionalities, we follow [MPR09A, ROS09] and develop a way of saying that one input x "achieves the same effect" as another input x', in the context of a reactive functionality. Intuitively, this happens when every behavior that can be induced by sending x at a certain point can also be induced by sending x' instead, and thereafter appropriately translating subsequent inputs and outputs. We can define this formally in terms of the UC security definition:

Definition 6 (Dominating Inputs)**.** *Let $\mathcal{F}$ be a DFF, and let $x, x' \in X$ be inputs for Alice. We say that x* dominates *x' in the first round of $\mathcal{F}$, and write $x \ge_A x'$, if there is a secure protocol for $\mathcal{F}$ in the $\mathcal{F}$-hybrid setting, where the protocol for Bob is to run the dummy protocol (as Bob), and the protocol for Alice has the property that whenever the environment provides input x' for Alice in the first round, the protocol instead sends x to the functionality in the first round.*

We define domination for Bob inputs analogously, with the roles of Alice and Bob reversed. Note that the definition requires that any behavior of $\mathcal{F}$ that is possible when Alice uses x' as her first input can also be induced *in an online fashion* by using x as her first input (and subsequently translating inputs/outputs according to some strategy). Domination is reflexive and transitive.

Definition 7 (Simple States)**.** *Let $\mathcal{F}$ be a DFF, and let q be one of its states. We define $\mathcal{F}[q]$ as the functionality obtained by modifying $\mathcal{F}$ so that its start state is q.*

We say that q is a simple state *if:*

- *The input/output behavior of $\mathcal{F}$ at state q — $(f_A(q, \cdot, \cdot), f_B(q, \cdot, \cdot))$ — is (isomorphic to) an exchange function; and*
- *For all Alice inputs $x, x' \in X$ such that $f_B(q, x, \cdot) \equiv f_B(q, x', \cdot)$, there exists an Alice input $x^* \in X$ such that $x^* \ge_A x$ and $x^* \ge_A x'$ in $\mathcal{F}[q]$; and*
- *For all Bob inputs $y, y' \in Y$ such that $f_A(q, \cdot, y) \equiv f_A(q, \cdot, y')$, there exists a Bob input $y^* \in Y$ such that $y^* \ge_B y$ and $y^* \ge_B y'$ in $\mathcal{F}[q]$.*

Suppose q is a simple state. Then we can define $x \overset{q}{\sim} x'$ if $f_B(q, x, \cdot) \equiv f_B(q, x', \cdot)$. The relation $\overset{q}{\sim}$ induces equivalence classes over X. When q is a simple state, then within each such equivalence class, there exists at least one input x^* which dominates all other members of its class. For each equivalence class, we arbitrarily pick a single such input x^* and call it a *master* input for state q. Similarly we define master inputs for Bob by exchanging the roles of Alice and Bob.

Definition 8. *Let $\mathcal{F}$ be a DFF, We say that a transition is* safe *if it leaves a simple state q on inputs (x, y), where x and y are both master inputs for state q.*

C.1 Exchange-Like Definition

Our generalization of exchange-like functionalities is in terms of these automata-theoretic properties.

Definition 9 (Exchange-like)**.** *We say that a DFF $\mathcal{F}$ is* exchange-like *if no non-simple state in $\mathcal{F}$ is reachable via a sequence of safe transitions from $\mathcal{F}$'s start state.*

We justify the use of the term "exchange-like" in the following lemma. Namely, exchange-like functionalities are equivalent to a collection of several (non-reactive) exchange functions.

Lemma 9. *Let $\langle \mathcal{F}_1, \dots, \mathcal{F}_n \rangle$ be a DFF which in the first round accepts input $k \in [n]$ from Alice, outputs k to Bob, and then simulates $\mathcal{F}_k$.*

If a DFF $\mathcal{G}$ is exchange-like, then $\mathcal{G}$ is equivalent (under $\sqsubseteq_{\text{STAT}}$ reductions) to some $\langle \mathcal{F}_{\text{EXCH}}^{i_1 \times j_1}, \dots, \mathcal{F}_{\text{EXCH}}^{i_n \times j_n} \rangle$.

Proof sketch. Given $\mathcal{G}$, we first define a related functionality $R(\mathcal{G})$ which is simply $\mathcal{G}$ with all non-safe transitions deleted. Then using an argument almost identical to [MPR09A], we have that $\mathcal{G} \sqsubseteq R(\mathcal{G}) \sqsubseteq \mathcal{G}$ (see the following lemmas). Intuitively, for these functionalities, the parties can be made to use only "master" inputs without loss of generality.

