Complexity of Multiparty Computation Problems: The Case of 2-Party Symmetric Secure Function Evaluation

Hemanta Maji* Manoj Prabhakaran* Mike Rosulek*

October 22, 2008

Abstract

In symmetric secure function evaluation (SSFE), Alice has an input x, Bob has an input y, and both parties wish to securely compute $f(x, y)$. We classify these functions f according to their "cryptographic complexities," and show that the landscape of complexity among these functions is surprisingly rich.

We give combinatorial characterizations of the SSFE functions f that have passive-secure protocols, and those which are protocols secure in the standalone setting. With respect to universally composable security (for unbounded parties), we show that there is an infinite hierarchy of increasing complexity for SSFE functions, That is, we describe a family of SSFE functions $f_1, f_2, \ldots$ such that there exists a UC-secure protocol for f_i in the f_j-hybrid world if and only if $i \leq j$.

Our main technical tool for deriving complexity separations is a powerful protocol simulation theorem which states that, even in the strict setting of UC security, the canonical protocol for f is as secure as any other protocol for f, as long as f satisfies a certain combinatorial characterization. We can then show intuitively clear impossibility results by establishing the combinatorial properties of f and then describing attacks against the very simple canonical protocols, which by extension are also feasible attacks against *any* protocol for the same functionality.

*Department of Computer Science, University of Illinois, Urbana-Champaign. {hmaji2,mmp,rosulek}@uiuc.edu.
Partially supported by NSF grant CNS 07-47027.

1 Introduction

In the classical setting of secure multiparty computation, Alice and Bob have private inputs x and y respectively, and they want to to jointly compute a common value $f(x, y)$ in a secure way. Starting from Yao's millionaire's problem [10], such *symmetric secure function evaluation* (SSFE) problems have remained the most widely studied multiparty computation problems. A SSFE problem is fully specified by the function table of f (i.e., a matrix M with $M_{x,y} = f(x, y)$); studying this matrix can tell us everything about the corresponding SSFE problem. Despite this apparent simplicity, and several works carefully exploring SSFE problems, the landscape of such problems has remained far from complete.

In this work we uncover a rich structure and delineate several distinct classes of functions with varying levels of "cryptographic complexity." To make the most fine-grained complexity distinctions among SSFE functions, we deploy the lens of the strongest security notion for multiparty computation: universally composable (UC) security [3] in computationally unbounded environments against active adversaries. We use this security notion to define our notion of reduction between functions, which in turn forms the basis of our complexity classifications. Under less stringent notions of security many of the separations we draw disappear. Nevertheless, our techniques and our results draw liberally from the more classical security notions, namely, security in standalone environments, and security against passive adversaries.

Kilian [6] characterized the SSFE functions which are complete, as those functions that contain a generalized "OR-minor." On the other end of the spectrum, it immediately follows from characterization of realizable SFE functions in [4, 9] that the only realizable SSFE functions in the UC framework are those which depend on only one party's input. The vast region between these two extreme levels of complexity is the subject of our investigation.[1]

1.1 Our Results

Security against passive adversaries. The oldest and most widely studied model for SSFE is security against passive adversaries (honest-but-curious security). Beaver [2] and Kushilevitz [8] independently showed that for the case of *perfect* security, there is a simple and natural combinatorial characterization for realizable SSFE functions. We prove that the same characterization also extends to the model wherein only statistical security is required (Theorem 2). Chor and Kushilevitz [5] and Kilian [7] also characterized completeness for (subclasses of) SSFE functions.

We show that the SSFE functions that have UC-secure protocols in the $\mathcal{F}_{\mathsf{com}}$-hybrid world (i.e., realizable against active adversaries in the UC framework, using the commitment functionality) are *exactly* the functions which have passive-secure protocols (Theorem 3).

Protocol Simulation. The characterization of Beaver and Kushilevitz gives very simple deterministic protocols for SSFE functions f, which we call *canonical protocols.* These protocols closely follow the structure required in f by the combinatorial characterization. Our extension of this characterization follows the intuition that *any* secure protocol for f must reveal information about inputs in the same order as the canonical protocol for f.

[1] Interestingly, in contrast, for *one-sided output* SFE (a.k.a *asymmetric SFE* in which only one party receives the output), such a region does not exist: for asymmetric SFE every function either is realizable or is complete for secure MPC. This follows from the characterizations in [7] and [9].

In Theorem 1, which is our main technical tool for deriving complexity separations, we formally express this intuitive connection between protocols for f and the steps of the canonical protocol. We show that, for a natural class of SSFE functions f, and any secure protocol π for f, the canonical protocol for f is "as secure as" π. That is, any feasible attack on the canonical protocol is also a feasible attack against π. Furthermore, this equivalence holds with respect to standalone as well as UC adversaries. Using Theorem 1, we are able to show separations among functionalities by simply describing attacks against canonical protocols.

Security against active adversaries in standalone environments. Unsurprisingly, the structure of SSFE functions is not as clear-cut in the standalone setting. Of particular interest is characterizing the functions which are standalone secure but not UC-secure. An interesting instance of a function which is securely realizable in standalone environments, but not in general environments, was pointed out by Backes et al. [1] However, the problem of identifying all such functions has remained open.

We give an explicit (and simple) combinatorial characterization of the functions which have standalone-secure protocols (Theorem 4). To show that certain functions do not have standalone-secure protocols, we crucially use the protocol simulation theorem Theorem 1, and show a simple attack on the corresponding canonical protocols.

We also observe that for every function that is realizable against standalone environments but not general environments, there is a "concurrent attack" where a standalone environment runs two sessions of the protocol with same roles for the two parties in both sessions. From Theorem 1 and our characterization of functions with standalone-secure protocols, it is easy to see that any such protocol will suffer from the same concurrent attack as the canonical protocol. This answers an open question from [1].

Separations in UC model. We further apply Theorem 1 to the setting of protocols that use ideal UC functionalities. We describe a combinatorial criterion of two functions f, g which implies that there is no UC-secure protocol for f that uses calls to an ideal functionality for g. (Theorem 5). Again, the result is obtained simply by describing a particular kind of attack on simple canonical protocols.

We apply these separations to particular functions f to obtain a strict, infinite hierarchy of complexities among SSFE functions. That is, we describe a sequence of functionalities $f_1, f_2, \ldots$ such that there is a secure protocol for f_i using calls to f_{i+1}, but not vice-versa. Since these functions f_i are standalone secure and passive-secure, this hierarchy also implies that there is no complete function for the class of SSFE functions with standalone-secure protocols, or the class of functions with passive-secure protocols (Corollary 7).

Our separation results also demonstrate functionalities of incomparable complexities. A function which has a passive-secure protocol but no standalone-secure protocol cannot be implemented using any function with standalone-secure protocol, since the class of standalone security is closed under this operation. Conversely, Theorem 5 shows that in many instances, there is no secure protocol for certain standalone-secure functions function using certain passive-secure functions.

2 Preliminaries

In this section we present some standard notions, and also introduce some concepts and definitions used in our constructions and proofs.

MPC Problems and Secure Realization. Following the Universal Composition [3] framework, a multi-party computation problem is defined by the code of a trusted (probabilistic, interactive) entity called a *functionality*. A protocol π is said to securely realize an MPC functionality $\mathcal{F}$, if for all (static) active adversaries, there exists a simulator such that for all environments, $|\Pr[\text{EXEC}_{\mathcal{F}} = 1] - \Pr[\text{EXEC}_{\pi}]| \leq \epsilon(k)$, for some negligible function ϵ. Here k is the *security parameter* that is an input to the protocol and simulator. $\text{EXEC}_{\mathcal{F}}$ denotes the environment's output distribution in the "ideal" execution: involving $\mathcal{F}$, the environment, and the simulator; EXEC_{π} denotes the environment's output in the "real" execution: involving an instance of the protocol, the environment, and the adversary. Throughout this paper, we consider a UC model in which all entities are computationally unbounded.

We also consider *hybrid worlds*, in which protocols may also have access to a particular ideal functionality. In a $\mathcal{G}$-hybrid world, parties running the protocol π can invoke up any number of independent, asynchronous instances of $\mathcal{G}$. Regular protocols could be considered to use the (authenticated) communication functionality. If a protocol π securely realizes a functionality $\mathcal{F}$ in the $\mathcal{G}$-hybrid world, we shall write $\mathcal{F} \sqsubseteq \mathcal{G}$. The $\sqsubseteq$ relation is transitive and reflexive, and it provides our basis for comparing the complexity of various functionalities.

We also consider two common restrictions of the security model, which will prove useful in our results. If we restrict the security definition to environments which do not interact with the adversary during the protocol execution, we get a weaker notion of security, called *standalone security.* If we restrict to adversaries and simulators which receive an input from the environment and run the protocol honestly, we get a different relaxation of security called *passive security.* Note that in this definition, simulators must behave honestly in the ideal world, which means they simply pass the input directly to the functionality.

Symmetric SFE Functionalities. In this paper we focus exclusively on classifying 2-party *symmetric secure function evaluation* (SSFE) functionalities. A SSFE functionality $\mathcal{F}_f$ is parameterized by a function $f : X \times Y \to Z$, where X, Y, Z are finite sets. The functionality $\mathcal{F}_f$ simply receives the inputs from the two parties, computes f on the inputs and sends the result to both the parties. However, as is standard, we also allow the adversary to first receive the outcome of the function evaluation and then block output delivery if desires (see Figure 1). For convenience, we shall often write f instead of $\mathcal{F}_f$.

