

# On the Structure of Unconditional UC Hybrid Protocols

Mike Rosulek\*

Morgan Shirley\*

## Abstract

We study the problem of secure two-party computation in the presence of a trusted setup. If there is an unconditionally UC-secure protocol for  $f$  that makes use of calls to an ideal  $g$ , then we say that  $f$  *reduces to*  $g$  (and write  $f \sqsubseteq g$ ). Some  $g$  are *complete* in the sense that *all* functions reduce to  $g$ . However, almost nothing is known about the power of an incomplete  $g$  in this setting. We shed light on this gap by showing a characterization of  $f \sqsubseteq g$  for incomplete  $g$ .

Very roughly speaking, we show that  $f$  reduces to  $g$  if and only if it does so by the simplest possible protocol: one that makes a single call to ideal  $g$  and uses no further communication. Furthermore, such simple protocols can be characterized by a natural combinatorial condition on  $f$  and  $g$ .

Looking more closely, our characterization applies only to a very wide class of  $f$ , and only for protocols that are deterministic or logarithmic-round. However, we give concrete examples showing that both of these limitations are inherent to the characterization itself. Functions not covered by our characterization exhibit qualitatively different properties. Likewise, randomized, superlogarithmic-round protocols are qualitatively more powerful than deterministic or logarithmic-round ones.

## 1 Introduction

In 2-party secure function evaluation (SFE), Alice holds a private input  $x \in X$ , Bob holds a private input  $y \in Y$ , and the parties interact to learn  $f(x, y)$  for some agreed-upon function  $f : X \times Y \rightarrow Z$ . Each party should learn no more than can be inferred from  $f(x, y)$  alone, even when that party behaves adversarially.

Different functions  $f$  have different inherent complexities. For instance, the XOR function can be securely realized in the presence of computationally unbounded, semi-honest adversaries, but the oblivious transfer function cannot. In this sense, oblivious transfer has inherently more “cryptographic complexity” than XOR.

Another way to compare the complexity of functions is to use a *reduction*. The most natural reduction from  $f$  to  $g$  is a secure protocol for  $f$  where the parties are allowed to use an ideally secure black-box for  $g$  (ideal here means that this black-box takes inputs from both parties, and reveals only the output of  $g$ ). Depending on the security required of the protocol for  $f$ , we obtain reductions of various strengths that can resolve finer distinctions in cryptographic complexity.

**Cryptographic complexity & related work.** In this work we exclusively focus on reductions between two-party, deterministic SFE functions. We consider a reduction defined in terms of UC security [Can01] against computationally unbounded adversaries. We write  $f \sqsubseteq g$  to denote that

---

\*Oregon State University. rosulekm@eecs.oregonstate.edu, shirlema@oregonstate.edu. Partially supported by NSF awards 1149647 & 1617197

there is a UC-secure protocol securely realizing  $f$  against unbounded adversaries, that makes calls to an ideal  $g$  functionality (i.e., a protocol in the “ $g$ -hybrid model”).

After defining a notion of reducibility, the most natural step is to identify which objects are *complete* for the reduction. A function  $g$  is **complete** (under  $\sqsubseteq$ ) if  $f \sqsubseteq g$  for all  $f$ . Otherwise we say that  $g$  is **incomplete**.

Kilian [Kil88] was the first to consider completeness of SFE functionalities, proving that the oblivious transfer function is complete. Although the result pre-dates the UC model, a variant of the construction from [Kil89] is likely to achieve UC security — i.e., oblivious transfer is complete under the  $\sqsubseteq$  reduction that we consider in this work. Later work characterized exactly which functions are complete (w.r.t. malicious, unconditional security): for symmetric SFE (where both parties receive the same output) [Kil91], for asymmetric SFE (where only one party receives output) [Kil00], and even for randomized SFE functions [MPR12, KMPS14].

When one or both of  $f, g$  are complete, the question of  $f \sqsubseteq g$  is simple to answer. If  $g$  is complete, then  $f \sqsubseteq g$ . If  $f$  is complete but  $g$  is not, then  $f \not\sqsubseteq g$ . The goal of this line of work is to therefore **understand when  $f \sqsubseteq g$ , for  $f$  and  $g$  which are incomplete**.

Write  $f \sqsubset g$  if  $f \sqsubseteq g$  but  $g \not\sqsubseteq f$ . Prabhakaran & Rosulek [PR08] gave an example of four functions that satisfy  $f_1 \sqsubset f_2 \sqsubset f_3 \sqsubset f_4$ . Maji, Prabhakaran & Rosulek [MPR09] extended this result to show an *infinite* strict hierarchy  $f_1 \sqsubset f_2 \sqsubset \dots \sqsubset f_i \sqsubset \dots$ , and also showed an example of a pair of functions that are incomparable under the reduction:  $f \not\sqsubseteq g \wedge g \not\sqsubseteq f$ . The same authors in [MPR10a] later proved several results of the form “ $f$  reduces to  $g$  with polynomial-time UC security if and only if one-way functions exist” and “ $f$  reduces to  $g$  with polynomial-time UC security if and only if semi-honest OT protocols exist”. These results imply that  $f \not\sqsubseteq g$  (i.e., against unbounded adversaries).

These results hint at a rich landscape of complexity with respect to the  $\sqsubseteq$  reduction, but fall well short of revealing the entire picture. First, they are not complete characterizations, but give only necessary conditions for  $f \sqsubseteq g$ . Second, the techniques in these works apply only to  $f$  and  $g$  that have *semi-honest-secure* protocols. This leaves a large class of functions that are neither complete nor admit any semi-honest protocol (simple characterizations of both properties are known [Bea89, Kus89, Kil91]). A canonical example of such a function is the so-called “spiral” function shown in Figure 1c. Currently almost nothing is known about reductions involving such intermediate functions.

**Other related work.** We consider a reduction based on UC security against unbounded adversaries. Weaker reductions have been studied, but they do not turn out to illuminate many distinctions in complexity. For example, when considering a reduction defined in terms of *polynomial-time* UC security, it turns out that every SFE is either complete or trivial (it reduces to every other function) [MPR10b]. The landscape is much simpler under polynomial-time-UC reductions, with only two distinct levels of complexity.

In this paper we prove qualitative differences in the power of randomized and deterministic protocols, even when realizing deterministic functions using other deterministic functions (e.g., we show specific, deterministic  $f$  and  $g$  where  $f \sqsubseteq g$  via a randomized protocol but not by any deterministic protocol). In the two-party setting, Dodis & Micali [DM99] show such a separation among *complete*  $f$  and  $g$ , for a special class of protocols (in which one party does not speak). Beimel & Malkin [BM04] show specific  $f$  and  $g$  for which randomized protocols make exponentially fewer calls to  $g$  than deterministic protocols.

## 2 Overview of Our Results

**Scope of Results: Incomplete, Non-Unilateral Functions.** As mentioned in the introduction, the question of  $f \sqsubseteq g$  is straight-forward when one of  $\{f, g\}$  are complete. We therefore focus on characterizing  $f \sqsubseteq g$  when both are incomplete.

We say that  $f$  is **unilateral** if there exists an input  $y^*$  for one of the parties (by symmetry, Bob) such that  $f(\cdot, y^*)$  is a constant function. That is, by choosing input  $y^*$  Bob can *unilaterally* fix the output of  $f$ . We characterize  $f \sqsubseteq g$ , when  $f$  is non-unilateral. In [Section 7.1](#) we show an example of unilateral  $f, g$  that do *not* obey the characterization, demonstrating that this restriction is tight.

**Statement of Main Result.** We show a complete and combinatorial characterization of  $f \sqsubseteq g$ , for a natural class of protocols. For this characterization, identify each function  $f$  with its 2-dimensional truth table (rows corresponding to Alice-inputs and columns to Bob-inputs). We say that  $f$  **embeds in**  $g$  if  $f$  appears as a submatrix of  $g$ , subject to some other restrictions (essentially, the other parts of  $g$  can't “interfere” with the  $f$ -submatrix — the formal definition is in [Section 4](#)). We then prove our main theorem:

**Theorem 1.** *The following are equivalent, when  $f$  and  $g$  are incomplete and  $f$  is non-unilateral.*

1.  $f \sqsubseteq g$  via a worst-case  $O(\log \kappa)$ -round protocol ( $\kappa$  is the security parameter)
2.  $f \sqsubseteq g$  via a deterministic protocol
3.  $f \sqsubseteq g$  via a deterministic protocol consisting of a single call to  $g$  and no additional communication.
4.  $f$  embeds in  $g$

**Technical Approach.** The most involved part of our main theorem is proving (1)  $\Rightarrow$  (3) and (2)  $\Rightarrow$  (3). Intuitively this involves “compressing” an arbitrary protocol for  $f \sqsubseteq g$  into a single call to  $g$ .

Our first step is to show that every secure protocol for  $f \sqsubseteq g$  can be transformed into one with the following **instantaneous** property:

- With overwhelming probability, the protocol terminates immediately following some call to  $g$ .
- Strictly before this terminal call to  $g$ , the protocol transcript leaks negligible information about either party's inputs.

Our main technical tool is that of *frontier analysis*, which was introduced in [\[MPR09\]](#) and extended in [\[MOPR11\]](#). A *frontier* in the protocol is simply the collection of partial transcripts where some statistical condition is true for the first time. Roughly speaking, we define two frontiers for each party: one expressing “the first time the simulator is likely to extract” and another for “the first time honest parties can reliably predict the final output.” We then argue that these frontiers must all be reached simultaneously, with overwhelming probability. As such, these events can happen only as the result of a call to  $g$ . Furthermore, the protocol can be safely truncated after reaching the frontiers (since both parties can already predict the final output). The result of truncation is a protocol with the “instantaneous” property described above.

Next, we show that a protocol with this instantaneous form can be compressed to a single call to  $g$ . Whereas the previous step applies for any protocol, for technical reasons we are only able to compress protocols that are deterministic or have logarithmic (worst-case) round complexity.

**Tightness of the Characterization: Protocol Structure.** Our main theorem does not apply to super-logarithmic-round, randomized protocols. We show that this limitation is inherent, and that there is actually a **qualitative difference** in the power of super-logarithmic, randomized protocols! In [Section 7.2](#) we demonstrate a specific pair  $f$  and  $g$  with the following properties:

1.  $f$  does not embed in  $g$ . Hence, by the classification theorem,  $f \not\sqsubseteq g$  via any deterministic protocol or (randomized) logarithmic-round protocol.
2.  $f \sqsubseteq g$  via a randomized protocol whose *expected* round complexity is constant, but whose *worst-case* round complexity is  $r(\kappa)$  for *any*  $r(\kappa) = \omega(\log \kappa)$ . Interestingly, this protocol still has the *instantaneous* property described above.

In this bizarre example, we have a secure protocol that leaks no information about the parties’ inputs before suddenly revealing all of  $f(x, y)$  in a single call to  $g$ . Yet any attempt to compress the protocol to just the “meaningful” call to  $g$  results in an insecure protocol!