Now for each reachable state q in $R(\mathcal{G})$, its input/output function at that state is an $i_q \times j_q$ exchange function. Let $\mathcal{F} = \langle \mathcal{F}_{\text{EXCH}}^{i_q \times j_q} \mid q \in Q \rangle$; then the protocol for $R(\mathcal{G})$ using access to $\mathcal{F}$ is for both parties to do the following: Maintain the current state q, and in each round, instantiate a new instance of $\mathcal{F}$ and ensure that Alice sends input q to $\mathcal{F}$ (Bob aborts otherwise). Then use $\mathcal{F}$ again to perform the input/output function of $\mathcal{G}$. The output of $\mathcal{F}$ uniquely determines both party's inputs, and thus the next value of $\mathcal{G}$'s internal state q, so we repeat.

To show that $\mathcal{F} \sqsubseteq R(\mathcal{G})$, for each reachable state q in $R(\mathcal{G})$, let $(x_1, y_1), \ldots, (x_n, y_n)$ be a sequence of inputs that leaves $R(\mathcal{G})$ in state q. The protocol for $\mathcal{F}$ is to have Alice first send her input q to Bob. Then both parties send the corresponding input sequence to $R(\mathcal{G})$ to place it in state q. Either party can determine from its view whether the other party has input the correct sequence. If this is not the case, then the parties abort. Otherwise, they send their next round inputs to $R(\mathcal{G})$ directly and use the output as their own output (after normalizing the inputs/outputs to $[i_q] \times [j_q]$). □

To complete the proof sketched above, we now describe the construction of a "normalized" version $R(\mathcal{F})$ of an exchange-like functionality $\mathcal{F}$. We first define an intermediate functionality:

Definition 10. *We define $r(\mathcal{F})$ to be the functionality which runs $\mathcal{F}$, except that in the first round only, it allows only* safe *transitions to be taken (i.e., transitions on master inputs only). $r(\mathcal{F})$ can be written as a copy of $\mathcal{F}$ plus a new start state. The new start state of $r(\mathcal{F})$ duplicates all the safe transitions of $\mathcal{F}$'s start state.*

Observation 8. *If a safe transition was just taken in $\mathcal{F}$, then Alice (resp. Bob) can uniquely determine Bob's (resp. Alice's) input in the previous round and the current state of $\mathcal{F}$, given only the previous state of $\mathcal{F}$ and Alice's (resp. Bob's) input and output in the previous round.*

Proof. We will show that Alice has no uncertainty about which master input Bob used, thus no uncertainty about the resulting state of $\mathcal{F}$. If a safe transition was just taken from q, then q was a simple state and its associated SFE $(f_A(q,\cdot,\cdot), f_B(q,\cdot,\cdot))$ is isomorphic to an exchange function. Note that our definition of dominating inputs subsumes the definition of redundant inputs in the context of function isomorphism.

Thus if y, y' are distinct master inputs for Bob, then $f_A(q,\cdot,y) \not\equiv f_A(q,\cdot,y')$. As such, for any master input x for Alice, $f_A(q,x,y) \neq f_A(q,x,y')$. Alice has no uncertainty about which master input Bob used. This argument is symmetric for Bob as well. □

Lemma 10. *If the start state of $\mathcal{F}$ is simple, then $r(\mathcal{F}) \sqsubseteq \mathcal{F} \sqsubseteq r(\mathcal{F})$. Furthermore, if q is reachable from the start state of $\mathcal{F}$ via a safe transition, then $\mathcal{F}[q] \sqsubseteq \mathcal{F}$.*

Proof. The protocol for $r(\mathcal{F}) \sqsubseteq \mathcal{F}$ is the dummy protocol, since $r(\mathcal{F})$ implements simply a subset of the behavior of $\mathcal{F}$. Simulation is trivial unless in the first round, the corrupt party (say, Alice) sends an input x to $\mathcal{F}$ which is not a master input for q_0. The simulator must send the corresponding master input x^* (from the $\overset{q_0}{\sim}$ equivalence class of x) in the ideal world, and then it uses the translation protocol guaranteed by the definition of $x^* \geq_A x$ to provide a consistent view to Alice and induce correct outputs for Bob.

Similarly, the protocol for $\mathcal{F} \sqsubseteq r(\mathcal{F})$ is simply the dual of the above protocol. On input x in the first round, Alice sends x^* to $r(\mathcal{F})$, where x^* is the master input from the $\overset{q_0}{\sim}$-equivalence class of x. Thereafter, Alice runs the protocol guaranteed by the fact that $x^* \geq_A x$. Bob's protocol is analogous. Simulation is a trivial dummy simulation, since any valid sequence of inputs to $r(\mathcal{F})$ in the real world also produces the same outcome in the $\mathcal{F}$-ideal world ($r(\mathcal{F})$ implements a subset of the behavior of $\mathcal{F}$).