- Wait for input x from Alice and input y from Bob.
- When both x and y have been received, send $(\text{OUTPUT}, f(x, y))$ to the adversary.
- On input DELIVER from the adversary, if both x and y have already been received, send $f(x, y)$ to both Alice and Bob.

Figure 1: Functionality $\mathcal{F}_f$: Symmetric Secure Function Evaluation of f with abort

2.1 Structure of Functions and Protocols

We say that two functions are isomorphic if each one can be computed using a single call to the other (after local processing of the inputs and output). That is, Alice and Bob can independently map their inputs for f function to inputs for g, carryout the computation for g, and then locally (using their private inputs) map the output of g to the output of f (and vice-versa).

Definition 1 (Function Isomorphism, $\cong$)**.** *Let $f : X \times Y \to D$ and $g : X' \times Y' \to D'$ be two functions. Suppose there exists functions $I_A : X \to X'$, $I_B : Y \to Y'$, $M_A : X \times D' \to D$ and $M_B : Y \times D' \to D$, such that for any $x \in X$ and $y \in Y$ $f(x, y) = M_A(x, g(I_A(x), I_B(y))) = M_B(y, g(I_A(x), I_B(y)))$; then we say $f \leq g$. If $f \leq g$ and $g \leq f$ then $f \cong g$ (f is isomorphic to g).*

For example, the functions $\begin{array}{|c|c|}\hline 0 & 1 \\ \hline 1 & 0 \\ \hline \end{array}$ (XOR) and $\begin{array}{|c|c|}\hline 1 & 2 \\ \hline 3 & 4 \\ \hline \end{array}$ are isomorphic to each other.

Definition 2 (Decomposable)**.** *A function $f : X \times Y \to \{0,1\}^*$ 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 $f(x, y) \neq f(x'y)$; and*
- *$f\big|_{X_i \times Y}$ is either a constant function or column decomposable, where $f\big|_{X_i \times Y}$ denotes the restriction of f to the domain $X_i \times Y$.*

We define being column decomposable *symmetrically with respect to X and Y. We say that f is simply* decomposable *if it is either constant, row decomposable, or column decomposable.*

Note that our definition insists that row and column decomposition steps must alternate. We say that a function f is *uniquely decomposable* if its decomposition is unique, up to re-indexing the sets $X_1, \ldots, X_k$ ($Y_1, \ldots, Y_k$) at each level.

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 f is decomposable, then a *canonical protocol* for f is a deterministic protocol defined inductively as follows [8]:

- If f is a constant function, both parties output the output value of f, without interaction.
- If 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 $f\big|_{X_i \times Y}$.
- If 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 $f\big|_{X \times Y_i}$.

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

Normal form for protocols. For simplicity in our proofs, we will often assume that a protocol is given in the following normal form:

1. At the end of the interaction, both parties include their outputs in the transcript as their last message before terminating, so that each party's output is a function of the transcript alone (i.e., not a function of their input and random tape, etc.). Since both parties should receive the same output, this is without loss of generality.

2. Each message in the protocol is a *single bit*, and the rounds alternate between the two parties.

3. The honest protocol does not require the parties to maintain a persistent state (in particular a random tape). Instead, the protocol is simply a mapping $P : \{0,1\}^* \times \{0,1\}^* \to [0,1]$, indicating that if t is the transcript so far, and a party's input is x, then its next message is 1 with probability $P(t,x)$, and 0 with probability $1 - P(t,x)$. For computationally unbounded parties, requiring this normal form is without loss of generality, because the probabilities $P(t,x)$ fully define the behavior of the parties in the protocol.[2]

SSFE Functions are "Deviation Revealing." In [9] it is shown that for the so-called "deviation revealing" functionalities, if a protocol π is a UC-secure realization of that functionality, then the same protocol is also secure against passive adversaries. It is easy to verify that SSFE functionalities are deviation revealing. Thus:

Lemma 1 ([9])**.** *Let π be a UC-secure protocol (perhaps in a hybrid world) for a SSFE f. Then π itself is a passive-secure protocol for f as well.*

We include in Appendix A an adapted version of the argument in [9]. Note that standalone security is a restriction of UC security to a particular class of environments, and thus the claim applies also to protocols π which are only standalone-secure.

Note that Lemma 1 does not hold in general for non-symmetric SFE functionalities. For example, the SFE where Alice gets no output but Bob gets the boolean-OR of both parties' inputs is not passively realizable. However, the protocol where Alice simply sends her input to Bob is secure in the UC setting, since a malicious Bob in the ideal world can always learn her input by sending 0 as its input to the functionality.

3 Simulation of Canonical Protocol in a General Protocol

In this section, we develop our main new technical tool, the protocol simulation theorem.

Definition 3. *We say that x, x', y, y' forms a* $\boxplus$-minor (resp. $\boxplus$-minor) in f *if:*

$$\begin{array}{ccc} f(x,y) & = & f(x,y') \\ \neq & & \neq \\ f(x',y) & \neq & f(x',y') \end{array} \qquad \left(\text{resp. if } \begin{array}{ccc} f(x,y) & \neq & f(x,y') \\ = & & \neq \\ f(x',y) & \neq & f(x',y') \end{array} \right)$$

In a canonical protocol for a function that is entirely a $\boxplus$-minor, Alice must completely reveal her input before Bob reveals anything about his input. We first show that, in general, this intuition carries through for *any* protocol, with respect to *any* $\boxplus$ or $\boxplus$-minor. There must be a point during the protocol at which Alice has completely made the distinction between two of her inputs, but Bob has not made any distinction between two of his inputs.

[2] Note that because of this, security against adaptive corruption and static corruption are the same for this setting.

For the remainder of this section we fix a SSFE f with domain $X \times Y$, and secure protocol π for f.

Definition 4. *Let* $\Pr[u|x,y]$ *denote the probability that* π *generates a transcript that has* u *as a prefix, when executed honestly with* x *and* y *as inputs.*

Let F *be a set of strings that is prefix-free. Define* $\Pr[F|x,y] = \sum_{u \in F} \Pr[u|x,y]$. *We call* F *a* frontier *if* F *is maximal – that is, if* $\Pr[F|x,y] = 1$ *for all* x, y. *Define* $\mathcal{D}_F^{x,y}$ *as the probability distribution over* F, *where* $u \in F$ *is chosen with probability* $\Pr[u|x,y]$.

Lemma 2 ($\boxplus$ Frontiers)**.** *For all* $x \neq x' \in X$, *there is a frontier* F *such that, for all* $y, y' \in Y$*:*

- *if* $f(x,y) \neq f(x',y)$, *then* $SD\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x',y}\Big) \geq 1 - \nu$, *and*
- *if* x, x', y, y' *form a* $\boxplus$*-minor, then* $SD\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x,y'}\Big), SD\Big(\mathcal{D}_F^{x',y}, \mathcal{D}_F^{x',y'}\Big) \leq \nu$.

where ν *is a negligible function of the security parameter.*

Proof. Suppose we have a protocol π in normal form, and let ϵ denote the simulation error of π (i.e., the statistical difference between real- and ideal-world outputs). The next message function depends only on the appropriate party's input and the transcript so far. For $b \in \{0,1\}$ and $u \in \{0,1\}^*$, denote by $\pi_A(ub, x)$ is the probability that Alice's next message is b when running on input x and the transcript so far is u. Define π_B for Bob analogously. Suppose we run π on inputs x, y, where Alice sends the first message The probability of obtaining a particular transcript prefix $u = u_1 \cdots u_n$ is (say n is even):

$$\begin{aligned}\Pr[u|x,y] &= \pi_A(u_1, x) \cdot \pi_B(u_1u_2, y) \cdot \pi_A(u_1u_2u_3, x) \cdots \pi_B(u_1 \cdots u_n, y) \\ &= \left(\prod_{i=1}^{n/2} \pi_A(u_1 \cdots u_{2i-1}, x)\right)\left(\prod_{i=1}^{n/2} \pi_B(u_1 \cdots u_{2i}, y)\right) = \alpha(u,x)\beta(u,y)\end{aligned}$$

where $\alpha(u,x)$ and $\beta(u,y)$ are defined to be the parenthesized quantities, respectively.

Given x, x' as above, and for a fixed $\mu < 1$, to be defined later, we define the frontier F as the *prefix-minimal* elements of:

$$\{u \mid u \text{ is a complete transcript, or } \big|\alpha(u,x) - \alpha(u,x')\big| \geq \mu(\alpha(u,x) + \alpha(u,x'))\}$$

By "complete transcript", we mean one on which the parties terminate and give output.

We partition F into $F_{\text{good}} \cup F_{\text{bad}}$, where F_{good} are the elements $u \in F$ which satisfy $|\alpha(u,x) - \alpha(u,x')| \geq \mu(\alpha(u,x) + \alpha(u,x'))$. Each element $u \in F_{\text{bad}}$ therefore satisfies $\alpha(u,x)/\alpha(u,x') \leq (1+\mu)/(1-\mu)$.