To the best of our knowledge, this is the first example of a round-complexity lower bound in the information-theoretic setting that is not *constant* in the security parameter. Almost all previous work characterizing  $f \sqsubseteq g$  focuses on the case where  $g$  is complete. In this setting, all protocols can achieve constant rounds:  $f$  can be realized in constant rounds from oblivious transfer [[IPS08](#)], which itself can be realized in constant rounds from any complete  $g$  [[KMQ11](#)].<sup>1</sup>

**Tightness of the Characterization: Unilateral functions.** Our main theorem also does not characterize  $f \sqsubseteq g$  when  $f$  is unilateral. This restriction is inherent. In [Section 7.1](#), we demonstrate an example pair of (unilateral, incomplete) functions  $f, g$  where  $f \not\sqsubseteq g$  by any protocol consisting entirely of a single call to  $g$ . Yet we show a protocol for  $f \sqsubseteq g$  consisting of a single call to  $g$  plus *one additional message* from one party to the other. Intuitively  $f$  has several unilateral inputs, and the extra protocol message is required to convey which one was chosen.

## 3 Preliminaries

### 3.1 Secure Function Evaluation, UC Security

We assume the reader has familiarity with the UC framework (a brief overview is given in [Appendix A](#)). In this work we study deterministic 2-party secure function evaluation (SFE), in the universal composability (UC) framework [[Can01](#)] against computationally unbounded adversaries that corrupt parties *statically* (i.e., once and for all before the protocol begins). We consider security-with-abort, meaning that malicious parties are allowed to learn their output first, and delay the honest parties from receiving output (perhaps indefinitely).

We use the following notation:

$f \sqsubseteq g$ : there is a secure protocol (UC, unconditional) for  $f$  that uses calls to an ideal  $g$  (i.e., a secure protocol in the  $g$ -hybrid model).

$f \sqsubseteq_1 g$ : same as  $f \sqsubseteq g$ , but via a protocol that makes only a single call to  $g$  and uses no additional communication.

---

<sup>1</sup>This argument results in a protocol with  $O(d_f)$  rounds, where  $d_f$  is the multiplicative depth of a circuit realizing  $f$ . It is indeed a major open problem to give a protocol with asymptotically fewer rounds. Since in our setting  $g$  is incomplete, our goal is not to securely realize *all*  $f$  from  $g$ . Instead, we consider a fixed  $f$  and  $g$ , and hence  $d_f$  is a constant with respect to the security parameter.

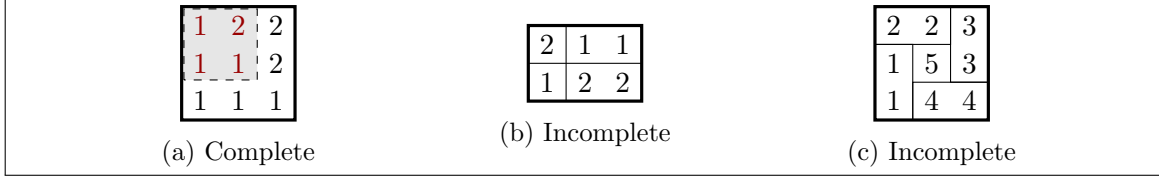


Figure 1: Example SFE functions.

### 3.2 Combinatorial Properties of Complete/Incomplete $f$

We review some basics of 2-party SFE. Let  $f$  be a 2-party SFE with domain  $X \times Y$ . A fundamental property of SFE has to do with decomposing the function into “rectangles.” For an SFE function  $f$ , define the following:

$$\text{rect}_f(x, y) = \{x' \mid f(x, y) = f(x', y)\} \times \{y' \mid f(x, y) = f(x, y')\}$$

We refer to  $\text{rect}_f(x, y)$  as a **rectangle** of  $f$ .

**Theorem 2** ([Kil91]). *The following are equivalent:*

1.  $f$  is incomplete.
2. For all  $x, x', y, y'$ , if  $f(x, y) = f(x', y) = f(x, y')$  then  $f(x', y') = f(x, y)$ .
3. For all  $x, y$ , the value  $\text{rect}_f(x, y)$  is uniquely determined by  $f(x, y)$  and just one of  $\{x, y\}$ . Likewise,  $f(x, y)$  is uniquely determined by  $\text{rect}_f(x, y)$  and just one of  $\{x, y\}$ .

The equivalence of 1 & 2 is a restatement of Kilian [Kil91], and the other equivalence follows easily from the definition of  $\text{rect}_f$ . Item 3 implies that without loss of generality we can think of the parties as computing the function  $(x, y) \mapsto \text{rect}_f(x, y)$  instead of the function  $f(x, y)$ .

Figure 1 shows three example functions, with the partition into rectangles given for the incomplete functions. Note that the second function has 4 rectangles: two rectangles with output 1, and two with output 2. In general the rectangles are more fundamental to the function than its actual outputs.

### 3.3 Properties of $g$ -Hybrid Protocols for Incomplete $g$

Fix a 2-party protocol  $\pi$ , and let  $t$  be a partial transcript (i.e., a prefix of a complete protocol transcript). We use  $\Pr_\pi[t|xy]$  to denote the probability of obtaining a protocol transcript with prefix  $t$ , when both parties run the protocol honestly with respective inputs  $x$  and  $y$ .

Write  $t$  as a sequence of messages  $t = (m_1, \dots, m_k)$ . Suppose Alice sends the odd-numbered messages. Then the choice of the odd-numbered (resp. even-numbered) messages depends only on the previous messages and  $x$  (resp.  $y$ ), but not on  $y$  (resp.  $x$ ). We can therefore write:

$$\begin{aligned} \Pr_\pi[t|xy] &= \prod_{i=1}^k \Pr_\pi[m_i|xy, m_1 \cdots m_{i-1}] \\ &= \left( \prod_{i \text{ odd}} \Pr_\pi[m_i|x, m_1 \cdots m_{i-1}] \right) \left( \prod_{i \text{ even}} \Pr_\pi[m_i|y, m_1 \cdots m_{i-1}] \right) \\ &\stackrel{\text{def}}{=} \Pr_\pi[t|x] \Pr_\pi[t|y] \end{aligned} \quad (\star)$$

Here we are defining  $\Pr_\pi[t|x]$  and  $\Pr_\pi[t|y]$  to be equal to the parenthesized quantities. Essentially,  $\Pr_\pi[t|x]$  is the probability that Alice behaves consistently with  $t$  when her input is  $x$ .

**In the  $g$ -hybrid model.** A similar property also holds when the parties can call an ideal functionality  $g$  (i.e., protocols in the  $g$ -hybrid model), but only when  $g$  is *incomplete*.

When parties invoke  $g$ , its output is added to the joint transcript. From [Theorem 2](#) this is equivalent to adding  $\text{rect}_g(\tilde{x}, \tilde{y})$  to the transcript, where  $\tilde{x}$  and  $\tilde{y}$  were the inputs that the parties gave to this instance of  $g$ . Let  $\tilde{X} \times \tilde{Y}$  be a particular rectangle in  $g$ , then:

$$\Pr[\text{rect}_g(\tilde{x}, \tilde{y}) = \tilde{X} \times \tilde{Y}] = \Pr[\tilde{x} \in \tilde{X}] \Pr[\tilde{y} \in \tilde{Y}]$$

Alice’s choice of  $\tilde{x}$  depends only on her  $f$ -protocol input and the transcript so far; similarly  $\tilde{y}$  depends only on Bob’s input and the transcript so far. Hence, even protocols in the  $g$ -hybrid model satisfy the product property  $(\star)$ .

**Stateless parties/adversaries.** The “standard” way of defining a protocol is for each party to initially choose a random tape. Their subsequent behavior is a deterministic function of the random tape, their input, and the transcript so far.

One way to interpret the product property  $(\star)$  — for protocols in the  $g$ -hybrid model for incomplete  $g$  — is that Alice’s view (including her private randomness) is independent of Bob’s view (including his randomness), given the transcript.

Therefore, *any  $g$ -hybrid protocol  $\pi$*  can be purged of stateful randomness in the following way. At each step, a stateless party can (1) sample a random tape conditioned on it being consistent with their private input and transcript so far; (2) use that (ephemeral) random tape to choose the next move in the protocol; (3) discard the ephemeral random tape. Note that this transformation may require exponential time, but we consider all parties to have unbounded computation.

Importantly, this transformation *also applies to adversaries* — that is, without loss of generality we consider only stateless adversaries. The ability to consider only stateless adversaries is perhaps the **fundamental property of incomplete  $g$**  that we use in this work.

## 4 Reducibility Characterization

We define the combinatorial condition at the heart of our main theorem. Intuitively,  $f$  embeds to  $g$  if one can identify a submatrix of  $g$  that “looks like”  $f$ . Of course, the outputs of  $f$  might be renamed relative to  $g$ . Such a submatrix property suffices for a semi-honest protocol for  $f$  using  $g$ , where parties simply use the subset of inputs of  $g$  that comprise the  $f$ -submatrix. However, such a protocol need not be secure in the presence of *malicious* adversaries, because other inputs of  $g$  may “interfere” with the  $f$ -submatrix. There are two main things that can go wrong, epitomized in the following examples:

$$f_1 = \begin{bmatrix} 1 & 3 \\ 1 & 4 \\ 2 & 4 \end{bmatrix} \not\subseteq \begin{bmatrix} 1 & 3 & 5 \\ 1 & 4 & 6 \\ 2 & 4 & 7 \end{bmatrix} = g_1; \quad f_2 = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \not\subseteq \begin{bmatrix} 1 & 3 & 3 \\ 2 & 2 & 4 \end{bmatrix} = g_2$$

Note that  $f_1$  appears as the white submatrix of  $g_1$ . But when a corrupt column-player cheats and uses the shaded column of  $g_1$ , he completely learns the row-players input. Yet no column of  $f_1$  legally allows this.

Similarly,  $f_2$  appears as a submatrix of  $g_2$ . Consider a corrupt column-player who uses the shaded column of  $g_2$ . There is no *single* input for  $f_2$  that “explains” the effect of this behavior for all possible inputs of the row-player. Concretely, there is no input of  $f_2$  that guarantees an output in  $\{2, 3\}$ .

The requirements for embedding are formalized in the following definition:

**Definition 3.** For two functions  $\alpha$  and  $\beta$  we say that  $\alpha$  **leaks no more than**  $\beta$  if  $\beta(y) = \beta(y') \Rightarrow \alpha(y) = \alpha(y')$  for all inputs  $y, y'$ . We say that  $\alpha$  **refines**  $\beta$  if  $\beta(y) \in \{\alpha(y), \perp\}$  for all inputs  $y$ .