Note that in $r(\mathcal{F})$, the added start state has no incoming transitions; thus $(r(\mathcal{F}))[q] = \mathcal{F}[q]$ if q is a state in $\mathcal{F}$. So to show $\mathcal{F}[q] \sqsubseteq \mathcal{F}$, it suffices to show that $(r(\mathcal{F}))[q] \sqsubseteq r(\mathcal{F})$. Suppose q is reachable in $\mathcal{F}$ from the start state via safe transition on master inputs x^*, y^*. The protocol for $\mathcal{F}[q]$ is for Alice and Bob to send x^* and y^* to $r(\mathcal{F})$, respectively, as a "preamble". Each party can determine with certainty, given their input and output in this preamble, whether $r(\mathcal{F})$ is in state q (since only safe transitions can be taken from the start state of $r(\mathcal{F})$). If the functionality is not in q, then the parties abort. Otherwise, the functionality is $r(\mathcal{F})$ in state q as desired, so the parties thereafter run the dummy protocol. Simulation is trivial – the simulator aborts if the corrupt party does not send its specified input (x^* or y^*) in the preamble; otherwise it runs a dummy simulation. □

The claim used in the proof of Lemma 9 is the following:

Lemma 11. *Let $R(\mathcal{F})$ be $\mathcal{F}$ with all non-safe transitions removed. Then $\mathcal{F} \sqsubseteq R(\mathcal{F}) \sqsubseteq \mathcal{F}$.*

Proof. First we show that $\mathcal{F} \sqsubseteq R(\mathcal{F})$. We prove a stronger claim; namely that if q is safely reachable (i.e., reachable from the start state by a sequence of safe transitions) in $\mathcal{F}$, then $\mathcal{F}[q] \sqsubseteq (R(\mathcal{F}))[q]$. To prove this stronger claim, we construct a family of protocols $\hat{\pi}_q$, for every such q.

First, let π_q denote the protocol guaranteed by $\mathcal{F}[q] \sqsubseteq r(\mathcal{F}[q])$ (Lemma 10). Then the protocol $\hat{\pi}_q$ is as follows:

1. Run π_q to interact with the functionality.
2. After the first round, we will have sent an input to the functionality and received an output. Assuming that the functionality was $(R(\mathcal{F}))[q]$, use the first round's input/output to determine the next state q' (Observation 8)
3. Continue running π_q, but hereafter, instead of letting it interact directly with the functionality, we recursively instantiate $\hat{\pi}_{q'}$. We let our π_q instance interface with $\hat{\pi}_{q'}$, which we let interact directly with the functionality.

The protocol is recursive, and after k rounds, must maintain a stack depth of size k. We prove by induction on k that $\hat{\pi}_q$ is a secure protocol for $\mathcal{F}[q]$ using $(R(\mathcal{F}))[q]$, against environments that run the protocol for $k \geq 0$ steps. The claim is trivially true for $k = 0$.

Note that simulation is trivial if either party is corrupt. Such an adversary is running the protocol interacting with $(R(\mathcal{F}))[q]$, which is a subset of the functionality $\mathcal{F}[q]$. Thus the simulator is a dummy simulator. It suffices to show that the output of the protocol is correct (indistinguishable from the ideal interaction) when both parties are honest.

In the first round, both parties are running π_q, interacting with $(R(\mathcal{F}))[q]$. Although π_q is designed to interact with $r(\mathcal{F}[q])$, the behavior of both these functionalities is identical in the first round (including the next-state function). Thus the first round of outputs is correct, by the security of π_q. For the same reason, step 2 of $\hat{\pi}_q$ correctly identifies the next state q' of $(R(\mathcal{F}))[q]$. Clearly $(R(\mathcal{F}))[q][q'] = (R(\mathcal{F}))[q']$, so after step 1 of the protocol, the functionality is identical to a fresh instantiation of $(R(\mathcal{F}))[q']$. At the same time, we also instantiate a fresh instance of $\hat{\pi}_{q'}$ to interact with this functionality. By the inductive hypothesis, hereafter π_q is interacting with an interface that is indistinguishable from an ideal interaction with $\mathcal{F}[q']$. However, an external functionality which behaves like $R(\mathcal{F})[q]$ in the first round, then after transitioning to state q' behaves like $\mathcal{F}[q']$, is simply the functionality $r(\mathcal{F}[q])$. In other words, the entire protocol $\hat{\pi}_q$ is indistinguishable from running π_q on $r(\mathcal{F}[q])$. By definition of π_q, this is indistinguishable from an ideal interaction with $\mathcal{F}[q]$ itself.

The protocol for $R(\mathcal{F}) \sqsubseteq \mathcal{F}$ is the dual of the above protocol. The protocol is the dummy protocol, and the simulator recursively uses the protocols π_q as above. □

C.2 Main Classification

We now prove the main result regarding exchange-like functionalities, that is:

Theorem 9. *Let $\mathcal{F}$ and $\mathcal{G}$ be DFFs. If $\mathcal{G}$ is exchange-like and non trivial, then either $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, or $\mathcal{F} \sqsubseteq \mathcal{G}$ is equivalent to the SHOT assumption.*

We split the proof into two parts, depending on whether $\mathcal{F}$ itself is also exchange-like.