Consider any y such that $f(x,y) \neq f(x',y)$. Let $A \subseteq F_{\text{bad}}$ be the transcripts in F_{bad} which do *not* induce output $f(x,y)$ (all elements of F_{bad} are complete transcripts, and the output is a function of the transcript alone). Similarly let $B \subseteq F_{\text{bad}}$ be the transcripts in F_{bad} which do *not* induce output $f(x',y)$. Thus $\Pr[A|x,y]$ and $\Pr[B|x',y]$ must be at most ϵ. Observe that $F_{\text{bad}} \subseteq A \cup B$.

Thus we have:

$$\begin{aligned}\Pr[F_{\text{good}}|x,y] &= 1 - \Pr[F_{\text{bad}}|x,y] \geq 1 - \sum_{u\in A}\alpha(u,x)\beta(u,y) - \sum_{u\in B}\alpha(u,x)\beta(u,y)\\ &\geq 1 - \left(\frac{1+\mu}{1-\mu}\right)\sum_{u\in A}\alpha(u,x)\beta(u,y) - \left(\frac{1+\mu}{1-\mu}\right)\sum_{u\in B}\alpha(u,x')\beta(u,y)\\ &= 1 - \left(\frac{1+\mu}{1-\mu}\right)\left(\Pr[A|x,y] + \Pr[B|x',y]\right) \geq 1 - 2\left(\frac{1+\mu}{1-\mu}\right)\epsilon\end{aligned}$$

The same bound holds for $\Pr[F_{\text{good}}|x',y]$. Now,

$$\begin{aligned}\text{SD}\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x',y}\Big) &= \frac{1}{2}\sum_{u\in F}\big|\alpha(u,x) - \alpha(u,x')\big|\beta(u,y) \geq \frac{1}{2}\sum_{u\in F_{\text{good}}}\mu\left(\alpha(u,x) + \alpha(u,x')\right)\beta(u,y)\\ &= \frac{\mu}{2}\Big(\Pr[F_{\text{good}}|x,y] + \Pr[F_{\text{good}}|x',y]\Big) \geq \mu\left(1 - 2\left(\frac{1+\mu}{1-\mu}\right)\epsilon\right)\end{aligned}$$

Now consider any y and y' such that x, x', y, y' form a $\boxplus$-minor. Since $f(x,y) = f(x,y')$, we must have $\text{SD}\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x,y'}\Big) \leq \epsilon$ by the security of the protocol. Let S be the set of all proper prefixes of elements F that correspond to points where Alice has just spoken. By Lemma 6 (proven in the appendix), we can express the statistical difference at F in terms of a linear combination of $\alpha(v,x)$ values for $v \in S$. Since by definition every $v \in S$ satisfies $\alpha(v,x')/\alpha(v,x) \leq (1+\mu)/(1-\mu)$, we have:

$$\begin{aligned}\text{SD}\Big(\mathcal{D}_F^{x',y}, \mathcal{D}_F^{x',y'}\Big) &= \frac{1}{2}\sum_{u\in F}\alpha(u,x')\big|\beta(u,y) - \beta(u,y')\big| = \frac{1}{2}\left(c_\pi + \sum_{v\in S}\alpha(v,x')\lambda(v,y,y')\right)\\ &\leq \frac{1}{2}\left(c_\pi + \frac{1+\mu}{1-\mu}\sum_{v\in S}\alpha(v,x)\lambda(v,y,y')\right) \leq \frac{1}{2}\left(\frac{1+\mu}{1-\mu}\right)\left(c_\pi + \sum_{v\in S}\alpha(v,x)\lambda(v,y,y')\right)\\ &= \left(\frac{1+\mu}{1-\mu}\right)\text{SD}\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x,y'}\Big) \leq \left(\frac{1+\mu}{1-\mu}\right)\epsilon\end{aligned}$$

Choosing $\mu = 1 - \sqrt{\epsilon}$ and defining $\nu = 5\sqrt{\epsilon}$, we get the desired bounds. □

Our main protocol simulation theorem applies this intuition to show that the information disclosed during the protocol must come in the same order as in the canonical protocol, as long as the canonical protocol is unique. This restriction is necessary, as different canonical protocols for the same f can admit completely different kinds of attacks (e.g., for the XOR function, depending on which party speaks first).

Theorem 1 (Protocol Simulation)**.** *If f has a unique decomposition, then for any protocol π for f, the canonical protocol for f is "as secure as" π. That is, for every adversary attacking the canonical protocol, there is an adversary attacking π which achieves the same effect.*

Proof Sketch. The approach is an inductive generalization of Lemma 2. For each step in the decomposition of $X \times Y$, say $X = X_0 \cup X_1$, we define an associated frontier. We show inductively that the transcript distribution at this frontier statistically reveals whether (in this case) Alice's

input is in X_0 or X_1, but the distribution is nearly independent of any further distinctions among either party's inputs within $X \times Y$. Furthermore, these frontiers are visited in the natural order, with overwhelming probability.

Given such frontiers, the required simulation is fairly natural. The simulator's job is to simulate the canonical protocol to $\mathcal{A}$, while interacting with the honest party in the protocol π. The simulator $\mathcal{S}$ simply keeps track of the current step in the decomposition of f, and runs π honestly with any representative input. Whenever it reaches the next frontier for a decomposition step by the honest party, it does the following: By the definition of the frontier it can determine with great certainty which part of the decomposition the honest party's input belongs to, and simulate the next step in the canonical protocol. Then the simulator receives the adversary's next move in the canonical protocol and changes its own input for π accordingly, if necessary. By the definition of the frontier, the transcripts so far are nearly independent of such distinctions among the adversary's input, so it is indeed possible to "hot-swap" inputs to π at this point and maintain a sound simulation.

The full details of the proof are given in Appendix A.3. □

4 Characterizing Passive Security

Beaver [2] and Kushilevitz [8] showed that f is decomposable if and only if f has a perfectly secure protocol against passive adversaries. Using Lemma 2, we can generalize this result to the case where only statistical security is required:

Theorem 2. *f is decomposable if and only if f has a (statistically secure) passive-secure protocol.*

Proof Sketch. The full proof follows in Appendix A.4. At an intuitive level, our proof approach generalizes Beaver's. We use $\boxplus$ and $\boxplus$-minors to identify the first points in the protocol at which the parties reveal significant information about their inputs. We obtain a contradiction by showing that neither party can safely be the first to reveal information. □

- On input (COMMIT, x) from party P_1, send COMMITTED to party P_2, and remember x.
- On input REVEAL from party P_1, if x has already been recorded, send x to party P_2.

Figure 2: Commitment functionality $\mathcal{F}_{\mathsf{com}}$

Theorem 3. *f has a passive-secure protocol if and only if f has a UC-secure protocol in the $\mathcal{F}_{\mathsf{com}}$-hybrid world, where $\mathcal{F}_{\mathsf{com}}$ denotes the commitment functionality defined in Figure 2.*

Proof Sketch. ($\Leftarrow$) By Lemma 1, any UC-secure protocol π for a symmetric SFE f is also passively secure. There is a trivial passive-secure protocol for $\mathcal{F}_{\mathsf{com}}$ (the committing party sends "COMMITTED" in the commit phase and sends the correct value in the reveal phase). We can compose π with the passive-secure $\mathcal{F}_{\mathsf{com}}$ protocol to obtain a passive-secure protocol for f in the plain model.

($\Rightarrow$) We will give a general-purpose "compiler" from passive-secure protocols to the UC-secure $\mathcal{F}_{\mathsf{com}}$-hybrid protocols. Suppose π is a passive-secure protocol for f, in normal form. For simplicity, assume for now that π is deterministic. We will describe later the necessary modifications for randomized protocols. The complete protocol is described in Figure 3, and proof of security is given in Appendix A.6. Here we sketch the main ideas behind the protocol.

Suppose Alice's input is $x \in \{1, \ldots, n\}$, and let $\chi \in \{0,1\}^n$ be the associated characteristic vector, where $\chi_i = 1$ if and only if $i = x$. Alice commits to both $\chi_{\sigma(1)}, \ldots, \chi_{\sigma(n)}$ and $\sigma(1), \ldots, \sigma(n)$ for many random permutations σ. For each pair of such commitments, Bob will (randomly) ask Alice to open either all of $\chi_{\sigma(1)}, \ldots, \chi_{\sigma(n)}$ (and verify that χ has exactly one 1) or open all $\sigma(i)$ (and verify that σ is a permutation). Bob also commits to his own input in a similar way, with Alice giving challenges. Then both parties simulate the π protocol one step at a time. When it is Alice's turn in π, she computes the appropriate next message $b \in \{0,1\}$ and sends it. For a deterministic protocol, the next message function is a function of the transcript so far and the input. Given the partial transcript so far t, both parties can compute the set $X_b = \{x' \mid \pi(t, x') = b\}$; i.e., the set of inputs for which the protocol instructs Alice to send b at this point. Then Alice can open enough of her commitments to prove that $\chi_i = 0$ for all $i \in X_{1-b}$ to prove that the message b was consistent with her committed input and the honest protocol.