Let  $f : X \times Y \rightarrow Z$  and  $g : \widehat{X} \times \widehat{Y} \rightarrow \widehat{Z}$ . Without loss of generality, assume  $f(x, y) = \text{rect}_f(x, y)$  and  $g(x, y) = \text{rect}_g(x, y)$ . We say that  $f$  **embeds in**  $g$  if:

1. (*f appears as a submatrix in g*) There exist two injective mappings,  $A : X \rightarrow \widehat{X}$  and  $B : Y \rightarrow \widehat{Y}$ , and a third mapping,  $C : \widehat{Z} \rightarrow Z \cup \{\perp\}$ , such that  $\forall x \in X, y \in Y : f(x, y) = C(g(A(x), B(y)))$ .
2. (*security guarantees*) There exist mappings  $\widehat{A} : \widehat{X} \rightarrow X$  and  $\widehat{B} : \widehat{Y} \rightarrow Y$  such that the following hold:
  - (a) (*g doesn't reveal too much information*)
    - for all  $\widehat{x} \in \widehat{X}$ ,  $g(\widehat{x}, B(\cdot))$  leaks no more than  $f(\widehat{A}(\widehat{x}), \cdot)$
    - for all  $\widehat{y} \in \widehat{Y}$ ,  $g(A(\cdot), \widehat{y})$  leaks no more than  $f(\cdot, \widehat{B}(\widehat{y}))$
  - (b) (*there are no ambiguous g-inputs*)
    - for all  $\widehat{x} \in \widehat{X}$ ,  $f(\widehat{A}(\widehat{x}), \cdot)$  refines  $C(g(\widehat{x}, B(\cdot)))$ .
    - for all  $\widehat{y} \in \widehat{Y}$ ,  $f(\cdot, \widehat{B}(\widehat{y}))$  refines  $C(g(A(\cdot), \widehat{y}))$ .

To understand this definition, it helps to see how the mappings  $A, B, C, \widehat{A}, \widehat{B}$  relate to a secure protocol demonstrating  $f \sqsubseteq_1 g$ :

**Lemma 4.** If  $f$  embeds in  $g$ , then  $f \sqsubseteq_1 g$  via a deterministic protocol. This proves (4)  $\Rightarrow$  [(1)  $\wedge$  (2)  $\wedge$  (3)] of [Theorem 1](#) (stated in [Section 2](#)).

*Proof.* Let  $f$  embed in  $g$ , with associated mappings as in [Definition 3](#). The protocol for  $f$  is as follows:

- Alice sends input  $A(x)$  to  $g$  where  $x$  is her  $f$ -input.
- Bob sends input  $B(y)$  to  $g$  where  $y$  is his  $f$ -input.
- The parties both output  $C(z)$  where  $z$  is the output they receive from  $g$  (they output  $\perp$  if  $g$  gives output  $\perp$ ).

Correctness follows from the first condition of [Definition 3](#). Due to the symmetry in the definitions/protocol, we show security only against a malicious Alice.

Suppose Alice sends input  $\widehat{x}$  to  $g$ . In the real protocol, Alice's view will consist of  $g(\widehat{x}, B(y))$  and Bob's output will be  $C(g(\widehat{x}, B(y)))$ . In the ideal world, the simulator will do the following:

- The simulator sends  $x^* = \widehat{A}(\widehat{x})$  to the ideal  $f$ , and obtains output  $f(x^*, y) = C(g(A(x^*), B(y)))$ .
- The simulator does not know Bob's input  $y$  but can choose any  $y'$  such that  $f(x^*, y') = f(x^*, y)$ . The simulator can give  $g(\widehat{x}, B(y'))$  to Alice as her simulated view. From part 2a of [Definition 3](#), we have that this is identical to the real view  $g(\widehat{x}, B(y))$ .
- The simulator checks whether  $C(g(\widehat{x}, B(y))) = \perp$ , and if so sends (DELIVER, 0) to  $f$ . In this case, Bob's real and ideal outputs will both be  $\perp$ . Otherwise, it sends (DELIVER, 1) to  $f$  and Bob will receive output  $f(x^*, y)$ .

Bob's ideal output is  $f(x^*, y) = C(g(A(x^*), B(y)))$ . From condition 2b of [Definition 3](#), this matches the real output  $C(g(\widehat{x}, B(y)))$ .  $\square$

**Lemma 5.** *For non-unilateral  $f$ , if  $f \sqsubseteq_1 g$  via a deterministic protocol with simulation error<sup>2</sup> less than 1, then  $f$  embeds in  $g$ . This proves (3)  $\Rightarrow$  (4) of [Theorem 1](#) (stated in [Section 2](#)).*

*Sketch.* The full proof is in [Appendix B](#). The fact that  $f \sqsubseteq_1 g$  by some deterministic protocol immediately reveals part 1 of the embedding: there must be some set of mappings that Alice and Bob use to map  $f$ -inputs to  $g$ -inputs and  $g$ -outputs to  $f$ -outputs.

The main technical portion of this proof is that, if simulator mappings  $\hat{A}$  and  $\hat{B}$  do not follow the rules in parts 2a and 2b of [Definition 3](#), then the simulation error is 1, which contradicts our assertion that the simulation error is less than 1. The intuition for these attacks is clear from the examples above.  $\square$

## 5 Instantaneous Protocols

In this section we show how to transform any secure protocol in the  $g$ -hybrid model into one that has an “instantaneous” property (described further in [Section 5.4](#)). The results in this section apply to arbitrary protocols. Later in [Section 6](#) we give further transformations that are restricted to deterministic or logarithmic-round protocols.

### 5.1 Frontier Basics

Recall  $\Pr_\pi[\mathcal{E}|xy]$  denotes the probability of event  $\mathcal{E}$  when the protocol is run honestly with inputs  $x$  and  $y$ . Following [Section 3.3](#), we write  $\Pr_\pi[\mathcal{E}|xy] = \Pr_\pi[\mathcal{E}|x] \Pr_\pi[\mathcal{E}|y]$ . Specifically, if  $t$  is a partial protocol transcript, we write  $\Pr_\pi[t|xy]$  to refer to the probability of generating a transcript that has  $t$  as a prefix.

We write  $\Pr_\pi[\mathcal{E}|txy]$  to denote the probability that event  $\mathcal{E}$  happens, given that the parties run with inputs  $x$  and  $y$ , and conditioned on  $t$  occurs as a prefix. More formally, the probability considers what happens when the parties run the protocol honestly with inputs  $x$  and  $y$ , but are initialized with  $t$  as the partial transcript.

Let  $F$  be any set of partial protocol transcripts, with the property that if  $t \in F$ , and  $t$  is a prefix of  $t'$ , then  $t' \in F$ . In other words,  $F$  describes an event in the protocol that happens and does not “unhappen.” In this case we call  $F$  a **frontier**, using the terminology of [\[MPR09\]](#).

It is sometimes helpful to associate the frontier  $F$  with its set of prefix-minimal elements, as these represents transcript where *some condition happened for the first time*. Let  $\text{first}(F)$  denote the prefix-minimal elements of  $F$ .

If  $F$  is a frontier, we use notation  $\Pr_\pi[F|xy]$  to denote the probability that  $F$  is encountered when running the protocol honestly on inputs  $x$  and  $y$ . More formally:

$$\Pr_\pi[F|xy] \stackrel{\text{def}}{=} \sum_{t \in \text{first}(F)} \Pr_\pi[t|xy]$$

Finally, if  $F$  and  $G$  are two frontiers, then “ $F < G$ ” denotes the event “either  $F$  happens strictly before  $G$ , or  $F$  happens and  $G$  never happens.” More formally,

$$\Pr_\pi[F < G|xy] \stackrel{\text{def}}{=} \sum_{t \in \text{first}(F \setminus G)} \Pr_\pi[t|xy]$$

---

<sup>2</sup>The simulation error of a protocol  $\pi$  is the maximum (supremum) over all environments of  $|p_{\text{real}} - p_{\text{ideal}}|$ , where  $p_{\text{real}}$  is the probability that the environment outputs 1 in the real interaction and  $p_{\text{ideal}}$  is the probability that it outputs 1 in the ideal interaction.



## 5.2 Our Frontiers

Our analysis relies on two types of frontiers that we introduce:

$F_{\mathbf{A-ext}}^x$ : captures the first time that the *simulator extracts* with reasonable probability, in an ideal-world interaction involving a corrupt Alice running *honestly* on input  $x$ .

$F_{\mathbf{A-out}}^x$ : captures the first time that Alice’s output becomes relatively fixed, in the following sense. If the parties continue with honest behavior from such a point in the protocol, and Alice has input  $x$ , then Alice has only one likely output, no matter what Bob’s input is.

We define such a frontier for every input  $x$ . We also define analogous frontiers with respect to Bob.

We have already defined  $\Pr_\pi[\cdot | xy]$  notation with respect to an honest execution of the protocol on inputs  $x, y$ . Since  $F_{\mathbf{A-ext}}^x$  refers to probabilities in an *ideal-model* interaction, we introduce notation to differentiate between the probabilities in real and ideal interactions. We write  $\Pr_{\mathbf{A-sim}}[\cdot | xy]$  to refer to probabilities induced by an ideal-model interaction among malicious Alice running the protocol honestly on input  $x$ , the simulator for corrupt Alice, and ideal honest Bob with input  $y$ .  $\Pr_{\mathbf{B-sim}}$  is defined analogously.

While frontiers have been used before to prove lower bounds about UC protocols [MPR09, MOPR11], one novel aspect of our approach is to define a frontier explicitly in terms of the *simulator’s behavior*. This choice appears to simplify the technical arguments about frontiers happening in a particular order. Previous work defined frontiers only in terms of the protocol’s real-interaction.

**Definition 6.** *At some point in an ideal interaction between corrupt Alice and the simulator, the simulator will at some point “extract” by sending an input to the ideal  $f$ . Define:*

$$\sigma_{\mathbf{A}}(t, x) \stackrel{\text{def}}{=} \Pr_{\mathbf{A-sim}}[\text{simulator has already extracted} | txy]$$

That is,  $\sigma_{\mathbf{A}}(t, x)$  is the probability that the simulator has extracted, given that the transcript so far is  $t$ . We define  $\sigma_{\mathbf{B}}$  analogously.

Note that before the simulator extracts, its view is perfectly independent of  $y$  in the ideal interaction. Its decision to extract, and hence the probability  $\sigma_{\mathbf{A}}(t, x)$ , depends only on  $x$  and not on  $y$ .

Note that as the transcript evolves, the probability of extraction cannot change as a result of a message sent by Alice. It can only change as a result of a message generated by the simulator, hence an output of  $g$  or a simulated Bob-message.

**Definition 7.** *Given a secure protocol  $\pi$  with security error  $\varepsilon$ , define the following for all inputs  $x, y$ :*

$$\begin{aligned} F_{\mathbf{A-ext}}^x &= \{t \mid \sigma_{\mathbf{A}}(t, x) > 4\sqrt{\varepsilon}\} \\ F_{\mathbf{B-ext}}^y &= \{t \mid \sigma_{\mathbf{B}}(t, y) > 4\sqrt{\varepsilon}\} \\ F_{\mathbf{A-out}}^x &= \{t \mid \forall y, y' : f(x, y) \neq f(x, y') \Rightarrow \min \left\{ \frac{\Pr_\pi[\text{out } f(x, y) | txy]}{\Pr_\pi[\text{out } f(x, y') | txy']} \right\} < 1 - \sqrt{\varepsilon}\} \\ F_{\mathbf{B-out}}^y &= \{t \mid \forall x, x' : f(x, y) \neq f(x', y) \Rightarrow \min \left\{ \frac{\Pr_\pi[\text{out } f(x, y) | txy]}{\Pr_\pi[\text{out } f(x', y) | txy']} \right\} < 1 - \sqrt{\varepsilon}\} \end{aligned}$$

Here  $\Pr_\pi[\text{out } z | txy]$  refers to the probability that honest parties output  $z$  when starting the protocol at partial transcript  $t$  and running honestly with inputs  $x$  and  $y$ .