When $\mathcal{F}$ is exchange-like

For the case where both $\mathcal{F}$ and $\mathcal{G}$ are exchange-like, we prove a simple combinatorial characterization for when $\mathcal{F} \sqsubseteq \mathcal{G}$, which generalizes our result for SFE functionalities. Namely, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ if and only if every exchange contained in $\mathcal{F}$ can "fit inside" an exchange contained in $\mathcal{G}$. Otherwise, the existence of a secure protocol is equivalent to the sh-OT assumption. More formally:

Lemma 12. *Let S and T be finite subsets of $\mathbb{Z}^2$. Say that $S \leq T$ if for every $(i, j) \in S$, there exists $(i', j') \in T$ such that $i \leq i' \wedge j \leq j'$ or $i \leq j' \wedge j \leq i'$.*

Let $\mathcal{F} = \langle \mathcal{F}^{i \times j}_{\text{EXCH}} \mid (i, j) \in S \rangle$ and $\mathcal{G} = \langle \mathcal{F}^{i \times j}_{\text{EXCH}} \mid (i, j) \in T \rangle$. Then if $S \leq T$, then $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, and if $S \not\leq T$, then $\mathcal{F} \sqsubseteq \mathcal{G}$ is equivalent to the sh-OT *assumption.*

The proof is a straight-forward generalization of our characterization for SFE. Since every exchange-like DFF is equivalent to such a collection of exchanges, this establishes the main result for DFFs as well.

When $\mathcal{F}$ is not exchange-like.

We describe how any non-exchange-like functionality can be used to unconditionally realize $\mathcal{F}_{\text{COM}}$. The argument here is reproduced directly from [MPR09A, ROS09] with minimal modification (corresponding to our looser definition of simple states).

In [MPR09A, ROS09], the following property about dominating inputs is given:

Lemma 13 ([MPR09A, ROS09])**.** *Let $\mathcal{F}$ be a DFF. Then there is an environment $\mathcal{Z}_0$ with the following properties:*

- *$\mathcal{Z}_0$ sends a constant number of inputs to $\mathcal{F}$,*
- *$\mathcal{Z}_0$ choses inputs for both parties at random,*
- *$\mathcal{Z}_0$ always outputs 1 when interacting with two parties running the dummy protocol on an instance of $\mathcal{F}$,*
- *For every $x, x' \in X$, if $x \not\geq_A x'$, then $\mathcal{Z}_0$ has a constant probability of outputting 0 when interacting with an Alice protocol that sends x instead of x' in the first round.*

Using this fact, we obtain the following claim, used in the proof of main theorem:

Lemma 14. *If a non-simple state in $\mathcal{F}$ is reachable via a sequence of safe transitions from $\mathcal{F}$'s start state, then either $\mathcal{F}_{\text{COM}} \sqsubseteq \mathcal{F}$ or $\mathcal{F}_{\text{OT}} \sqsubseteq \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq \mathcal{F}$.*

Proof. Without loss of generality (by Lemma 10) we assume that the start state of $\mathcal{F}$ is non-simple.

First, suppose the start state q_0 of $\mathcal{F}$ is non-simple because its input/output behavior in the first round is not an exchange function. Then in the $\mathcal{F}$-hybrid setting we can easily securely realize the SFE functionality $\mathcal{G} = (f_A(q_0, \cdot, \cdot), f_B(q_0, \cdot, \cdot))$, by the simple dummy protocol. Even though $\mathcal{F}$ may keep in its memory arbitrary information about the first-round inputs, the information can never be accessed since honest parties never send inputs to $\mathcal{F}$ after its first round, and $\mathcal{F}$ waits for inputs from both parties before giving any output. Thus $\mathcal{G} \sqsubseteq \mathcal{F}$. By Lemma 1, we have that either $\mathcal{F}_{\text{OT}} \sqsubseteq \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq \mathcal{F}$ in this case.

Otherwise, assume that the input/output behavior in the first round is an exchange function SFE, and that q_0 is non-simple for one of the other reasons in the definition of simple states. The two cases are symmetric, and we present the case where Alice can commit to Bob. Suppose there are Alice inputs $x_0^*, x_1^* \in X$ such that $f_B(q_0, x_0^*, \cdot) \equiv f_B(q_0, x_1^*, \cdot)$, but for all $x \in X$, either $x \not\geq_A x_0^*$ or $x \not\geq_A x_1^*$. Intuitively, this means that $\mathcal{F}$ *binds* Alice to her choice between inputs x_0^* and x_1^* — there are behaviors of $\mathcal{F}$ possible when her first input is x_b^*, which are not possible when her first input is x_{1-b}^*. We formalize this intuition by using the first input round of $\mathcal{F}$ to let Alice commit a bit to Bob.