To compile a randomized protocol, we make each party commit to its random tape as well. However, security only holds if the random tape is chosen uniformly. To ensure this, we use a coin-tossing-into-the-well approach. Let S be the possible space of (input, random tape) pairs. On input x, Alice chooses a random tape r and "commits" to (x, r) by committing to many random permutations and permuted characteristic vectors. For each pair of commitments, Bob randomly asks Alice to completely open one or the other. Then Bob chooses another random tape s (for Alice) and sends it to Alice. Now Alice runs each step of the protocol on x with random tape $r \oplus s$. Both parties know s, so at each move Alice opens the commitments corresponding to the (x', r') pairs such that the protocol prescribes the opposite output on input x' and random tape $r' \oplus s$. $\square$

We note that this $\mathcal{F}_{\mathsf{com}}$ "compiler" works for any passive-secure protocol. In particular, it does not use the fact that the protocol realizes a functionality with symmetric, deterministic outputs. Thus the compiler may be of more general interest.

5 Characterizing Standalone Security (Against Active Adversaries)

From Theorem 2 we know that any passive-secure SSFE function is decomposable. However for many of these functions, the canonical protocol is clearly not secure against active adversaries. In this section we show that for every decomposition tree, there is exactly one function (up to isomorphism) that is standalone-secure, and such functions are in fact securely realized by the canonical protocol. Given any decomposition tree T (i.e., a tree with alternating levels of internal nodes marked as Alice nodes and Bob nodes) we shall associate a single function $f^T_{\max}$ with it (Definition 7), and obtain:

Theorem 4. *f is standalone-secure iff $f \cong f^T_{max}$ for some decomposition tree T. Further, such an f has a unique decomposition with decomposition tree T, and its canonical protocol is standalone-secure.*

To prove Theorem 4, we first show that non-uniquely decomposable functions are not standalone-secure. We can then apply Theorem 1 to show that standalone-secure functions are isomorphic to their corresponding $f^T_{\max}$.

Lemma 3. *If f has a standalone-secure protocol, then f has a unique decomposition.*

Proof Sketch. We first show in Lemma 9 that any protocol in which the parties agree on the output is susceptible to a certain kind of "attack," whereby a malicious party may be able to fix the output a certain way. We then show in Lemma 10 that when f has multiple decompositions, the "attack" described in Lemma 9 is not possible in the ideal world; thus it is a true attack against the protocol. □

Definition 5 (Tree-Strategy)**.** *Let T be a decomposition tree. V_A be the set of nodes which are marked as 'Alice nodes' and V_B be the set of nodes marked as 'Bob nodes'. S_A be a subtree of T such that every Alice node has exactly one outgoing edge and every Bob node has all its outgoing edges. S_A is called an Alice strategy. Similarly, we define Bob Strategy.*

We observe that given an Alice strategy and a Bob strategy, their intersection is a path from the root to a unique leaf of T.

Definition 6 (Input-Strategy)**.** *Let f be a two argument function on $X \times Y$ with decomposition T. Let (x, y) be any input from the domain $X \times Y$. Let $B(u)$ be the subset of the input domain corresponding to the the node $u \in V(T)$. We mark nodes $u \in V(T)$ such that $(x, y) \in B(u)$. These marked nodes form a path from the root to a leaf. Say, the path is $p^{(x,y)}$. Define $S_A^{(x)}$ as the tree obtained by the union of all $p^{(x,y)}$, for all $y \in Y$. $S_A^{(x)}$ is called an X-input strategy. Similarly, we define Y-input strategies.*

It is easy to observe that every X-input strategy is an Alice strategy and every Y-input strategy is a Bob strategy.

Lemma 4. *f has a standalone secure protocol against active adversaries if and only if f has a unique tree decomposition T, and every Alice and Bob strategy for T is, respectively, an X-input strategy and Y-input strategy.*

Proof. ($\Rightarrow$): We know that if f is standalone secure against active adversaries, then f has a unique decomposition T (Lemma 3). If π is a standalone secure protocol for f against active adversaries, then π is a secure protocol for f against passive adversaries (Lemma 1). Since f has unique tree decomposition and f has a passive secure protocol π, then for any adversary against the canonical protocol for f, there exists an adversary which will achieve the same effect in π (Theorem 1). However, there is a straightforward correspondence between Alice/Bob strategies and steps in the canonical protocol. Suppose S_B is a Bob strategy which is no corresponding a Y-input strategy. So, for any y' that the simulator sends to the functionality, there exists x' such that the output of the ideal world does not match the output in the real world. So, we have a strategy for Bob on the canonical tree T which can not be simulated in the ideal world. Hence, contradiction. Thus, every Alice and Bob strategies are X-input strategies and Y-input strategies, respectively.

($\Leftarrow$): Given the decomposition T for f, we show that the canonical protocol for f is standalone-secure. Let $\mathcal{F}_f$ be the ideal functionality for f and suppose Bob is corrupt. The simulator fixes the randomness of Bob and runs Bob for each input $x \in X$ (it is a rewinding simulator). Suppose the output corresponds to the leaf $d(x)$ in T. We construct the tree T' by taking union of all the paths from the root of T to $d(x)$ for all possible $x \in X$. T' is subgraph of some Bob strategy (say S_B). Say S_B corresponds to Y-input strategy $S_B^{(y)}$. The simulator sends y to $\mathcal{F}_f$ and receives $f(x, y)$. Using $f(x, y)$ and y, it can find some $x' \in X$ such that $f(x, y) = f(x', y)$. Using the same randomness as earlier, we simulate the protocol with input x'. □

Definition 7 (Maximal function of a decomposition)**.** *Let f be a two argument function with decomposition T. Let X be the set of all Alice strategies and Y be the set of all Bob strategies. Let D be the set of leaves of T. We will define a function $f^T_{max} : X \times Y \to D$. Let $S_A \in X$ and $S_B \in Y$. We define $f^T_{max}(S_A, S_B) = d$, where d is the leaf in the path corresponding to the intersection of S_A and S_B. f^T_{max} is called the maximal function of the decomposition T.*

Lemma 5. *f has a standalone secure protocol against active adversaries iff $f \cong f^T_{max}$ for decomposition T of f.*

Proof. ($\Rightarrow$): If f has a standalone secure protocol against active adversaries, then there exists a bijection between Alice strategies and X-input strategies and Bob strategies and Y-input strategies.

First we show that $f^T_{\max} \leq f$: We will show that there exists functions such that $f_{\max}(S_A, S_B) = M_A(x, f(I_A(S_A), I_B(S_B))) = M_B(y, f(I_A(S_A), I_B(S_B)))$. We define $I_A(S_A) = x$, where $S_A^{(x)}$ is the X-input strategy corresponding to S_A. Similarly, we define $I_B(S_B) = y$, where $S_B^{(y)}$ is the Y-input strategy corresponding to S_B. Now, we can determine the leaf in T given $f(x, y)$ and either x or y. Hence $f_{\max}(T) \leq f$.

Now we will show that $f \leq f^T_{\max}$: We will show that there exists functions such that $f(x, y) = M(f^T_{\max}(I_A(x), I_B(y)))$. We define $I_A(x) = S_A$, where S_A is the Alice strategy corresponding to $S_A^{(x)}$. Similarly, $I_B(y) = S_B$, where S_B is the Bob strategy corresponding to $S_B^{(y)}$. Now $f^T_{\max}(S_A, S_B) = d$, where the intersection graph of S_A and S_B is the path from the root of T to d. $M(d) = t$, where t is the value of f over the domain represented by d. Hence $f \leq f_{\max}(T)$. So, $f \cong f_{\max}(T)$.

($\Leftarrow$): It is easy to see that the canonical protocol for $f^T_{\max}$ is a perfectly secure protocol for f against active adversaries. □

6 A Strict Infinite Hierarchy

Theorem 5. *Let f be a SSFE function that has a unique decomposition of depth n. Let $\mathcal{F}$ be any (possibly randomized) UC functionality that has a passive-secure protocol with $m < n$ rounds. Then there is no UC-secure protocol for f in the $\mathcal{F}$-hybrid world.*

Proof Sketch. Suppose for contradiction that π is a protocol for f in the $\mathcal{F}$-hybrid world. By Lemma 1, we have that π is also a passive-secure protocol. Let us replace each call in π to $\mathcal{F}$ with the m-round passive-secure protocol for $\mathcal{F}$ to obtain $\widehat{\pi}$, a passive-secure protocol for f in the plain model. For all $\widehat{\pi}$ transcript prefixes, we mark every prefix that corresponds to a turn in the $\mathcal{F}$ subprotocol, except for the first turn of that subprotocol. Intuitively, a $\widehat{\pi}$-adversary should always behave consistently during a marked span, since there may be no equivalent way to achieve the same effect in π, where $\mathcal{F}$-subprotocols are replaced with calls to the ideal $\mathcal{F}$.

Since f is uniquely decomposable, we will describe an attack on the canonical protocol for f, and apply Theorem 1 to obtain an equivalent adversary $\mathcal{S}$ for $\widehat{\pi}$. However, not every adversary attacking $\widehat{\pi}$ can be mapped to an adversary attacking π in the $\mathcal{F}$-hybrid world. $\mathcal{S}$ runs the $\widehat{\pi}$ protocol honestly, though it may change the input occasionally. As long as $\mathcal{S}$ can be modified to change its input only at an unmarked node, then the adversary runs every $\mathcal{F}$ subprotocol entirely honestly, with a consistent input. Then every subprotocol call could be replaced with appropriate calls to $\mathcal{F}$, giving an adversary attacking π in the $\mathcal{F}$-hybrid model.