To understand  $F_{A\text{-out}}^x$ , observe that for  $t \in F_{A\text{-out}}^x$  there is at most one output that can be induced with probability at least  $1 - \sqrt{\varepsilon}$ . It may be the case that no valid output can be induced with this probability, in which case only  $\perp$  output is likely from starting point  $t$ .

Note that if  $\varepsilon$  is a negligible function of the security parameter, then  $\sqrt{\varepsilon}$  is a larger function but also still negligible.

### 5.3 Properties of the Frontiers

We now show that, roughly speaking, all the frontiers that we have defined must occur simultaneously, with overwhelming probability. Proofs for the lemmas in this section are given in [Appendix C](#). Note that all lemmas hold with the roles of Alice and Bob reversed.

**Lemma 8.** *For all  $x, y$ :  $\Pr_{\pi}[F_{A\text{-ext}}^x < F_{B\text{-out}}^y \mid xy] < 2\sqrt{\varepsilon}$ .*

*sketch.* A partial transcript  $t \in F_{A\text{-ext}}^x \setminus F_{B\text{-out}}^y$  represents a situation where there is reasonable probability that the simulator would have extracted from Alice in the ideal-interaction ( $t \in F_{A\text{-ext}}^x$ ), but in the real-interaction Alice can still induce two different outputs for Bob, each with good probability ( $t \notin F_{B\text{-out}}^y$ ). Intuitively, the simulator has extracted prematurely. This event should be rare.  $\square$

Next we show that  $F_{A\text{-out}}^x$  is a point at which the honest parties can predict their eventual output.

**Definition 9.** *Fix  $x$  and let  $t \in F_{A\text{-out}}^x$ . Then there is at most one value  $z$  such that  $\exists y : \Pr_{\pi}[\text{out } z \mid xy t] > 1 - \sqrt{\varepsilon}$ . Let  $\text{guess}_A(t, x)$  denote this value  $z$ , and note that the value could be  $\perp$ . We extend the notation  $\text{guess}_A(t, x) = \perp$  in the case that  $t \notin F_{A\text{-out}}^x$ .*

**Lemma 10.** *For  $z \neq \perp$  define  $G^z = \{t \mid \text{guess}_A(t, x) = z\}$ . Then for all  $x, y$ :  $\Pr_{\pi}[G^{f(x,y)} \mid xy] > 1 - \varepsilon/2$ . Intuitively, upon reaching  $F_{A\text{-out}}^x$ , Alice can predict her eventual output with error at most  $\varepsilon/2$ .*

**Lemma 11.** *For all  $x, y$ , if  $x$  is not a unilateral input for  $f$ , then  $\Pr_{\pi}[F_{A\text{-out}}^x < F_{A\text{-ext}}^x \mid xy] < 16\varepsilon$ .*

*Sketch.* A partial transcript  $t \in F_{A\text{-out}}^x \setminus F_{A\text{-ext}}^x$  represents a situation where Alice can predict what the output will be ( $t \in F_{A\text{-out}}^x$ ), but the simulator probably has not extracted yet ( $t \notin F_{A\text{-ext}}^x$ ). This event should be rare, since in the ideal interaction Alice can gain no information about the  $f$ -output before the simulator extracts (assuming  $x$  is not a unilateral input, so that the output indeed depends on Bob's input).  $\square$

**Lemma 12.** *For all  $x, y$ , if  $y$  is not a unilateral input, then  $\Pr_{\pi}[F_{B\text{-out}}^y < F_{A\text{-out}}^x \mid xy] < 18\sqrt{\varepsilon}$ .*

*Sketch.* This follows from the fact that if  $F_{B\text{-out}}^y < F_{A\text{-out}}^x$  then either  $F_{B\text{-out}}^y < F_{B\text{-ext}}^y$  or  $F_{B\text{-ext}}^y < F_{A\text{-out}}^x$ , both of which are negligibly likely from [Lemmas 8 & 11](#).  $\square$

**Lemma 13.** *For all  $x, y', y$ , none of them unilateral,  $\Pr_{\pi}[F_{B\text{-out}}^{y'} < F_{B\text{-out}}^y \mid xy] < 42\sqrt{\varepsilon}$ .*

*Sketch.* If  $F_{B\text{-out}}^{y'} < F_{B\text{-out}}^y$  then either  $F_{B\text{-out}}^{y'} < F_{A\text{-ext}}^x$  or  $F_{A\text{-ext}}^x < F_{B\text{-out}}^y$ .

We can argue that the first case  $F_{B\text{-out}}^{y'} < F_{A\text{-ext}}^x$  would be negligibly likely, if the parties run honestly on inputs  $x, y'$ . Unfortunately here we are using input  $y$  for Bob. But consider the ideal interaction with corrupt Alice. We are interested in an event in which the simulator is not likely to have extracted from Alice ( $t \notin F_{A\text{-ext}}^x$ ). Conditioned on the simulator not extracting, the protocol transcript is independent of Bob's input. Hence whatever is unlikely with input  $y'$  for Bob is also unlikely with input  $y$  for Bob.

The second case  $F_{A\text{-ext}}^x < F_{B\text{-out}}^y$  is negligibly likely by [Lemma 8](#).  $\square$

## 5.4 Securely Truncating a Protocol

**Lemma 14.** *Let  $\pi$  be a secure protocol for  $f$  in the  $g$ -hybrid model. Define  $\pi'$  to be the following:*

- *On input  $x$  for Alice and  $y$  for Bob, both parties run  $\pi$  honestly on their given inputs.*
- *When the protocol transcript  $t$  reaches  $F_{A\text{-out}}^{\tilde{x}}$  for any  $\tilde{x}$ , or reaches  $F_{B\text{-out}}^{\tilde{y}}$  for any  $\tilde{y}$ , the parties terminate the protocol.*
- *Alice outputs  $\text{guess}_A(t, x)$  and Bob outputs  $\text{guess}_B(t, y)$ .*

*Then the truncated protocol  $\pi'$  is also a secure protocol for  $f$ .*

*Proof.* Let  $\varepsilon$  denote the simulation error of  $\pi$ . First, we argue that  $\pi'$  is correct. Alice's output is  $\text{guess}_A(t, x)$ , which differs from the correct answer  $f(x, y)$  only in the following events:

- $t \notin F_{A\text{-out}}^x$  because the protocol reached  $F_{A\text{-out}}^{x'}$  and terminated strictly before reaching  $F_{A\text{-out}}^x$  for  $x' \neq x$ . By [Lemma 12](#), this can happen only with probability  $O(\sqrt{\varepsilon})$ .
- $t \notin F_{A\text{-out}}^x$  because the protocol reached  $F_{B\text{-out}}^y$  and terminated strictly before reaching  $F_{A\text{-out}}^x$ . By [Lemma 13](#), this can happen only with probability  $O(\sqrt{\varepsilon})$ .
- $t \in F_{A\text{-out}}^x$  but  $\text{guess}_A(t, x) \neq f(x, y)$ . By [Lemma 10](#), this can only happen with probability  $O(\varepsilon)$ .

As for security, the only difference between  $\pi$  and  $\pi'$  is that  $\pi'$  truncates early based on some condition. But this condition is public and *independent of either party's private inputs*. Hence the simulation for  $\pi'$  works as follows. It simply runs the simulator for  $\pi$  but terminates the protocol when the transcript reaches the public termination condition.

Overall  $\pi'$  is a secure protocol with negligible simulation error  $O(\sqrt{\varepsilon})$ . □

Observe that the new protocol  $\pi'$  has the “instantaneous” property discussed in [Section 2](#). Importantly for our purposes in the next section, with overwhelming probability  $1 - O(\sqrt{\varepsilon})$  the protocol terminates on a transcript that is both in  $F_{A\text{-out}}^x$  and  $F_{B\text{-out}}^y$ . Such a transcript must end with a message produced by the simulator in both ideal interactions (i.e., when either party is corrupt). Hence the last protocol message must be an output of  $g$ , with overwhelming probability.

## 6 Collapsing Protocols to a Single Call to $g$

We complete our main theorem with the following lemmas.

**Lemma 15.** *If, for incomplete and non-unilateral  $f$  and  $g$ ,  $f \sqsubseteq g$  via a protocol with strict upper bound on number of rounds  $r = O(\log \kappa)$ , then  $f$  embeds in  $g$ . This proves (1)  $\Rightarrow$  (4) of [Theorem 1](#) (stated in [Section 2](#)).*

*Sketch.* The full proof is in [Appendix D](#). Without loss of generality (from [Lemma 14](#)) the last step in  $\pi$  (in particular, the action in final round  $r$ ) is a call to  $g$  with overwhelming probability.

We consider two cases. Consider a call to  $g$  that happens in the last round, following some partial transcript  $t$ . Imagine a new protocol where the parties simply “fast-forward” directly to this  $g$ -call by behaving as if the transcript so far was  $t$ . The result is a protocol consisting of a single call to  $g$ . If any call to  $g$  yields a *secure* protocol for  $f$  in this way, then we are done (we in fact have a *1-round* protocol for  $f$ ).

In the other case, there may be no call to  $g$  during the final round of  $\pi$  that yields a secure protocol for  $f$  in this way. Intuitively, every time the protocol runs for the full  $r$  rounds there would

have been a successful attack on the final call to  $g$ ! Hence it must be negligibly unlikely that  $\pi$  would ever run for  $r$  rounds. We show that, in this case, truncating  $\pi$  after  $r - 1$  rounds results in a secure protocol for  $f$ .

We can repeatedly apply this argument at most  $r - 1$  times until we are guaranteed to obtain a 1-round protocol demonstrating  $f \sqsubseteq_1 g$ . The parameters are such that after truncating  $r - 1$  rounds, the resulting protocol has simulation error  $c^{r-1}\sqrt{\varepsilon}$  for some constant  $c$ . Such a protocol is secure as long as  $r = O(\log \kappa)$ , since  $c^{O(\log \kappa)}\sqrt{\varepsilon} = \text{poly}(\kappa)\sqrt{\varepsilon}$ , which is negligible.  $\square$

**Corollary 16.** *If  $f \sqsubseteq g$  via a deterministic protocol (of any number of rounds) then  $f$  embeds in  $g$ . This proves (2)  $\Rightarrow$  (4) of [Theorem 1](#) (stated in [Section 2](#)).*

*Proof.* Deterministic protocols have zero simulation error (without loss of generality). Therefore, the same reasoning as in the previous proof applies but without any error accumulating with each round.  $\square$

## 7 Tightness of the Characterization

In this section we discuss why our main characterization does not extend (without modification) to consider unilateral functions or superlogarithmic-round, randomized protocols.

### 7.1 Unilateral Functions

In [Figure 2](#) we give  $f$  and  $g$  which are unilateral (similar examples can be constructed in which  $g$  is non-unilateral). Bob is the column-player and thus has 2 unilateral inputs labeled  $B$  and  $C$ .