Recall the "complete" environment $\mathcal{Z}_0$ from Lemma 13, and suppose it runs for m rounds and has a distinguishing probability $p > 0$. Our protocol for $\mathcal{F}_{\text{COM}}$ is to instantiate $N = 2\lceil \log_{1-p} 0.5 \rceil \kappa = \Theta(\kappa)$ independent instances of $\mathcal{F}$, where κ is the security parameter. We will write $\mathcal{F}_i$ to refer to the ith instance of $\mathcal{F}$. The protocol is as follows:

1. (Commit phase, on Alice input (COMMIT, b), where $b \in \{0, 1\}$) Alice sends x_b^* to each $\mathcal{F}_i$. For each i, Bob sends a random $y_{i1} \in Y$ to $\mathcal{F}_i$ and waits for output $f_B(q_0, y_{i1}, x_0^*) = f_B(q_0, y_{i1}, x_1^*)$. If he receives a different input, he aborts. Otherwise, he outputs COMMITTED.
2. (Reveal phase, on Alice input REVEAL) Alice sends b to Bob. For each i, Alice sends her input/output view of $\mathcal{F}_i$ to Bob (x_b^* and the first-round response from $\mathcal{F}_i$). If any of these reported views involve Alice sending something other than x_b^* to $\mathcal{F}_i$, then Bob aborts. Otherwise, Bob sets $x_{i1} = x_b^*$ for all i.
3. For $j = 2$ to m:
   (a) Bob sends Alice a randomly chosen $x_{ij} \in X$. Alice sends x_{ij} to $\mathcal{F}_i$.
   (b) Bob sends a randomly chosen input $y_{ij} \in Y$ to $\mathcal{F}_i$.
   (c) For each i, Alice reports to Bob her output from $\mathcal{F}_i$ in this round.
4. If for any i, Alice's reported view or Bob's outputs from $\mathcal{F}_i$ does not match the (deterministic) behavior of $\mathcal{F}$ on input sequence $(x_{i1}, y_{i1}), (x_{i2}, y_{i2}), \ldots$, then Bob aborts. Otherwise, he outputs (REVEAL, b).

When Bob is corrupt, the simulation is to do the following for each i: When Bob sends y_{i1} to $\mathcal{F}$ in the commit phase, simulate $\mathcal{F}_i$'s response as $f_B(q_0, x_0^*, y_{i1}) = f_B(q_0, x_1^*, y_{i1})$. In the reveal phase, to open to a bit b, simulate that Alice sent Bob x_b^* and the view that is consistent with that input: $f_A(q_0, x_b^*, y_{i1})$. Maintain the corresponding state q_i of $\mathcal{F}_i$ after seeing inputs (x_b^*, y_{i1}). Then when Bob sends x_{ij} to Alice and y_{ij} to $\mathcal{F}_i$, simulate that $\mathcal{F}_i$ gave the correct output to Bob and that Alice reported back the correct output from $\mathcal{F}_i$ that is consistent with $\mathcal{F}$ receiving inputs x_{ij}, y_{ij} in state q_i. Each time, also update the state q_i according to those inputs. It is clear that the simulation is perfect.

When Alice is corrupt, the simulation is as follows: The simulator faithfully simulates each instance of $\mathcal{F}$ and the behavior of an honest Bob. If at any point, the simulated Bob aborts, then the simulation aborts. Suppose Alice sends $\tilde{x}_{i1}$ to each $\mathcal{F}_i$ in the commit phase, and that the simulation has not aborted at the end of the commit phase. If the majority of $\tilde{x}_{i1}$ values satisfy $\tilde{x}_{i1} \geq_A x_0^*$, then the simulator sends $(\text{COMMIT}, 0)$ to $\mathcal{F}_{\text{COM}}$; otherwise it sends $(\text{COMMIT}, 1)$. Note that by the properties of $\mathcal{F}$, each $\tilde{x}_{i1}$ cannot dominate both x_0^* and x_1^*. Let b be the bit that the simulator sent to $\mathcal{F}_{\text{COM}}$.

If the simulated Bob ever outputs (REVEAL, b), then the simulator sends REVEAL to $\mathcal{F}_{\text{COM}}$. The simulation is perfect except for the case where the simulated Bob outputs $(\text{REVEAL}, 1-b)$ (in this case, the real world interaction ends with Bob outputting $(\text{REVEAL}, 1-b)$, while the ideal world interaction aborts). We show that this event happens with negligible probability, and thus our overall simulation is statistically sound.