With overwhelming probability, $\mathcal{S}$ encounters each of the n frontiers in strict order (the frontiers consist of points where alternating parties speak). Note that in any $n - 1$ consecutive steps in the

protocol $\widehat{\pi}$, at least one of the steps must correspond to an unmarked transcript prefix. Since there are n frontiers, it must be the case that either the probability of reaching an unmarked Bob step in the transcript after the first Alice frontier but not after the last Bob frontier is at least half, or the probability of reaching an unmarked Alice step in the transcript after the last Bob frontier but not after the first Alice frontier is at least half.

By symmetry, suppose that the probability of Bob reaching such an unmarked node is at least half. Then as long as we describe an attack for a malicious Bob on the canonical protocol for f that allows $\mathcal{S}$ to change at *any* time after Alice's first move and before Bob's last move, the attack by $\mathcal{S}$ will be successful against π with probability at least half.

Let (x, y) and (x, y') be two inputs with $f(x, y) \neq f(x, y')$ which cause the canonical protocol to take a maximum number of rounds. Let x' be such that Alice's first message in the canonical protocol differs when she has x versus x'. Our attack against the canonical protocol is as follows: If necessary, send the first message consistent with y (which should also be consistent with y'). After Alice's first move, determine whether her input is x or x', and report it to the environment. Then receive either y or y' from the environment, and complete the protocol with answers consistent with y or y', as indicated by the environment. The environment outputs 1 if the adversary succeeds (if the adversary reported the correct input for Alice, and induced the output consistent with either y or y', as requested). Since the canonical protocol behavior is the same for y and y' until Bob's last round, our simulator $\mathcal{S}$ has sufficient time to switch its input internally, as desired.

It is straight-forward to see that if the environment chooses Alice's input $\{x, x'\}$ randomly, and chooses the adversary's challenge $\{y, y'\}$ randomly, no ideal-world adversary can succeed with probability exceeding 1/2. However, the canonical-protocol adversary can clearly succeed with probability 1, thus the $\widehat{\pi}$-attacking adversary $\mathcal{S}$ succeeds with probability $\approx 3/4$. □

Let $g_n : \{0, 2, \ldots, 2n\} \times \{1, 3, \ldots, 2n+1\} \to \{0, 1, \ldots, 2n\}$ be defined as $g_n(x, y) = \min\{x, y\}$. It can be seen by inspection that g_n has a unique decomposition of depth $2n$. The corresponding canonical protocol is the one in which Alice first announces whether $x = 0$, then (if necessary) Bob announces whether $y = 1$, and so on.

Corollary 6. *The functions $g_1, g_2, \ldots$ form a strict, infinite hierarchy of increasing UC complexity. That is, $g_i \sqsubseteq g_{i+1}$, but $g_{i+1} \not\sqsubseteq g_i$ for all i.*

Proof. By Theorem 5, we know that there is no g_n protocol in the g_m-hybrid world, when $m < n$. It suffices to show that there is a UC-secure g_n protocol in the g_{n+1}-hybrid world. The input domain of g_n is a subset of the domain of g_{n+1}, so the g_n protocol is to simply use g_{n+1} as if it were an g_n functionality. If the response from g_{n+1} is out of bounds for g_n (i.e., $2n+1$ or $2n+2$), then abort; otherwise, use the response as the output. Suppose a real-world adversary attempts to send s to g_{n+1} in the protocol. If $s \leq 2n+1$, then s is a valid input for g_n, and the other party will accept, so the simulator simply sends s as the input in the ideal world, and delivers the output. If the corrupt party is Bob, and $s = 2n+3$, then since Alice is honest, her input will always be the minimum. The simulator can send $2n+1$ to g_n in the ideal world (the maximum possible value) and deliver the output. If the corrupt party is Alice and $s = 2n+2$, then if Bob's input to the protocol was $2n+1$, he would abort in the real world interaction. So the simulator sends $2n$ to g_n and delivers the output unless the response is $2n$. □

Corollary 7. *There is no function f which is complete (with respect to the $\sqsubseteq$ relation) for the class of SSFE functions with passive-secure protocols, or for the class of SSFE functions with standalone-secure protocols.*

Proof. Any candidate function f can only implement functions with decomposition depth at most n, where n is the minimum round complexity of a protocol for f. Yet there are functions with arbitrarily high decomposition depth. □

References

[1] M. Backes, J. Müller-Quade, and D. Unruh. On the necessity of rewinding in secure multiparty computation. In S. P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 157–173. Springer, 2007.

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

[3] R. 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.

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

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

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

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

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

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

[10] A. C. Yao. Protocols for secure computation. In *Proc. 23rd FOCS*, pages 160–164. IEEE, 1982.

A Additional Proofs

A.1 Proof of Lemma 1

Proof of Lemma 1. We show that, without loss of generality, the simulator for π maps passive real-world adversaries to passive ideal-world adversaries. A passive adversary $\mathcal{A}$ for π is one which receives an input x from the environment, runs π honestly, and then reports its view to the environment. Suppose $\mathcal{S}$ is the simulator for $\mathcal{A}$. Since in the ideal world, both parties produce output with overwhelming probability, $\mathcal{S}$ must also allow the other party to generate output in the ideal world with overwhelming probability. $\mathcal{S}$ must receive x from the environment, send some x' to the ideal functionality f, receive the output $f(x', y)$, deliver the output, and and then simulate a view to send to the environment.

Suppose x' is the input sent by $\mathcal{S}$ to f. If $f(x, y) \neq f(x', y)$ for some input y for the other party, then consider an environment that uses y for the other party's input. In this environment, the other party will report $f(x, y)$ in the real world, but $f(x', y)$ in the ideal world, so the simulation is unsound. Thus with overwhelming probability, $\mathcal{S}$ sends an input x' such that $f(x, \cdot) \equiv f(x', \cdot)$. We may modify $\mathcal{S}$ to always send x (originally obtained from the environment) to f instead of x'. With overwhelming probability, the reply from f is unaltered by replacing x' with x, and thus the remainder of its simulation remains the same. This modified $\mathcal{S}$ is a *passive* ideal-world adversary: it receives x from the environment, sends x to f, and delivers the output. □

A.2 Proof of Lemma 2

In Lemma 2, we use the following technical lemma which relates the statistical difference on a frontier to a function of the prefixes of the frontier.

Let F be the frontier defined in the proof of Lemma 2 (a set of transcript prefixes corresponding to points where Alice has just spoken), and let S be the set of proper prefixes of F that correspond to points where Alice has just spoken.

Lemma 6. *There exist constants $c_\pi \geq 0$ and $\lambda(v, y, y') \geq 0$ for all $v \in S$ such that:*

$$\sum_{u \in F} \alpha(u, x) \big| \beta(u, y) - \beta(u, y') \big| = c_\pi + \sum_{v \in S} \alpha(v, x) \lambda(v, y, y')$$

Proof. Let S_0 be the prefix-minimal elements of S (if Alice speaks first in the protocol, then $|S_0| = 2$, otherwise S_0 contains just the empty string). We will define a sequence $S_0 \subset S_1 \subset \cdots \subset S_t$ inductively as follows. Let L_i be the set of leaves of S_i. We define a *pop* operation on elements of L_i which are not in F. When we pop an element $u \in S_i$, we expand the partial transcript u by two steps, setting $S_{i+1} = S_i \cup \{u00, u01, u10, u11\}$. After some finite number of operations we reach S_t such that $L_t = F$.

We define the *weight* of a tree S_i as:

$$w(S_i) = \sum_{u \in L_i} \alpha(u, x) \big| \beta(u, y) - \beta(u, y') \big|$$

Observe that $w(S_t) = \mathrm{SD}\left(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x,y'}\right)$. For the base case $w(S_0) = c_\pi \geq 0$. We define the intermediate quantity $\delta(u, y, y') = |\beta(u, y) - \beta(u, y')|$. Suppose the node v was popped in S_i to obtain

S_{i+1}. Then,

$$w(S_{i+1}) - w(S_i) = \alpha(v00, x)\delta(v00, y, y') + \alpha(v01, x)\delta(v01, y, y') + \alpha(v10, x)\delta(v10, y, y') + \alpha(v11, x)\delta(v11, y, y') - \alpha(v, x)\delta(v, y, y')$$

Since $v0$ and $v1$ are nodes where Bob has just spoken, we get that $\delta(v00, y, y') = \delta(v01, y, y') = \delta(v0, y, y')$ and $\delta(v10, y, y') = \delta(v11, y, y') = \delta(v1, y, y')$. Similarly, $\alpha(v0, x) = \alpha(v1, x) = \alpha(v, x)$. By definition, we have $\alpha(v00, x) + \alpha(v01, x) = \alpha(v0, x)$ and $\alpha(v10, x) + \alpha(v11, x) = \alpha(v1, x)$. Using these observation we get:

$$\begin{aligned} w(S_{i+1}) - w(S_i) &= \left(\alpha(v00, x) + \alpha(v01, x)\right)\delta(v0, y, y') \\ &\quad + \left(\alpha(v10, x) + \alpha(v11, x)\right)\delta(v1, y, y') - \alpha(v, x)\delta(v, y, y') \\ &= \alpha(v0, x)\delta(v0, y, y') + \alpha(v1, x)\delta(v1, y, y') - \alpha(v, x)\delta(v, y, y') \\ &= \alpha(v, x)\left[\delta(v0, y, y') + \delta(v1, y, y') - \delta(v, y, y')\right] \end{aligned}$$