First, we argue that  $f \not\sqsubseteq_1 g$ . Suppose for sake of contradiction that such a protocol exists. Consider the simulator for a corrupt Bob who chooses his  $f$ -input uniformly at random and runs the protocol semi-honestly. The only message that the simulator sees is Bob's input to  $g$ , after which the simulator must extract an output to send to  $f$ . The simulator gets only one bit of information about Bob's input, while there are 3 possibilities for it to send to  $f$ . It follows that with constant probability the simulator must extract the wrong input, and this error will be evident in the output of  $f$ .

However, there is a simple protocol for  $f$  using  $g$ : Alice sends her  $f$  input directly to  $g$ . If Bob has  $f$ -input  $A$ , he should choose  $g$ -input  $A'$ . In this case, the parties will see that the  $g$ -output is in  $\{0, 1\}$  and they terminate with this as their  $f$ -output. Otherwise, if Bob has  $f$ -input  $B$  or  $C$ , he should choose  $g$ -input  $B'$ . In this case, the parties will see that the  $g$  output is 2, and then Alice will wait for Bob to send a plain message containing either "2" or "3." Alice takes this message to be her output.

It is simple to see that this protocol is secure against a malicious Alice. For a malicious Bob, the simulator does the following. If Bob chooses  $g$ -input  $A'$ , then the simulator extracts Bob's ideal  $f$ -input as  $A$  and simulates the  $g$ -output to equal the ideal  $f$ -output. If Bob chooses  $g$ -input  $B'$ , then the simulator gives 2 as the simulated  $g$ -output, then waits for a message from Bob (either "2" or "3") and uses this as the extracted ideal  $f$ -input. The reason the simulation is secure is that in the second case (Bob chooses  $g$ -input  $B'$ ), the fact that this is a unilateral input means that the simulator doesn't need to know Alice's input to perfectly simulate the  $g$ -output. Hence the simulator can delay extraction until the second protocol message, where intuitively Bob resolves which unilateral input he has.

Hence, we have  $f \sqsubseteq g$  via a protocol consisting of a single call to  $g$ , plus (in some cases) one extra message. It is a deterministic, constant-round protocol, and yet  $f \not\sqsubseteq_1 g$ . This example shows that our classification does not extend to unilateral functions.

<i>A</i>	<i>B</i>	<i>C</i>
0	2	3
1	2	3

*f*

<i>A'</i>	<i>B'</i>
0	2
1	2

*g*

A	A	B
C	D	D
C	E	E

*f*

A	1	2	A	1	2	B	1	2
3	A	4	3	A	4	4	B	3
6	5	A	6	5	A	5	6	B
C	1	2	D	1	2	D	1	2
3	C	4	3	D	4	4	D	3
6	5	C	6	5	D	5	6	D
C	1	4	E	1	4	4	1	E
6	C	2	6	E	2	E	6	2
3	5	C	3	5	E	5	E	3

*g*

Figure 2: Unilateral functions violating the main theorem.

Figure 3: Functions violating the main theorem via a superlogarithmic-round protocol.

## 7.2 Deterministic / Logarithmic-Round Protocols

Consider the functions  $f$  and  $g$  in Figure 3. We first claim that  $f$  does not embed in  $g$ . Any embedding would map 3  $f$ -columns into 3 distinct  $g$ -columns.<sup>3</sup> For any 3 columns of  $g$ , there exists a row for which these columns have distinct entries – this is simple (albeit time-consuming) to verify. However, there is no row in  $f$  that has three distinct values. Hence the embedding would contradict rule 2a of the embedding definition. Concretely, any candidate protocol for  $f \sqsubseteq_1 g$  would allow a corrupt row-player to learn the column-player’s input in its entirety, which is not allowed by  $f$ .

However, there is a protocol for  $f$  that uses  $g$ . We group the rows and columns of  $g$  into groups of three, as distinguished by the dotted lines in the figure. Associate the first row of  $f$  with the first row group of  $g$ , etc. Similarly, associate the first column of  $f$  with the first column group of  $g$ , etc. The protocol for  $f$  is as follows:

- Alice chooses a  $g$ -input from the row group associated with her  $f$ -input, uniformly at random.
- Bob chooses a  $g$ -input from the column group associated with his  $f$ -input, uniformly at random.
- They call  $g$  with their selected  $g$ -inputs.
- If the output of the  $g$ -call was in  $\{A, B, C, D, E\}$ , terminate the protocol with that output. Otherwise, repeat (with fresh random choices for the  $g$ -inputs).

The correctness of this protocol is clear. By only sending  $g$ -inputs in the group associated with their  $f$ -inputs, each party restricts any terminating output of  $g$  to be one that was possible given their  $f$ -input.

To see that the protocol is secure, consider the following simulation. Suppose corrupt Alice chooses some  $g$ -input (row). With probability  $1/3$ , the simulator decides that the protocol will terminate at this round. It converts the  $g$ -input to an  $f$ -input (according to its row group), sends that  $f$ -input to the ideal  $f$ , then simulates the  $g$ -output as the ideal  $f$ -input. With probability  $2/3$ , the simulator decides that the protocol will continue. Note that in any row, there are 2 non-terminal  $g$ -outputs (for example, in the second row only 3 and 4 are possible), which are equally

<sup>3</sup>Perhaps columns are mapped to rows if the roles of Alice and Bob are swapped during the embedding. The analysis is the same for this scenario.

likely no matter which column group Bob has selected. The simulator simply chooses one of these two with equal probability as the simulated  $g$ -output. Then the same process repeats.

The parties' inputs will "match" by giving a terminal output with probability  $1/3$ , meaning that the expected number of rounds is 3. The probability that the protocol continues for at least  $r$  rounds is  $(2/3)^r$ . We can get a protocol with a strict upper bound on round complexity by having the parties simply abort after some limit  $r$  number of rounds. If we set this limit as  $r(\kappa) = \omega(\log \kappa)$ , then the correctness of the protocol suffers by an amount  $(2/3)^{\omega(\log \kappa)} = \kappa^{-\omega(1)}$ , which is negligible. However, the simulation is still perfect, and the protocol is secure.

In summary,  $f \sqsubseteq g$  via a randomized, (worst-case) superlogarithmic-round protocol, but  $f$  does not embed in  $g$  and so  $f \not\sqsubseteq g$  via any deterministic protocol or any strict logarithmic-round protocol. Note that our model considers protocols that make only *sequential* calls to  $g$ , and hence this example is a round-complexity lower bound for such protocols. In [Appendix E](#) we discuss extending this lower bound to the more realistic model that allows parallel calls to  $g$  in each round. We emphasize that the restriction to sequential  $g$ -calls is without loss of generality with respect to the security of the protocol (synchronous parallel calls cannot be *enforced* in the UC model).

## 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.
- [BM04] Amos Beimel and Tal Malkin. A quantitative approach to reductions in secure computation. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 238–257. Springer, Heidelberg, February 2004.
- [BR06] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006.
- [Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.
- [DM99] Yevgeniy Dodis and Silvio Micali. Lower bounds for oblivious transfer reductions. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 42–55. Springer, Heidelberg, May 1999.
- [IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 572–591. Springer, Heidelberg, August 2008.
- [Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *20th ACM STOC*, pages 20–31. ACM Press, May 1988.
- [Kil89] Joe Kilian. *Uses of Randomness in Algorithms and Protocols*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1989.
- [Kil91] Joe Kilian. A general completeness theorem for two-party games. In *23rd ACM STOC*, pages 553–560. ACM Press, May 1991.

- [Kil00] Joe Kilian. More general completeness theorems for secure two-party computation. In *32nd ACM STOC*, pages 316–324. ACM Press, May 2000.
- [KMPS14] Daniel Kraschewski, Hemanta K. Maji, Manoj Prabhakaran, and Amit Sahai. A full characterization of completeness for two-party randomized function evaluation. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 659–676. Springer, Heidelberg, May 2014.
- [KMQ11] Daniel Kraschewski and Jörn Müller-Quade. Completeness theorems with constructive proofs for finite deterministic 2-party functions. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 364–381. Springer, Heidelberg, March 2011.
- [Kus89] Eyal Kushilevitz. Privacy and communication complexity. In *30th FOCS*, pages 416–421. IEEE Computer Society Press, October / November 1989.
- [MOPR11] Hemanta K. Maji, Pichayoot Ouppaphan, Manoj Prabhakaran, and Mike Rosulek. Exploring the limits of common coins using frontier analysis of protocols. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 486–503. Springer, Heidelberg, March 2011.
- [MPR09] 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 2009*, volume 5444 of *LNCS*, pages 256–273. Springer, Heidelberg, March 2009.
- [MPR10a] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Cryptographic complexity classes and computational intractability assumptions. In Andrew Chi-Chih Yao, editor, *ICS 2010*, pages 266–289. Tsinghua University Press, January 2010.
- [MPR10b] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. A zero-one law for cryptographic complexity with respect to computational UC security. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 595–612. Springer, Heidelberg, August 2010.
- [MPR12] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. A unified characterization of completeness and triviality for secure function evaluation. In Steven D. Galbraith and Mridul Nandi, editors, *INDOCRYPT 2012*, volume 7668 of *LNCS*, pages 40–59. Springer, Heidelberg, December 2012.
- [PR08] Manoj Prabhakaran and Mike Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 262–279. Springer, Heidelberg, August 2008.

## A UC Security Recap

A 2-party SFE task is a deterministic function  $f : X \times Y \rightarrow Z$ . We identify  $f$  with the following ideal functionality: The functionality waits for input  $x \in X$  from Alice and input  $y \in Y$  from Bob. If no party is corrupt, the functionality gives output  $f(x, y)$  to both parties. If any party is corrupt, the functionality gives  $f(x, y)$  to the adversary and waits for a command  $(\text{DELIVER}, b)$  from the adversary, where  $b \in \{0, 1\}$ . If  $b = 0$ , the functionality gives output  $\perp$  to the honest party; if  $b = 1$ ,

the functionality gives  $f(x, y)$  to the honest party. Hence, we consider **security with abort** — the functionality does not guarantee fairness or output delivery.

We assume basic familiarity with the UC framework, but briefly review the main concepts (specialized here for 2-party SFE). An execution in the framework begins with an *environment*  $\mathcal{Z}$  (an arbitrary interactive TM) that chooses inputs for both of the *parties*. The parties interact with each other, and an ideal functionality for some function  $g$ , according to the protocol. The parties eventually give an output to the environment, who outputs a single bit. Throughout the entire interaction, there is an *adversary*  $\mathcal{A}$  (an arbitrary interactive TM) who interacts arbitrarily with the environment. The adversary may also choose to *corrupt* one of the parties, which causes the party to come under complete control of the adversary. In that case, the party may deviate from the protocol. In this work we consider only **static** corruption, where the adversary must choose to corrupt a party before the protocol begins.

Given the description above,  $\text{EXEC}[\pi, g, \mathcal{A}, \mathcal{Z}, 1^\kappa]$  denotes the probability that the environment outputs 1, when  $\kappa$  is the security parameter of the protocol.

A particular protocol of interest is the dummy protocol  $\pi_{\text{dummy}}$ . In this protocol, each party receives an input from the environment and sends it directly to the ideal functionality. When the ideal functionality delivers an output, the party gives it directly to the environment as output.