Suppose Alice sends $b' = 1 - b$ at the beginning of the reveal phase. Say that an instance $\mathcal{F}_i$ is *bad* if $\tilde{x}_{i1} \not\geq_A x_{1-b}^*$. Note that at least half of the instances of $\mathcal{F}_i$ are bad. When an instance $\mathcal{F}_i$ is bad, $\mathcal{Z}_0$ can distinguish with probability at least p between the cases of $\mathcal{F}$ receiving first input $\tilde{x}_{i1}$ and x_{1-b}^* from Alice. However, in each instance of $\mathcal{F}_i$, Bob is sending random inputs to Alice (who sent $\tilde{x}_{i1}$ as the first input to $\mathcal{F}_i$), sending random inputs himself to $\mathcal{F}_i$, obtaining his own output and Alice's reported output from $\mathcal{F}_i$ in an on-line fashion, and comparing the result to the known behavior of $\mathcal{F}$ (when x_{1-b}^* is the first input of Alice). This is exactly what $\mathcal{Z}_0$ does in the definition of $\tilde{x}_{i1} \geq_A x_{1-b}^*$, so Bob will detect an error with probability p in each bad instance. In the real world, Bob would accept in this reveal phase with probability at most $(1-p)^{-N/2} \leq 2^{-\kappa}$, which is negligible as desired. □

Lemma 15. *If $\mathcal{G}$ is exchange-like and non-trivial, and $\mathcal{F}$ is not exchange-like, then $\mathcal{F} \sqsubseteq \mathcal{G}$ is equivalent to the* sh-OT *assumption.*

Proof. From [MPR09A], we have that $\mathcal{G}$ is $\sqsubseteq$-complete given the sh-OT assumption, and thus $\mathcal{F} \sqsubseteq \mathcal{G}$. The main challenge is proving that $\mathcal{F} \sqsubseteq \mathcal{G}$ implies the sh-OT assumption.

Since $\mathcal{F}$ is not exchange-like, then either $\mathcal{F}_{\text{OT}} \sqsubseteq \mathcal{F}$, $\mathcal{F}_{\text{CC}} \sqsubseteq \mathcal{F}$, or $\mathcal{F}_{\text{COM}} \sqsubseteq \mathcal{F}$. Thus, by the universal composition theorem we assume that we have a secure protocol for either $\mathcal{F}_{\text{CC}}$, $\mathcal{F}_{\text{OT}}$, or $\mathcal{F}_{\text{COM}}$ using $\mathcal{G}$. The cases involving $\mathcal{F}_{\text{CC}}$ and $\mathcal{F}_{\text{OT}}$ have been addressed already in the proof of the characterization for non-reactive exchange-like functionalities.

Thus we describe the case where $\mathcal{F}_{\text{COM}} \sqsubseteq \mathcal{G}$ via protocol π. It is similar to the proof of the case involving $\mathcal{F}_{\text{CC}}$ for non-reactive functionalities. Without loss of generality, we assume that $\mathcal{G}$ is simply a collection of exchange functions, as in Lemma 9. The semi-honest protocol for $\mathcal{F}_{\text{OT}}$ is as follows, with Alice the sender (having inputs x_0, x_1) and Bob the receiver (having input b):

- The parties instantiate two parallel instances of π, with Alice acting as the sender. Since there is no access to an external $\mathcal{G}$, Bob will simulate Alice's interface with instances of $\mathcal{G}$— that is, Alice will send her $\mathcal{G}$ inputs directly to Bob, and he will give simulated responses from instances of $\mathcal{G}$. Alice commits to bit x_0 in the first instance, and x_1 in the second instance. Alice honestly runs π and halts after the commitment phase finishes.
- In protocol instance $(1-b)$, Bob carries out the simulation of $\mathcal{G}$-instances and the π protocol completely honestly.
- In protocol instance b, Bob honestly runs the UC simulator for π, treating Alice as the adversary (including simulating Alice's interface with $\mathcal{G}$-instances). At the end of the commitment phase, the simulator extracts Alice's bit x_b, which Bob outputs.

By the UC security of π, Alice's view is computationally independent of b (i.e., she cannot distinguish an interaction with π's simulator from an interaction in which the receiver and $\mathcal{G}$ are honest). Bob correctly learns x_b, and we must argue that he has no advantage guessing x_{1-b}. If all $\mathcal{G}$-instances were external to the $(1-b)$ interaction as ideal functionalities, then the security of π would imply that Bob has no advantage in guessing x_{1-b} after the commitment phase. Being a collection of exchange functions, $\mathcal{G}$ has the property that Bob always learns all of Alice's inputs. Thus Alice can send her $\mathcal{G}$-inputs directly to Bob, without loss of generality. This is exactly what happens in the $(1-b)$ interaction. $\square$

D Oblivious Transfer Amplification

We first establish the following convenient technical lemma:

Lemma 16 (Noisy Channel Bounds). *Consider a noisy channel $\mathcal{C}$, which either forwards an input element $x \in \mathbb{Z}_N$ unchanged with probability q, and otherwise replaces it uniformly chosen element from $\mathbb{Z}_N \setminus \{x\}$.*

Suppose a string $s = s_1 \dots s_k \in \mathbb{Z}_N^k$ is passed through $\mathcal{C}$, and $t = t_1 \dots t_k$ is the result. Then the probability that $\sum_{i=1}^k t_i = \sum_{i=1}^k s_i$ is at most

$$\frac{1}{N} + \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right)$$

.