Define $\lambda(v, y, y') = \delta(v0, y, y') + \delta(v1, y, y') - \delta(v, y, y')$ and it is easy to see that $\lambda(v, y, y') \geq 0$. Thus we obtain the desired result:

$$w(S_t) = c_\pi + \sum_{v \in S} \alpha(x, v)\lambda(v, y, y') \qquad \square$$

A.3 Proof of Theorem 1

We now prove our main technical result, Theorem 1. First, we establish a convenient lemma:

Lemma 7. *For frontiers F and G, let "$[G \prec F|x, y]$" denote the event that when running the protocol on inputs x, y, the transcript reaches G strictly before reaching F. Then*

$$SD\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x',y'}\Big) \leq SD\Big(\mathcal{D}_G^{x,y}, \mathcal{D}_G^{x',y'}\Big) + \frac{1}{2}(\Pr[G \prec F|x, y] + \Pr[G \prec F|x', y'])$$

Proof. Partition F into F_1 and F_2, where F_1 are the strings in F which are prefixes of strings in G. Thus the distribution over the frontier F_1 is a function of the distribution over the frontier G. Then

$$\begin{aligned} \mathrm{SD}\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x',y'}\Big) &= \frac{1}{2}\sum_{u \in F_1} |\alpha(u, x)\beta(u, y) - \alpha(u, x')\beta(u, y')| + \frac{1}{2}\sum_{u \in F_2} |\alpha(u, x)\beta(u, y) - \alpha(u, x')\beta(u, y')| \\ &\leq \mathrm{SD}\Big(\mathcal{D}_G^{x,y}, \mathcal{D}_G^{x',y'}\Big) + \frac{1}{2}\sum_{u \in F_2} \big(\alpha(u, x)\beta(u, y) + \alpha(u, x')\beta(u, y')\big) \\ &= \mathrm{SD}\Big(\mathcal{D}_G^{x,y}, \mathcal{D}_G^{x',y'}\Big) + \frac{1}{2}\big(\Pr[F_2|x, y] + \Pr[F_2|x', y']\big) \\ &= \mathrm{SD}\Big(\mathcal{D}_G^{x,y}, \mathcal{D}_G^{x',y'}\Big) + \frac{1}{2}(\Pr[G \prec F|x, y] + \Pr[G \prec F|x', y']) \end{aligned} \qquad \square$$

We can now prove Theorem 1. For simplicity, we restrict attention to the case where the decomposition of f is binary (that is, each row decomposition is $X = X_0 \cup X_1$ and each column decomposition is $Y = Y_0 \cup Y_1$), as that is all that is needed for our impossibility results, and the proof is much simpler.

As before, let ϵ be the statistical error of the protocol π, let ν be the negligible error guaranteed by Lemma 2, and define the quantity μ_i according to the recurrence $\mu_0 = 2\epsilon$ and $\mu_{i+1} = 11\nu + 16\mu_i$. Note that for fixed i, μ_i is negligible whenever ϵ and ν are negligible.

The bulk of the proof is in the following inductive lemma. For each step $X \times Y$ in the decomposition of f, we will construct a frontier $F(X, Y)$ with the following property:

Lemma 8. *Let d be the depth (number of decomposition steps) of the decomposition of $f\big|_{X\times Y}$. For all $x, x' \in X$ and $y, y' \in Y$, if (x, y) and (x', y') are in different parts of the* next *decomposition step of $f\big|_{X\times Y}$, then $SD\Big(\mathcal{D}^{x,y}_{F(X,Y)}, \mathcal{D}^{x',y'}_{F(X,Y)}\Big) \geq 1 - \mu_d$; otherwise, $SD\Big(\mathcal{D}^{x,y}_{F(X,Y)}, \mathcal{D}^{x',y'}_{F(X,Y)}\Big) \leq \mu_d$.*

Proof. We prove the claim inductively according to the decomposition of f. For $X \times Y$ such that f is constant on $X \times Y$, we define $F(X, Y)$ as the set of *complete* transcripts (i.e., those on which the parties terminate and give output). By the security of the protocol, we have that $\mathrm{SD}\Big(\mathcal{D}^{x,y}_{F(X,Y)}, \mathcal{D}^{x',y'}_{F(X,Y)}\Big) \leq 2\epsilon = \mu_0$ for all $x, x' \in X$ and $y, y' \in Y$.

Otherwise suppose that $X \times Y$ is further decomposed as $X = X_0 \cup X_1$ (the case for a column decomposition is symmetric). Our first step is to pick representatives $x_0 \in X_0$ and $x_1 \in X_1$ so that there are certain $\boxplus$-minors involving x_0 and x_1, and we will be able to apply Lemma 2 directly. We pick the representatives as follows:

If f is constant on $X_1 \times Y$, then we pick the representative $x_0 \in X_0$ arbitrarily. Otherwise, suppose $X_1 \times Y$ is decomposed as $Y = Y_0 \cup Y_1$. This column decomposition $Y_0 \cup Y_1$ is invalid for $X_0 \times Y$, since if it were valid, the decomposition of f would not be unique. Therefore there must be an $x_0 \in X_0$, $y \in Y_0$, $y' \in Y_1$ such that $f(x_0, y) = f(x_0, y')$. We use this x_0 as the representative for X_0. In this way, we have ensured that for any $x_1 \in X_1$, these values x_0, x_1, y, y' form a $\boxplus$-minor, which (intuitively) witnesses the fact that the $X = X_0 \cup X_1$ decomposition must happen before the $Y = Y_0 \cup Y_1$ decomposition. The representative for X_1 is chosen analogously.

Given these representatives x_0, x_1, we define $F(X, Y)$ to be the frontier given by Lemma 2 for x_0, x_1. We refer to $F(X, Y)$ as F for short, and $F(X_0, Y)$ as G.

We wish to show that for all $x, x' \in X_0$ and $y, y' \in Y$, $\mathrm{SD}\Big(\mathcal{D}^{x,y}_{F}, \mathcal{D}^{x',y'}_{F}\Big) \leq \mu_d$ (the analogous claim for X_1 is symmetric). If f is constant on $X_0 \times Y$, then the statistical difference is in fact at most 2ϵ by the security of the protocol, and we are done. Otherwise, if $X_0 \times Y$ is decomposed as $Y = Y_0 \cup Y_1$, there must be $y_0 \in Y_0$, $y_1 \in Y_1$ such that x_0, x_1, y_0, y_1 are a $\boxplus$ minor, by our choice of x_0, x_1. Thus by Lemma 2, we have that $\mathrm{SD}\big(\mathcal{D}^{x_0,y_0}_{F}, \mathcal{D}^{x_0,y_1}_{F}\big) \leq \nu$. However, inductively we also have that $\mathrm{SD}\big(\mathcal{D}^{x_0,y_0}_{G}, \mathcal{D}^{x_0,y_1}_{G}\big) \geq 1 - \mu_{d-1}$. So by Lemma 7, $\Pr[F \prec G | x_0, y_0] \geq 1 - 2(\nu + \mu_{d-1})$.

Now, take any $x \in X_0$, $y \in Y_0$. We have inductively that $\mathrm{SD}\big(\mathcal{D}^{x,y}_{G}, \mathcal{D}^{x_0,y_0}_{G}\big) \leq \mu_{d-1}$. Note that the condition "$F \prec G$" for a transcript can be determined by seeing only the transcript up to the point G. This suggests a statistical test for distributions on G. Thus we must have

$$\Big|\Pr[F \prec G | x, y] - \Pr[F \prec G | x_0, y_0]\Big| \leq \mathrm{SD}\big(\mathcal{D}^{x,y}_{G}, \mathcal{D}^{x_0,y_0}_{G}\big) \leq \mu_{d-1}.$$

Combining what we know, this implies that $\Pr[F \prec G | x, y] \geq 1 - (2\nu + 3\mu_{d-1})$. An analogous statement also holds for with Y_1 in place of Y_0.

Now take any $x \in X_0$, $y \in Y_0$. By Lemma 7, and the inductive hypothesis,

$$\mathrm{SD}\big(\mathcal{D}^{x,y}_{F}, \mathcal{D}^{x_0,y_0}_{F}\big) \leq \mathrm{SD}\big(\mathcal{D}^{x,y}_{G}, \mathcal{D}^{x_0,y_0}_{G}\big) + \frac{1}{2}\Big(\Pr[G \prec F | x_0, y_0] + \Pr[G \prec F | x, y]\Big) \leq 2\nu + 4\mu_{d-1}$$

The same bound holds with y_1 in place of y_0 and Y_1 in place of Y_0.

Finally, combining everything, for any $x, x' \in X_0$, $y \in Y_0$, and $y' \in Y_1$, we have by the triangle inequality:

$$\begin{aligned}\mathrm{SD}\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x',y'}\Big) &\le \mathrm{SD}\big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x_0,y_0}\big) + \mathrm{SD}\big(\mathcal{D}_F^{x_0,y_0}, \mathcal{D}_F^{x_0,y_1}\big) + \mathrm{SD}\Big(\mathcal{D}_F^{x_0,y_1}, \mathcal{D}_F^{x',y'}\Big) \\ &\le (2\nu + 4\mu_{d-1}) + \nu + (2\nu + 4\mu_{d-1}) = 5\nu + 8\mu_{d-1} \le \mu_d\end{aligned}$$

The case where $y, y' \in Y_0$ or $y, y' \in Y_1$ is shown in a similar way. This proves the first part of the inductive claim.