**Definition 17.** We say  $f$  reduces to  $g$ , and write  $f \sqsubseteq g$ , if there exists a protocol  $\pi$  such that for all  $\mathcal{A}$  there exists a  $\mathcal{S}$  such that for all environments  $\mathcal{Z}$ , we have:

$$\left| \text{EXEC}[\pi, g, \mathcal{A}, \mathcal{Z}, 1^\kappa] - \text{EXEC}[\pi_{\text{dummy}}, f, \mathcal{S}, \mathcal{Z}, 1^\kappa] \right| \text{ is negligible in } \kappa$$

We write  $f \sqsubseteq_1 g$  if furthermore  $\pi$  has the following property: the parties make only one call to  $g$  and exchange no other messages.

In the execution of  $\pi_{\text{dummy}}$  we can think of  $\mathcal{S}$  as simulating protocol  $\pi$  with  $\mathcal{A}$ . The security arises from the fact that the simulator only communicates with the honest party via a single call to ideal  $f$ . This means that everything prior to the call  $\mathcal{S}$  makes to ideal  $f$  must be independent of the honest party’s input, and everything after that call must not affect the honest party’s output. We call this event the *extraction*, as  $\mathcal{S}$  must “extract” enough information about the input of  $\mathcal{A}$  in order to correctly call  $f$ .

## B Proofs for the Reducibility Characterization

**Lemma** (Restatement of Lemma 5). *For non-unilateral  $f$ , if  $f \sqsubseteq_1 g$  via a deterministic protocol with simulation error less than 1, then  $f$  embeds in  $g$ .*

*Proof.* During the deterministic protocol, Alice and Bob both map their  $f$ -inputs to  $g$ -inputs and then immediately terminate the protocol, meaning that they were each able to map the  $g$ -output to the same  $f$ -output. The input maps are  $A$  and  $B$  in the embedding. The output map is  $C$  in the embedding.

By symmetry, we only consider security against a malicious Alice.

Clearly,  $A$  is injective. If it is not, then choose some pair  $(x, x')$  where  $A(x) = A(x')$ . Choose  $y$  so that  $f(x, y) \neq f(x', y)$  — since  $f$  is not unilateral, such an input exists. Consider two environments — they ask the parties to honestly run the protocol with inputs  $x, y$  and  $x', y$  respectively, and return 1 if the output is correct. In the real world, in both cases the parties will return the same output, so one of them is incorrect. Therefore, one of the environments has simulation error 1.



Because this protocol is UC-secure, there must be some method by which the simulator takes the parties'  $g$ -inputs and translate them to  $f$ -inputs upon extraction. Call these mappings for Alice and Bob  $\widehat{A}$  and  $\widehat{B}$ , respectively. It suffices to show that these mappings satisfy 2a and 2b of [Definition 3](#). We show that if the mappings violate [Definition 3](#), then the simulation error of the protocol is 1.

**(2a)** Assume that  $\widehat{A}$  violates part 2a. That is, there exists an  $\widehat{x}$  where it is not the case that  $g(\widehat{x}, B(\cdot))$  leaks no more than  $f(\widehat{A}(\widehat{x}), \cdot)$ . In particular, there is some pair  $(y, y')$  where  $f(\widehat{A}(\widehat{x}), y) = f(\widehat{A}(\widehat{x}), y')$  but  $g(\widehat{x}, B(y)) \neq g(\widehat{x}, B(y'))$ .

Consider two environments:

- Adversary Alice uses input  $\widehat{x}$ , honest Bob uses input  $y$ , return 1 if Alice's view is  $g(\widehat{x}, B(y))$ .
- Adversary Alice uses input  $\widehat{x}$ , honest Bob uses input  $y'$ , return 1 if Alice's view is  $g(\widehat{x}, B(y'))$ .

In the real world, both environments output 1 with probability 1. In the ideal world, the simulator's view in both environments is  $f(\widehat{A}(\widehat{x}), y) = f(\widehat{A}(\widehat{x}), y')$ . Since the simulator is deterministic, it must give the same simulated  $g$ -output for both environments. However,  $g(\widehat{x}, B(y)) \neq g(\widehat{x}, B(y'))$ , so in at least one of the environments the probability of outputting 1 is 0. Therefore, the simulation error is 1.

**(2b)** Assume that  $\widehat{A}$  violates part 2b. Then there exists some  $\widehat{x}$  and  $y$  where  $C(g(\widehat{x}, B(y))) \notin \{f(\widehat{A}(\widehat{x}), y), \perp\}$ . Consider the environment in which corrupt Alice uses  $g$ -input  $\widehat{x}$  and honest Bob uses input  $y$ , and the environment outputs 1 if Bob's output is  $C(g(\widehat{x}, B(y)))$ . The environment outputs 1 with probability 1 in the real world. But in the ideal world, Bob's output is either  $f(\widehat{A}(\widehat{x}), y)$  or  $\perp$ . This environment demonstrates a simulation error of 1.  $\square$

## C Proofs for Frontier Properties

**Lemma** (Restatement of [Lemma 8](#)). *For all  $x, y$ :  $\Pr_{\pi}[F_{A\text{-ext}}^x < F_{B\text{-out}}^y \mid xy] < 2\sqrt{\varepsilon}$ .*

*Proof.* Let  $\text{bad} = \text{first}(F_{A\text{-ext}}^x \setminus F_{B\text{-out}}^y)$ , whose probability we wish to bound. A partial transcript  $t \in \text{bad}$  represents a situation where there is reasonable probability that a simulator would have extracted an effective input for Alice ( $t \in F_{A\text{-ext}}^x$ ), but in the protocol Alice can still induce two different outputs for Bob, each with good probability ( $t \notin F_{B\text{-out}}^y$ ). Intuitively, the simulator has extracted prematurely. This event should be rare.

Consider the following strategy for corrupt Alice and environment:

- Run the protocol with input  $y$  for honest Bob, and Alice initially behaving semi-honestly with input  $x$ .
- If the protocol transcript avoids  $\text{bad}$ , then the adversary gives up and the environment outputs 0.
- Otherwise, when the partial transcript reaches  $t \in \text{bad}$  for the first time, then the properties of  $\text{bad}$  guarantee that there are two values  $x_0, x_1$  such that  $f(x_0, y) \neq f(x_1, y)$  and  $\Pr_{\pi}[\text{out } f(x_c, y) \mid tx_cy] \geq 1 - \sqrt{\varepsilon}$  for both  $c \in \{0, 1\}$ .
- The adversary sends  $x_0, x_1$  to the environment, who chooses a random  $c \leftarrow \{0, 1\}$ .

- The adversary switches strategies to run the protocol honestly with input  $x_c$ . The environment outputs 1 if Bob's eventual output is  $f(x_c, y)$ . Otherwise the environment outputs 0.

Let  $\text{succ}$  denote the event that the environment outputs 1.

In the real interaction, the environment outputs 1 only when the transcript hits  $\text{bad}$  and the adversary is successful in forcing Bob's output, which happens with probability at least  $1 - \sqrt{\varepsilon}$  by the properties of  $\text{bad}$ . So:

$$\Pr_{\pi}[\text{succ}] \geq \Pr_{\pi}[\text{bad}|xy](1 - \sqrt{\varepsilon})$$

In the ideal interaction (between the adversary and simulator), Bob's output is now determined differently, as the output of the ideal  $f$ . There are two ways the environment outputs 0: (1) when the simulated transcript avoids  $\text{bad}$ ; (2) when the transcript reaches  $\text{bad}$  but the simulator has already extracted. In the latter case, the environment's choice of  $c$  is independent of the simulator's extraction, so with further probability at least  $1/2$  the honest Bob will not output  $f(x_c, y)$ . So:

$$\Pr_{\text{A-sim}}[\text{succ}] \leq (1 - (4\sqrt{\varepsilon})/2) \Pr_{\text{A-sim}}[\text{bad}|xy]$$

From the security of the protocol:

$$\begin{aligned} |\Pr_{\pi}[\text{succ}] - \Pr_{\text{A-sim}}[\text{succ}]| &< \varepsilon \\ |\Pr_{\pi}[\text{bad}|xy] - \Pr_{\text{A-sim}}[\text{bad}|xy]| &< \varepsilon \end{aligned}$$

Hence:

$$\begin{aligned} \varepsilon &> \Pr_{\pi}[\text{succ}] - \Pr_{\text{A-sim}}[\text{succ}] \\ &\geq \Pr_{\pi}[\text{bad}|xy](1 - \sqrt{\varepsilon}) - (1 - 2\sqrt{\varepsilon}) \Pr_{\text{A-sim}}[\text{bad}|xy] \\ &\geq \Pr_{\pi}[\text{bad}|xy](1 - \sqrt{\varepsilon}) - (1 - 2\sqrt{\varepsilon})(\Pr_{\pi}[\text{bad}|xy] + \varepsilon) \\ &= \Pr_{\pi}[\text{bad}|xy]\sqrt{\varepsilon} - \varepsilon(1 - 2\sqrt{\varepsilon}) \\ &> \Pr_{\pi}[\text{bad}|xy]\sqrt{\varepsilon} - \varepsilon \end{aligned}$$

Solving for  $\Pr_{\pi}[\text{bad}|xy]$ :

$$\Pr_{\pi}[\text{bad}|xy] < \frac{2\varepsilon}{\sqrt{\varepsilon}} = 2\sqrt{\varepsilon}$$

□

**Lemma** (Restatement of **Lemma 10**). *For  $z \neq \perp$  define  $G^z = \{t \mid \text{guess}_{\text{A}}(t, x) = z\}$ . Then for all  $x, y$ :  $\Pr_{\pi}[G^{f(x,y)}|xy] > 1 - \varepsilon/2$ . Intuitively, upon reaching  $F_{\text{A-out}}^x$ , Alice can predict her eventual output with error at most  $\varepsilon/2$ .*

*Proof.* Define  $\text{bad} = F_{\text{A-out}}^x \setminus G^{f(x,y)}$ . Intuitively, these are the places in the protocol where  $\text{guess}_{\text{A}}(t, x) \neq f(x, y)$ .

From the correctness of the protocol, we have:

$$\begin{aligned} \varepsilon &> \Pr_{\pi}[\text{output not } f(x, y)|xy] \\ &\geq \sum_{t \in \text{first}(\text{bad})} \Pr_{\pi}[t|xy] \Pr_{\pi}[\text{out } \text{guess}_{\text{A}}(t, x)|txy] \\ &\geq \sum_{t \in \text{first}(\text{bad})} \Pr_{\pi}[t|xy](1 - \sqrt{\varepsilon}) = (1 - \Pr_{\pi}[G^{f(x,y)}|xy])(1 - \sqrt{\varepsilon}) \end{aligned}$$

Solving for the probability expression:

$$\Pr_\pi[G^{f(x,y)}|xy] \geq 1 - \frac{\varepsilon}{1-\sqrt{\varepsilon}} > 1 - \varepsilon/2$$

□

**Lemma** (Restatement of [Lemma 11](#)). *For all  $x, y$ , if  $x$  is not a unilateral input for  $f$ , then  $\Pr_\pi[F_{A\text{-out}}^x < F_{A\text{-ext}}^x | xy] < 16\varepsilon$ .*

*Proof.* Let  $\text{bad} = \text{first}(F_{A\text{-out}}^x \setminus F_{A\text{-ext}}^x)$ , whose probability we wish to bound. A partial transcript  $t \in \text{bad}$  represents a situation where Alice can predict what the output will be ( $t \in F_{A\text{-out}}^x$ ), but the simulator probably has not extracted yet ( $t \notin F_{A\text{-ext}}^x$ ). This event should be rare, since in the ideal world Alice can gain no information about the  $f$ -output before the simulator extracts.