Proof. Without loss of generality, suppose that $\sum_{i=1}^k s_i = 0$. Consider the following polynomial:

$$f(x) = \left(q + \frac{1-q}{N-1}x + \dots \frac{1-q}{N-1}x^{N-1}\right)^k$$

Observe that the probability that $\sum_{i=1}^k t_i = 0$ is given by the following expression:

$$\sum_{\lambda \in \mathbb{Z}} [x^{\lambda N}] f(x) = \frac{\sum_{i=0}^{n-1} f(\omega^i)}{N},$$

where $1, \omega, \dots, \omega^{N-1}$ are distinct roots of $z^N = 1$. We can evaluate the expression in the following manner:

$$\begin{aligned}
\frac{1}{N}\sum_{i=0}^{N-1} f(\omega^i) &= \frac{1}{N}\sum_{i=0}^{N-1}\left(\frac{Nq-1}{N-1} + \frac{1-q}{N-1}\sum_{j=0}^{N-1}\omega^{ij}\right)^k \\
&= \frac{1}{N} + \frac{(N-1)\left(\frac{Nq-1}{N-1}\right)^k}{N} \\
&= \frac{1}{N} + \left(1 - \frac{1}{N}\right)\left(1 - \frac{1-q}{N-1}\right)^k \\
&\le \frac{1}{N} + \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right)
\end{aligned}$$ $\square$

We now define our variant of a weak Oblivious Transfer (OT) and how it can be amplified to obtain the conventional one-out-of-two OT.

Similar to the definition of (p,q)-OT used in [DKS99], we introduce a notion of a weak OT.

Definition 11 (q-weak-OT). *A q-weak-OT is a protocol that satisfies the following conditions:*

- *The sender has inputs $(x_0, x_1) \in \mathbb{Z}_N^2$. The receiver has input $b \in \{0,1\}$ to the functionality and receives x_b as output.*
- *A passively corrupt sender has no advantage in guessing the bit b.*
- *No passively corrupt receiver can guess x_{1-b} with probability greater than q, when the sender's inputs are random.*

Thus, $\frac{1}{N}$-weak-OT is a standard OT with sender input set $\mathbb{Z}_N$.

We can amplify a q-weak-OT using an algorithm taken from [DKS99].

Definition 12 (R-Reduce). *R-Reduce$(k, \mathcal{W})$ is defined as the following protocol, where $\mathcal{W}$ is a weak-OT.*

1. *Let $(x_0, x_1) \in \mathbb{Z}_N^2$ be the input of the sender; and $b \in \{0,1\}$ be the input of the receiver.*
2. *The sender generates random $(x_{0i}, x_{1i}) \in \mathbb{Z}_N^2$, for $i \in [k]$. Let $r_0 = \sum_{i=1}^k x_{0i}$ and $r_1 = \sum_{i=1}^k x_{1i}$. The sender sends $z_0 = x_0 + r_0$ and $z_1 = x_1 + r_1$ to the receiver*
3. *Both parties execute $\mathcal{W}$, k times with input $(x_{0i}, x_{1i}) \in \mathbb{Z}_N^2$ for the sender and input b for the receiver.*
4. *The receiver outputs $x_b = z_b - (\sum_{i=1}^k x_{b,i})$.*

Lemma 17. *If $\mathcal{W}$ is a q-weak-OT, then R-Reduce$(k, \mathcal{W})$ is a $(\frac{1}{N} + \nu(q,k))$-weak-OT, where:*

$$\nu(q,k) \leq \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right)$$

Proof. We consider the probability that the receiver can successfully guess x_{1-b}. Let $s = s_1 \ldots s_k \in \mathbb{Z}_N^k$ be chosen uniformly at random.Suppose we are given a string $t_1 \ldots t_k \in \mathbb{Z}_N^k$ which has the property that $t_i = s_i$ with probability q. Observe that if t_i is wrong, it adds an error $s_i - t_i$ which is uniformly random over $\mathbb{Z}_N$. So, in general with probability q it either adds 0 error; or adds a random error from the set $\mathbb{Z}_N \setminus \{0\}$ with probability $(1-q)/(N-1)$. Then, using Lemma 16, the probability that $\sum_{i=1}^k s_i = \sum_{i=1}^k t_i$ is at most:

$$\frac{1}{N} + \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right) \qquad \square$$

Thus, if $q \leq 1 - \frac{1}{\text{poly}(\kappa)}$, then R-Reduce$(\kappa/(1-q), \mathcal{W})$ is a full-fledged 1-out-of-2 OT protocol.