For the other part of the claim, since $X = X_0 \cup X_1$ is a valid row-decomposition of $X \times Y$, we know that $f(x_0, y) \neq f(x_1, y)$ for every $y \in Y$. Thus $\mathrm{SD}\big(\mathcal{D}_F^{x_0,y}, \mathcal{D}_F^{x_1,y}\big) \ge 1 - \nu$ by Lemma 2 and the definition of F. By a similar triangle inequality argument, we have that for all $x \in X_0$, $x' \in X_1$, $y, y' \in Y$:

$$\begin{aligned}\mathrm{SD}\Big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x',y'}\Big) &\ge -\mathrm{SD}\big(\mathcal{D}_F^{x,y}, \mathcal{D}_F^{x_0,y}\big) + \mathrm{SD}\big(\mathcal{D}_F^{x_0,y}, \mathcal{D}_F^{x_1,y}\big) - \mathrm{SD}\Big(\mathcal{D}_F^{x_1,y}, \mathcal{D}_F^{x',y'}\Big) \\ &\ge 1 - \nu - 2(5\nu + 8\mu_{d-1}) \ge 1 - \mu_{d-1}.\end{aligned}$$

□

Given the previous lemma, we prove Theorem 1:

Proof of Theorem 1. Given such frontiers corresponding to each decomposition step, the required simulation is natural. The simulator's job is to simulate the canonical protocol to $\mathcal{A}$, while interacting with the honest party in the protocol π. The simulator $\mathcal{S}$ simply keeps track of a subdomain in the decomposition of f, and runs π honestly with any representative input. Whenever it reaches the next frontier for a decomposition step by the honest party, it does the following: By the definition of the frontier it can determine (with only negligible error) which part of the decomposition the honest party's input belongs to, and simulate the next step in the canonical protocol to the adversary. Then the simulator receives the adversary's next move in the canonical protocol and changes its own input for π accordingly, if necessary. After the last message of the canonical protocol, $\mathcal{S}$ continues honestly running π until it terminates.

The simulation is perfect except for the following possible sources of error: (1) $\mathcal{S}$ may give an incorrect guess for the honest party's next canonical-protocol message; (2) when honestly executing π with the honest party, $\mathcal{S}$ often changes its input; and (3) $\mathcal{S}$ may reach the frontiers in an unexpected order. Events 1 and 3 both happen with negligible probability by the constructions of the frontiers. Furthermore, conditioned on the frontiers being encountered in the expected order, $\mathcal{S}$ only changes its input only at points where the transcript distribution is nearly independent of its input. In other words, a transcript in which $\mathcal{S}$ changes its input, say from x to x', is indistinguishable from a transcript in which $\mathcal{S}$ had used x' as its input the whole time. Thus with overwhelming probability, the transcript seen by the honest party is indistinguishable from one in which $\mathcal{S}$ executes π entirely honestly with an input that is consistent with the entire canonical-protocol transcript, so the effect given by $\mathcal{S}$ (against π) is the same as that of $\mathcal{A}$ (against the canonical protocol). □

A.4 Proof of Theorem 2

Proof of Theorem 2. ($\Rightarrow$) The canonical protocol for f is passively secure (in fact, with perfect security) [8].

($\Leftarrow$) Suppose f has a passive-secure protocol π. Then it cannot have an OR-minor, since such functions are complete [7]. One can easily verify that if f does not have both a $\boxplus$- and $\boxplus$-minor,

then it is decomposable, and the claim is proven. Otherwise, suppose for contradiction that f is not decomposable.

Intuitively, by Lemma 2, each $\boxplus$-minor x, x', y, y' corresponds to a combination of inputs where Alice must differentiate her input before Bob. Similarly, for a $\boxplus$-minor, Bob must differentiate first. We will leverage these constraints to obtain a contradiction.

For each such minor in f, we can compute the associated frontier F in π. Let $\mathcal{F}_v$ be the union of all F associated with $\boxplus$-minors, and $\mathcal{F}_h$ be the union of all F associated with $\boxplus$-minors. Finally, let F_v be the prefix-minimal elements of $\mathcal{F}_v$, and F_h be the prefix-minimal elements of $\mathcal{F}_h$.

For an arbitrary $x \in X$, $y^* \in Y$, we will show that we will inductively construct a sequence $Y_1 \subset \cdots \subset Y_n = Y$ such that $\mathrm{SD}\left(\mathcal{D}_{F_v}^{x,y^*}, \mathcal{D}_{F_v}^{x,y}\right) \leq i \cdot \nu$ for all $y \in Y_i$, where ν is the negligible quantity guaranteed by Lemma 2. That is, at F_v, transcripts are nearly independent of Bob's input. The base case is $Y_1 = \{y^*\}$.

For the inductive step, we know that $Y = Y_i \cup \overline{Y_i}$ is not a valid column decomposition of f. We must have $f(x', y) = f(x', y')$ for some $x' \in X$, $y \in Y_i$, $y' \in \overline{Y_1}$. We now consider the 2×2 minor induced by x, x', y, y'. If $f(x, y) = f(x, y')$, then clearly by the security of the protocol, $\mathrm{SD}\left(\mathcal{D}_{F_v}^{x,y}, \mathcal{D}_{F_v}^{x,y'}\right) \leq \epsilon \leq \nu$. If $f(x, y) \neq f(x, y')$, then we cannot have $f(x, y) = f(x', y)$ or $f(x, y') = f(x', y')$, since this induces an OR-minor. The only other case is that this 2×2 minor is a $\boxplus$-minor. Let F' be the minor given by Lemma 2 for this minor. Since F_v contains only prefixes of F', we must have $\mathrm{SD}\left(\mathcal{D}_{F_v}^{x,y}, \mathcal{D}_{F_v}^{x,y'}\right) \leq \mathrm{SD}\left(\mathcal{D}_{F'}^{x,y}, \mathcal{D}_{F'}^{x,y'}\right) \leq \nu$ by Lemma 2. In either of the above cases, we have:

$$\mathrm{SD}\left(\mathcal{D}_{F_v}^{x,y^*}, \mathcal{D}_{F_v}^{x,y'}\right) \leq \mathrm{SD}\left(\mathcal{D}_{F_v}^{x,y^*}, \mathcal{D}_{F_v}^{x,y}\right) + \mathrm{SD}\left(\mathcal{D}_{F_v}^{x,y}, \mathcal{D}_{F_v}^{x,y'}\right) \leq (i-1)\nu + \nu = i \cdot \nu$$

Setting $Y_{i+1} = Y_i \cup \{y'\}$ completes the inductive step.

Summarizing, for any $x \in X$, the transcript distribution at F_v is nearly independent of Bob's input. Symmetrically, for any $y \in Y$, the transcript distribution at F_h is nearly independent of Alice's input. We now obtain a contradiction by showing that in any transcript, the probability of reaching F_v before F_h is negligible, as is the probability of reaching F_h before F_v. Since the two sets are disjoint (they correspond to rounds in which different parties speak) and must be reached with overwhelming probability, this is a contradiction.

Let x, x', y, y' denote a $\boxplus$-minor (that is, Alice must disclose first) with associated frontier F given by Lemma 2. We must have $\mathrm{SD}\left(\mathcal{D}_{F}^{x,y}, \mathcal{D}_{F}^{x',y}\right) \geq 1 - \nu$. However, at F_h, we must have $\mathrm{SD}\left(\mathcal{D}_{F_h}^{x,y}, \mathcal{D}_{F_h}^{x',y}\right) \leq O(\nu)$ by the above argument. Thus, only a negligible weight of prefixes in F have a continuation in F_h. Since F_v is composed of the prefix-minimal boundary of at most $O(|X|^2 + |Y|^2)$ such frontiers F, a union bound shows that the only a negligible weight of prefixes in F_v have a continuation in F_h. The symmetric argument follows similarly, completing the proof. $\square$

A.5 Proof of Lemma 3

Lemma 9. *Let π be an arbitrary protocol for two parties to agree on a value in a domain Z with probability 1. (That is, the* outcome *of the protocol is a function of the transcript.) Then, for every partition $Z = C \cup D$, the protocol π falls into one of the following types:*

- *"A" if Alice has a strategy to force the outcome to be a value in C with probability 1, and a strategy to force the outcome to be a value in D with probability 1.*

- *"B" if Bob has a strategy to force the outcome to be a value in C with probability 1, and a strategy to the outcome to be a value in D with probability 1.*
- *"C" if both Alice and Bob have a strategy to force the outcome to be in C with probability 1.*
- *"D" if both Alice and Bob have a strategy to force the outcome to be in D with probability 1.*

Proof. The proof is by induction on the maximum number of rounds in π, when π is described in the normal form for protocols. If π has 0 rounds, π has a constant outcome. So for any partition $Z = C \cup D$, π is a C-type or D-type protocol.