Let  $x, y$  be given in the premise of the lemma. Since  $x$  is not a unilateral input, let  $y'$  be such that  $f(x, y) \neq f(x, y')$ . Consider the following interaction with a corrupt Alice and environment:

- Alice initially runs the protocol honestly with input  $x$ . The environment randomly chooses input  $y^* \leftarrow \{y, y'\}$  for Bob. If the transcript avoids  $\text{bad}$  then the adversary gives up and the environment outputs 0.
- Otherwise, if the partial transcript reaches  $t \in \text{bad}$ , there is a unique  $z = \text{guess}_A(t, x)$  such that  $\Pr_\pi[\text{out } z | tx] > 1 - \sqrt{\varepsilon}$ . The adversary reports  $z$  to the environment.
- The environment outputs 1 if  $z = f(x, y^*)$ .

Let  $\text{succ}$  denote the probability that the environment outputs 1.

In the real interaction, the environment outputs 0 only if the transcript avoids  $\text{bad}$  or if  $\text{guess}_A(t, x)$  is incorrect. By the union bound and [Lemma 10](#),

$$\Pr_\pi[\neg \text{succ}] \leq \Pr_\pi[\neg \text{bad} | xy^*] + \Pr_\pi[\neg G^{f(x,y^*)} | xy^*] \leq 1 - \Pr_\pi[\text{bad} | xy^*] + \varepsilon/2$$

In the ideal interaction, the environment outputs 0 in the following (mutually exclusive) scenarios: (1) the simulated transcript avoids  $\text{bad}$ ; (2) the transcript reaches  $\text{bad}$  and the simulator has not yet extracted. In the latter case, the adversary's view is independent of the environment's choice of  $y^*$ , and so the environment outputs 0 with further probability at least  $1/2$ . Hence:

$$\begin{aligned} \Pr_{A\text{-sim}}[\neg \text{succ}] &\geq \Pr_{A\text{-sim}}[\neg \text{bad} | xy^*] + \Pr_{A\text{-sim}}[\text{bad} \wedge \text{no extract} | xy^*] / 2 \\ &\geq 1 - \Pr_{A\text{-sim}}[\text{bad} | xy^*] + \Pr_{A\text{-sim}}[\text{bad} | xy^*] (1 - 4\sqrt{\varepsilon}) / 2 \\ &= 1 - \Pr_{A\text{-sim}}[\text{bad} | xy^*] (\frac{1}{2} + 2\sqrt{\varepsilon}) \end{aligned}$$

Combining:

$$\begin{aligned} \varepsilon &> \Pr_\pi[\text{succ}] - \Pr_{A\text{-sim}}[\text{succ}] \\ &\geq \Pr_\pi[\text{bad} | xy^*] - \varepsilon/2 - \Pr_{A\text{-sim}}[\text{bad} | xy^*] (\frac{1}{2} + 2\sqrt{\varepsilon}) \\ &\geq \Pr_\pi[\text{bad} | xy^*] - \varepsilon/2 - (\Pr_\pi[\text{bad} | xy^*] + \varepsilon) (\frac{1}{2} + 2\sqrt{\varepsilon}) \\ &= \Pr_\pi[\text{bad} | xy^*] (\frac{1}{2} - 2\sqrt{\varepsilon}) - \varepsilon(1 - 2\sqrt{\varepsilon}) \end{aligned}$$

Solving for the probability expression:

$$\Pr_\pi[\text{bad} | xy^*] \leq \frac{\varepsilon(2 - 2\sqrt{\varepsilon})}{(\frac{1}{2} - 2\sqrt{\varepsilon})} < \frac{2\varepsilon}{1/4} = 8\varepsilon$$

Since  $\Pr_\pi[\text{bad} | xy^*]$  is the average of  $\Pr_\pi[\text{bad} | xy]$  and  $\Pr_\pi[\text{bad} | xy']$ , it follows that  $\Pr_\pi[\text{bad} | xy] < 16\varepsilon$ . □

**Lemma** (Restatement of [Lemma 12](#)). *For all  $x, y$ , if  $y$  is not a unilateral input, then  $\Pr_\pi[F_{B\text{-out}}^y < F_{A\text{-out}}^x | xy] < 18\sqrt{\varepsilon}$ .*

*Proof.* By a union bound,

$$\begin{aligned} \Pr_\pi[F_{B\text{-out}}^y < F_{A\text{-out}}^x | xy] &\leq \Pr_\pi[F_{B\text{-out}}^y < F_{B\text{-ext}}^y | xy] + \Pr_\pi[F_{B\text{-ext}}^y < F_{A\text{-out}}^x | xy] \\ &\leq 16\varepsilon + 2\sqrt{\varepsilon} < 18\sqrt{\varepsilon} \end{aligned}$$

□

Before proving [Lemma 13](#), we introduce two helper lemmas:

**Lemma 18.** *For all  $x, y$ , if neither  $x$  nor  $y$  are a unilateral input, then  $\Pr_\pi[F_{B\text{-out}}^y < F_{A\text{-ext}}^x | xy] < 34\sqrt{\varepsilon}$ .*

*Proof.* By a union bound,

$$\begin{aligned} \Pr_\pi[F_{B\text{-out}}^y < F_{A\text{-ext}}^x | xy] &\leq \Pr_\pi[F_{B\text{-out}}^y < F_{B\text{-ext}}^y | xy] \\ &\quad + \Pr_\pi[F_{B\text{-ext}}^y < F_{A\text{-out}}^x | xy] \\ &\quad + \Pr_\pi[F_{A\text{-out}}^x < F_{A\text{-ext}}^x | xy] \\ &\leq 16\varepsilon + 2\sqrt{\varepsilon} + 16\varepsilon < 34\sqrt{\varepsilon} \end{aligned}$$

□

**Lemma 19.** *Let  $F$  be any frontier in the protocol. For all  $x, y, y'$ ,*

$$\left| \Pr_\pi[F < F_{A\text{-ext}}^x | xy] - \Pr_\pi[F < F_{A\text{-ext}}^x | xy'] \right| < 6\sqrt{\varepsilon}$$

*Proof.* Let  $G = F \setminus F_{A\text{-ext}}^x$ . The main idea is that in the ideal interaction with corrupt Alice, it is unlikely that the simulator has extracted before the protocol has reached  $G$ . Conditioned on the simulator not yet extracting, the transcript is completely independent of Bob's input.

Consider running the ideal interaction and halting it when the transcript reaches either  $F$  or  $F_{A\text{-ext}}^x$ . Halting at this point is sufficient to determine whether the event  $F < F_{A\text{-ext}}^x$  happened. We obtain two interactions depending on whether Bob is given input  $y$  or  $y'$ . In the terminology of Bellare-Rogaway [[BR06](#)], these are two *identical-until-bad games*, where the “bad” event is that the simulator extracts but  $F_{A\text{-ext}}^x$  is not immediately reached. By the definition of  $F_{A\text{-ext}}^x$ , the bad event happens with probability at most  $4\sqrt{\varepsilon}$ . By the lemma in [[BR06](#)], this probability of the bad event bounds the distinguishing bias between the two games.

Then applying the security of the protocol we have:

$$\begin{aligned} \left| \Pr_\pi[F < F_{A\text{-ext}}^x | xy] - \Pr_\pi[F < F_{A\text{-ext}}^x | xy'] \right| \\ \leq \left| \Pr_{A\text{-sim}}[F < F_{A\text{-ext}}^x | xy] - \Pr_{A\text{-sim}}[F < F_{A\text{-ext}}^x | xy'] \right| + 2\varepsilon \\ < 4\sqrt{\varepsilon} + 2\varepsilon < 6\sqrt{\varepsilon} \end{aligned}$$

□

**Lemma** (Restatement of [Lemma 13](#)). *For all  $x, y', y$ , none of them unilateral,  $\Pr_\pi[F_{B\text{-out}}^{y'} < F_{B\text{-out}}^y | xy] < 42\sqrt{\varepsilon}$ .*

*Proof.* Let  $\text{bad} = \text{first}(F_{\text{B-out}}^{y'} \setminus F_{\text{B-out}}^y)$ , whose probability we wish to bound. We partition  $\text{bad}$  into two parts:  $\text{bad}_1 = \text{bad} \cap F_{\text{A-ext}}^x$  and  $\text{bad}_2 = \text{bad} \setminus F_{\text{A-ext}}^x$ .

Since  $\text{bad}_1 \subseteq F_{\text{A-ext}}^x \setminus F_{\text{B-out}}^y$  (i.e., the event  $F_{\text{A-ext}}^x < F_{\text{B-out}}^y$  is true for these transcripts), [Lemma 8](#) implies that

$$\Pr_\pi[\text{bad}_1|xy] < 2\sqrt{\varepsilon}.$$

Since  $\text{bad}_2$  happens strictly before the  $F_{\text{A-ext}}^x$  event, [Lemma 19](#) implies that

$$\left| \Pr_\pi[\text{bad}_2|xy] - \Pr_\pi[\text{bad}_2|xy'] \right| < 6\sqrt{\varepsilon}.$$

Since  $\text{bad}_2 \subseteq F_{\text{B-out}}^{y'} \setminus F_{\text{A-ext}}^x$ , [Lemma 18](#) implies that

$$\Pr_\pi[\text{bad}_2|xy'] < 34\sqrt{\varepsilon}.$$

Putting everything together, we have:

$$\begin{aligned} \Pr_\pi[\text{bad}|xy] &\leq \Pr_\pi[\text{bad}_1|xy] + \Pr_\pi[\text{bad}_2|xy] \\ &< 2\sqrt{\varepsilon} + \Pr_\pi[\text{bad}_2|xy'] + 6\sqrt{\varepsilon} \\ &< 2\sqrt{\varepsilon} + 34\sqrt{\varepsilon} + 6\sqrt{\varepsilon} \\ &= 42\sqrt{\varepsilon} \end{aligned}$$

□

## D Proofs for Collapsing Protocols

In order to collapse a protocol to a single round, we use two important properties of instantaneous protocols. First, by [Lemma 14](#) we can consider only protocols that end with a call to  $g$ . Second, by [Lemma 19](#) before the final call to  $g$  the parties' inputs do not have a noticeable effect on the distribution of transcripts.