E Expanding the Framework

The framework that we have presented is quite general thanks to the general nature of functionalities. However, there are several dimensions in which this framework can be extended, leading to possibly further computational complexity assumptions to appear. We mention a few such extensions below.

Using Infinite Domain/Memory Functionalities. In this paper, we have confined ourselves to "finite" functionalities (which are finite state machines with finite input/output alphabets). One could instead consider the more general class, with infinite alphabets and/or infinite state space. On the positive side, this allows modeling "object-oriented" cryptography which involves such concepts as encryption and signatures (as opposed to "service-oriented" cryptography which is more natural in the setting of multi-party computation). Note that currently, the sh-OT assumption and the assumption that a key-agreement protocol exists, do not specify the number of rounds of these protocols. Indeed a 2-round key-agreement protocol *is* a public-key encryption scheme and a 2-round sh-OT protocol is a computational version of the so-called dual-mode encryption scheme.[9]

On the other hand, this considerably complicates things: for instance, Proposition 1 does not hold any more. [MPR09A] points out an (infinite domain) functionality $\mathcal{G}$ such that $\mathcal{G}$ is not trivial, but say $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{PPT}} \mathcal{G}$ does not hold under the sh-OT assumption. It is likely that several intermediate assumptions, like in the hierarchy of assumptions presented in [GKM$^+$00] will manifest in this framework. A similar complication was encountered in [HNRR06]. We leave it is an interesting challenge to extend the framework to include infinite functionalities, but without sacrificing the economy and conceptual clarity of the set of complexity assumptions produced by the framework.

Using Alternate Reductions. If we use a different notion of reduction in place of $\sqsubseteq_{\text{PPT}}$, the assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ will change its meaning. While it is not clear if any reasonable notion of reduction will give a *larger set of assumptions* when all pairs $(\mathcal{F}, \mathcal{G})$ are considered, it certainly is true that for specific pairs $(\mathcal{F}, \mathcal{G})$ the assumption becomes different. The variants one could consider weaker notions of security like security against passive (semi-honest) or standalone adversaries, or stronger notions of security like simultaneous security against passive and active adversaries, or against adaptive adversaries. One could also consider tighter non-standard reduction notions derived by imposing constraints on the protocols carrying out the reductions, like constant round complexity, for instance. Another example is to restrict to protocols which use a given functionality in only one direction (with say, Alice and Bob in the protocol playing fixed roles, say sender and receiver respectively, when interacting with the given functionality).

It will indeed be interesting if a computational complexity assumption manifests when using some other (meaningful) notions of reduction, but not the one we use.

More Functionalities. In this work, we restricted ourselves to a large, but restricted class of functionalities. In particular, our functionalities are not "fair": they al-

[9] Here, a dual-mode encryption scheme is a public-key encryption scheme, in which there is an alternate mode to generate a public-key (which remains indistinguishable from a key generated in the normal mode) such that the semantic security of the encryption is retained even given the randomness used in key generation. Dual mode encryption introduced in [PVW08, KN08] is stronger in that the alternate mode is required to result in a public-key such that a ciphertext produced using that key is statistically independent of the message.

low the adversary to learn the output and then decide whether to deliver the output or not, to the other party. Another class we considered only briefly is that of randomized functionalities. These classes of functionalities are understood only in bits and pieces. A systematic study of their cryptographic complexities remains an open problem.

Special Pairs. Finally, we mention that when considering some of the above extensions, it might be interesting to consider (only) special pairs of functionalities, where the reduction may have extra meaning. For instance, when restricting protocols to use a given 2-party functionality $\mathcal{F}$ in only one direction, it is particularly interesting to consider reducing the functionality $\mathcal{F}^{-1}$ which reverses the roles of its parties. Another interesting question is of "parallel repetition" which considers reducing $\mathcal{F}^t$, a synchronous repetition of t copies of $\mathcal{F}$, to the functionality $\mathcal{F}$ (wherein the protocol is allowed to use multiple asynchronous copies of $\mathcal{F}$). For instance, $\mathcal{F}_{\text{COIN}}^t$ reduces to $\mathcal{F}_{\text{COIN}}$ unconditionally, but by Theorem 1 we know that a parallel repetition of $\mathcal{F}_{\text{EXCH}}^{2,2}$ reduces to $\mathcal{F}_{\text{EXCH}}^{2,2}$ iff the $\mathsf{sh\text{-}OT}$ assumption holds.