Assuming the inductive hypothesis to hold for protocols up to $n-1$ rounds, we consider a protocol π of n rounds. Suppose the first message is sent by Alice. We define two protocols π_0 and π_1 which are defined as the protocols obtained by assuming the first message in an execution of π to be 0, and to be 1 respectively. Now, by the inductive hypothesis, π_0 and π_1 are one of the four types. We need to consider all combinations of possibilities, but up to symmetries, we need to consider only the following five cases.

1. If π_0 is of A-type so is π, because Alice can first choose to send 0, and then follow the strategies in π_0.
2. If π_0 is of B-type and π_1 is of C-type, then π is of C-type: Alice can force a C outcome by sending 1 and following her strategy in π_1 after that; for Bob, in both π_0 and π_1 he has a strategy to force the outcome to be in C, one of which he will be able to follow after the first message from Alice.
3. If both π_0 and π_1 are of B-type, then π is of B-type.
4. If π_0 is of C-type and π_1 is of D-type, then π is of A-type.
5. If both π_0 and π_1 are of C-type then π is of C-type too. □

Lemma 10. *If f is an SSFE function such that it has two distinct decompositions (in the normal form for decompositions), then there is a partition of the range of f as $Z = C \cup D$ such that one of the following holds:*

1. *Alice has no x such that $\forall y,\ f(x,y) \in C$, and Bob has no y such that $\forall y,\ f(x,y) \in D$.*
2. *Alice has no x such that $\forall y,\ f(x,y) \in D$, and Bob has no y such that $\forall y,\ f(x,y) \in C$.*

Proof. In Claim 1 we show that if f has two distinct decompositions, then there is a decomposition with some node u in the decomposition with domain $X_u \times Y_u$ such that $f|_{X_u \times Y_u}$ is both row and column decomposable. Let T be the decomposition tree for this decomposition, and u be a node whose domain is both row and column decomposable.

We show that the leaves of T can be colored as C and D such that the condition in the lemma holds. For this first we color the leaves of the subtree under the node u, so that both conditions in the lemma hold. In particular, if u is an Alice node then condition 1 holds and if u is a Bob node then condition 2 holds. Now we walk up the tree T from the node u to the root and extend the coloring as follows: at a node v, if v is an Alice node, we color all uncolored leaves under v as D, and if v is a Bob node, we color all uncolored leaves under v as C. Inductively this maintains the property that at Alice nodes condition 1 holds and at Bob nodes condition 2 holds. □

Claim 1. *If f has two distinct decompositions, then there is a decomposition for f with some node u in the decomposition with domain $X_u \times Y_u$ such that $f|_{X_u \times Y_u}$ is both row and column decomposable.*

Proof. Given two distinct decompositions for f, let T and T' be their decomposition trees. If the T starts with a row decomposition and T' starts with a column decomposition, then we can take $X^* = X$ and $Y^* = Y$. Else we can start traversing T and T' until we find two nodes v and v', both say, Alice nodes, in T and T' respectively such that the domains associated with them are equal: $X_v \times Y_v = X'_{v'} \times Y'_{v'}$. Further, $X_v = X_1 \cup \ldots \cup X_\ell$ and $X'_{v'} = X'_1 \cup \ldots \cup X'_{\ell'}$ are not identical. Then it must be the case that there exist X_i, X'_j such that X_i has non-empty intersections with both X'_j and $X'_{v'} \backslash X'_j$ (or symmetrically X'_j has non-empty intersections with X_i and $X_v \backslash X_i$). Then, $X_i \times Y_v$ must be row decomposable as $X_i = X_i \cap X'_j) \cup (X_i \backslash X'_j)$. Further, since f is not constant in $X_i \times Y_v$ (as it is row decomposable) this node is not a leaf in T; then, since v is an Alice node in T, it must be the case that $X_i \times Y_v$ is a Bob node in T, and hence has a column decomposition as well. Thus we conclude that in the decomposition T there is a node u (namely the child of v with domain $X_u = X_i$), such that $f|_{X_u \times Y_u}$ is both row and column decomposable. □

The above lemmas implies Lemma 3, since there is an adversary who can always force an input in C (by symmetry), yet there is no corresponding ideal-world input for f that would allow this.

A.6 Compiler from Passive-Secure to UC-Secure

Lemma 11. *If π is a passive-secure protocol for f, then the compiled version of π (Figure 3) is a UC-secure protocol in the $\mathcal{F}_{\mathsf{com}}$-hybrid world.*

Proof. We describe the simulator for a corrupt Alice (the other case is symmetric). In the setup phase, the simulator simulates $\mathcal{F}_{\mathsf{com}}$ to immediately extract the $\sigma[i, j]$ and $\chi[i, j]$ commitments. It then picks random b_i challenges. If any of the b_i will result in the adversary being "caught" in the setup phase, then the simulator does not contact the ideal functionality for f. The simulator gives random consistent values for party 2's commitments in the setup phase, gives the b_i challenges, and then aborts after the adversary opens its commitments.

Otherwise, if the b_i challenges would not cause an abort, the simulator will contact the ideal functionality. We say that $\sigma[i, \cdot]$ is valid if it is a permutation of $\{1, \ldots, n_x\}$. We say that $\chi[i, \cdot]$ is valid if all but one of the values is zero. If for all i, either $\sigma[i, \cdot]$ or $\chi[i, \cdot]$ are valid (and yet the b_i challenges did not catch the adversary — an event which can happen only with probability 2^{-k}), the simulator aborts. Otherwise consider an arbitrary i such that both $\sigma[i, \cdot]$ and $\chi[i, \cdot]$ are valid. Let x be the unique value such that $\chi[i, \sigma[i, x]] = 1$. The simulator will send x to the ideal functionality on behalf of the adversary, and receive $f(x, y)$. It is finally the simulator's task to decide whether to deliver the output to honest Bob.

Given x and $f(x, y)$, choose any y^* such that $f(x, y) = f(x, y^*)$. The simulator then proceeds to simulate Bob honestly as if it had input y^*. The simulator can always allow Bob to open his commitments consistently. If at any point the simulated Bob aborts, the simulation ends without delivering the output in the ideal world. If simulated Bob terminates successfully, the simulator delivers the output. Note that the adversary cannot successfully deviate from the steps of π that are inconsistent with having input x, by virtue of there being a consistent $\chi[i, \cdot]$ and $\sigma[i, \cdot]$ uniquely committed to x. If the adversary tries sending a bit in π inconsistent with its input, there will be no way for it to consistently open this particular committing $\sigma[i, \cdot]$ and $\chi[i, \cdot]$. Since $f(x, y) = f(x, y^*)$, the steps of π simulated by the simulator must be statistically indistinguishable. Furthermore, the commitments opened by the simulator are distributed identically as when Bob runs the protocol with input y. By the security of π, the simulation is indistinguishable from the real-world interaction (though with an added error probability of 2^{-k}). □

Our protocol is essentially symmetric with respect to the two parties. We describe only the behavior of Alice. We denote the parties' input ranges as $X = \{1, \ldots, n_x\}$ and $Y = \{1, \ldots, n_y\}$, respectively. The protocol's security parameter is k.

Setup phase:

1. Choose random permutations $\sigma_1, \ldots, \sigma_k$ of X. Let χ be the characteristic vector of x in X. Commit to the values $\chi[i, j] = \chi_{\sigma_i^{-1}(j)}$ and $\sigma[i, j] = \sigma_i(j)$ for $i \leq k$ and $j \leq n_x$.

2. Wait for Bob to similarly commit to his own $\chi'[i, j]$ and $\sigma'[i, j]$ values.

3. Receive $b_i \in \{0, 1\}$ for $i \leq k$. For each i, if $b_i = 0$, then open each $\sigma[i, j]$ commitment; otherwise, open each $\chi[i, j]$ commitment.

4. Similarly, choose random $b_i \leftarrow \{0, 1\}$ for all $i \leq k$ and send them to Bob. For each i such that $b_i = 0$, expect $\sigma'[i, j]$ to be opened and verify that they constitute a permutation of $\{1, \ldots, n_y\}$, else abort. For each i such that $b_i = 1$, expect each $\chi'[i.j]$ to be opened and verify that all but one of them are equal to 0, else abort.

Then, for each step of π:

1. If this is a step where Alice sends a message, then let $b \in \{0, 1\}$ be the message prescribed by the protocol at this point. Let Z be the set of inputs which would have prescribed sending message $1 - b$ at this point in the protocol. Send b to Bob, and and for each $i \leq k$ and $j \in Z$, open the commitments of $\sigma[i, j]$ and $\chi[i, \sigma_i(j)]$ $(= \chi_j)$. (Half of these commitments will already be opened).

2. If this is a step where Bob sends a message, then expect a next message $b \in \{0, 1\}$ for the π protocol simulation. Let Z be the set of inputs for Bob which would have prescribed that Bob send message $1 - b$ at this point in the protocol. from party 2. For each $i \leq k$ and $j \in Z$, expect $\sigma'[i, j]$ to be opened (if not already). If any $\sigma'[i, j] \notin \{1, \ldots, n_y\}$, or if $\sigma'[i, j] = \sigma'[i, j']$ for any $j \neq j'$, then abort. Expect $\chi'[i, \sigma'[i, j]]$ to be opened to 0 (if not already), else abort.

When π terminates, terminate with the same output.

Figure 3: UC-secure compilation of passive-secure protocol π in the $\mathcal{F}_{\mathsf{com}}$-hybrid world