**Lemma 20.** *For all  $f$  and  $g$  there is a constant  $c_{f,g}$  such that if  $f \sqsubseteq_1 g$  via a protocol  $\pi$  with simulation error  $\varepsilon$ , then  $f \sqsubseteq_1 g$  via a deterministic protocol  $\pi'$  with simulation error at most  $c_{f,g}\varepsilon$ .*

*Proof.* Since  $\pi$  consists of only one call to  $g$ , the only choices Alice, Bob, and the simulator can make in the protocol are:

- The mapping of Alice's  $f$ -input to her  $g$ -input
- The mapping of Bob's  $f$ -input to his  $g$ -input
- The mappings of either party's  $g$ -input to a suitable  $f$ -input in the simulator
- The mapping of the  $g$ -output to an  $f$ -output

The only ways that randomness can manifest in the protocol are in the choice of these mappings.

Let  $c_{f,g}$  be the number of possible combinations of such mappings based on these random coins. This is certainly a constant, although it is perhaps very large.

Select the mapping combination that was most likely to be chosen in  $\pi$ . Consider the deterministic protocol  $\pi'$  constructed by locking in these choices at the start of the protocol. The probability that Alice, Bob, and the simulator in  $\pi$  match the behavior of  $\pi'$  is at least  $1/c_{f,g}$ . Then, if the simulation error of  $\pi'$  is  $\delta$ , the simulation error of  $\pi$  must be at least  $\delta/c_{f,g}$ . Therefore, if  $\pi$  has simulation error  $\varepsilon$ , then  $\pi'$  must have simulation error at most  $c_{f,g}\varepsilon$ . □

Given a protocol  $\pi$  with a strict upper limit of  $r$  rounds,  $\text{trunc}(\pi, i)$  is the protocol constructed by truncating  $\pi$  after  $r - i$  rounds, outputting  $\perp$  if  $\pi$  was not finished. Note that  $\text{trunc}(\pi, 0) = \pi$ .

Let  $\mathcal{R}$  be the transcripts of  $\text{trunc}(\pi, i)$  which are  $r - i - 1$  rounds (that is, there is one action to go in the protocol) but have not terminated yet.

**Lemma 21.** *If  $\pi$  has simulation error  $\varepsilon$ , then for all  $x, y, y'$  and all  $i$ :*

$$\left| \Pr_{\text{trunc}(\pi, i)}[\mathcal{R}|xy] - \Pr_{\text{trunc}(\pi, i)}[\mathcal{R}|xy'] \right| < 6\sqrt{\varepsilon}$$

*Proof.* By Lemma 19 (in Appendix C) we know that this is true in  $\text{trunc}(\pi, 0) = \pi$ , as these  $\mathcal{R}$  transcripts are strictly before  $F_{\mathbf{A}\text{-ext}}^x$ . Truncating doesn't change simulator extraction probabilities, as the simulator for  $\text{trunc}(\pi, i)$  just runs the simulator for  $\pi$  up until the truncated transcript. Therefore, the lemma still holds with respect to  $\text{trunc}(\pi, i)$ .  $\square$

**Lemma 22.** *If a protocol  $\pi$  is not  $\varepsilon$ -secure against malicious Alice, then there is an environment ENV for  $\pi$  with the following properties:*

1. ENV chooses inputs for Bob uniformly at random
2.  $\Pr_{\pi}[\text{ENV outputs } 1] > \frac{1}{2} + \frac{\varepsilon}{4|Y|}$ .
3.  $\Pr_{\mathbf{A}\text{-sim}}[\text{ENV outputs } 1] < \frac{1}{2} - \frac{\varepsilon}{4|Y|}$

Of course, a symmetrical lemma holds for protocol  $\pi$  that is insecure against malicious Bob.

*Proof.* Take an environment  $\text{ENV}_0$  for which  $\pi$  has simulation error  $\varepsilon$ . Construct ENV as follows.

- Choose Bob's input  $y^*$  uniformly at random.
- Let  $p(y)$  be the probability that  $\text{ENV}_0$  chooses  $y$ . Let  $p_{\max}$  be the maximum  $p(y)$ .
- Flip a coin with probability  $1 - (p(y^*)/p_{\max})$ . If it comes up heads, abort and return 0.
- Otherwise, run  $\text{ENV}_0$ .

Note that  $\text{ENV} \equiv \text{ENV}_0$  conditioned on ENV not aborting. We abort with probability at most  $(|Y| - 1)/|Y|$  (as we never abort for the  $y$  where  $p(y) = p_{\max}$ ).  $(|Y| - 1)/|Y|$  is a constant. The simulation error of ENV is therefore at least  $\varepsilon/|Y|$ .

At this point, possibly invert the output of ENV such that  $\Pr_{\pi}[\text{ENV outputs } 1]$  is greater than  $\Pr_{\mathbf{A}\text{-sim}}[\text{ENV outputs } 1]$ . This will not affect the simulation error.

The probability of ENV returning 1 in the real or simulated environments differs by  $\varepsilon/|Y|$ , and is centered around some constant  $p$ . That is, the probabilities are at least  $p + \delta$  and at most  $p - \delta$  respectively, where  $\delta = \varepsilon/(2|Y|)$ . Perform the following operations to ensure that the probability is centered around  $1/2$  as required by the lemma.

1. If  $p > 1/2$ , run normally with probability  $\frac{1}{2p}$ . Otherwise, return 0.
2. if  $p < 1/2$ , run normally with probability  $\frac{1}{2(1-p)}$ . Otherwise, return 1.

This will "normalize" the average of the probabilities in the real and ideal world to  $1/2$ . This might shrink  $\delta$  slightly – up to a factor of 2. The minimum value of  $\delta$  is  $\frac{\varepsilon}{4|Y|}$ , as required by the lemma.  $\square$

Define  $\varepsilon_0 = \sqrt{\varepsilon}$  and  $\varepsilon_i = (52nc)\varepsilon_{i-1} = (52nc)^i\varepsilon_0$  where  $c = c_{f,g}$  is the constant defined in [Lemma 20](#) and  $n$  is the maximum of  $|X|$  and  $|Y|$ .

**Lemma 23.** *If  $\text{trunc}(\pi, i)$  has simulation error at most  $\varepsilon_i$  then either  $f$  embeds in  $g$  or  $\text{trunc}(\pi, i+1)$  has simulation error at most  $\varepsilon_{i+1}$ .*

*Proof.* Consider any partial transcript  $t$  where the next action in  $\text{trunc}(\pi, i)$  is for the parties to make a call to  $g$ . Let protocol  $\pi_t$  be defined as follows: the parties “fast-forward” to  $t$  by imagining the transcript up to that round. They then complete the call to  $g$  and exit immediately afterwards.

If there is any  $\pi_t$  with simulation error less than  $1/c$  (call such a  $t$  *good*),  $f$  embeds in  $g$ : by [Lemma 20](#), there exists a single-round deterministic protocol for  $f$  in a  $g$ -hybrid world with simulation error less than 1. Then, by [Lemma 5](#),  $f$  embeds in  $g$ .

Assume that there is no good  $t$  in round  $r - i - 1$  of  $\text{trunc}(\pi, i)$  (that is, in  $\mathcal{R}$ ). We wish to bound the probability  $\Pr_{\text{trunc}(\pi, i)}[\mathcal{R}|xy]$  for all  $x, y$  in this case. Note that some  $t \in \mathcal{R}$  may have simulation error when run against one malicious party but not the other. Let  $\mathcal{R}_A$  be those  $t$  which have unacceptable simulation error against Alice, and let  $\mathcal{R}_B$  be similarly defined for Bob. Then:

$$\Pr_{\text{trunc}(\pi, i)}[\mathcal{R}|xy] \leq \Pr_{\text{trunc}(\pi, i)}[\mathcal{R}_A|xy] + \Pr_{\text{trunc}(\pi, i)}[\mathcal{R}_B|xy]$$

Consider, then, the probability  $\Pr_{\text{trunc}(\pi, i)}[\mathcal{R}_A|xy]$ . A symmetric argument will work for  $\mathcal{R}_B$ , and therefore we can use these to get a bound on the overall probability  $\Pr_{\text{trunc}(\pi, i)}[\mathcal{R}|xy]$ .

For each  $t$  in  $\mathcal{R}$ , consider  $\pi_t$ . The simulation error is at least  $1/c$ , so by [Lemma 22](#) we can construct an environment  $\text{ENV}_t$  satisfying:

1.  $\text{ENV}_t$  chooses inputs for Bob uniformly at random
2.  $\Pr_{\pi_t}[\text{ENV}_t \text{ outputs } 1] > \frac{1}{2} + \frac{1}{4|Y|c}$
3.  $\Pr_{\text{A-sim-for-}\pi_t}[\text{ENV}_t \text{ outputs } 1] < \frac{1}{2} - \frac{1}{4|Y|c}$

Fix a particular  $x$  and  $y$ . Consider the following attack:

- Alice runs on  $x$ , and the environment chooses input  $y^*$  for Bob uniformly at random.
- If the parties reach  $t \in \mathcal{R}$ , let  $\text{ENV}_t$  sample an input for Bob. If it samples  $y^*$ , run  $\text{ENV}_t$ , possibly allowing Alice to change her input maliciously. Otherwise, output 0.

The probability that during this attack we output 1 is the probability that we reached  $\mathcal{R}$  times the probability that  $\text{ENV}_t$  chooses input  $y^*$  times the probability that  $\text{ENV}_t$  succeeds at its attack given  $y^*$ . These probabilities are independent.

$$\sum_{t \in \mathcal{R}_A} \sum_{y^*} \Pr_{\text{trunc}(\pi, i)}[t|xy^*] \frac{1}{|Y|} \Pr_{\pi_t}[\text{ENV}_t \text{ returns } 1|y^*]$$

If we eliminate the summations, we get the following expression:

$$\left( \Pr_{\text{trunc}(\pi, i)}[\mathcal{R}_A|xy^*] \right) \left( \frac{1}{2} + \frac{1}{4|Y|c} \right)$$

By [Lemma 21](#) we can replace  $y^*$  with the input for Bob we desire to bound probability against,  $y$ , with a small change in probability of reaching a transcript.

The probability the environment outputs 1 in the real world is therefore at least:

$$\left( \Pr_{\text{trunc}(\pi, i)} [\mathcal{R}_A | xy] - 6\sqrt{\varepsilon} \right) \left( \frac{1}{2} + \frac{1}{4|Y|c} \right)$$

By a similar series of arguments, the probability the environment outputs 1 in the ideal world is at most:

$$\left( \Pr_{\text{A-sim-for-trunc}(\pi, i)} [\mathcal{R}_A | xy] + 6\sqrt{\varepsilon} \right) \left( \frac{1}{2} - \frac{1}{4|Y|c} \right)$$

Because the simulation error of  $\text{trunc}(\pi, i)$  is  $\varepsilon_i$ , the above expression is at most the following value:

$$\left( \Pr_{\text{trunc}(\pi, i)} [\mathcal{R}_A | xy] + 6\sqrt{\varepsilon} + \varepsilon_i \right) \left( \frac{1}{2} - \frac{1}{4|Y|c} \right)$$

We know that, because  $\text{trunc}(\pi, i)$  has a simulation error of  $\varepsilon_i$ , the difference between the output probabilities in the real and ideal worlds is at most  $\varepsilon_i$ , which means that, in particular:

$$\begin{aligned} \varepsilon_i &\geq \Pr_{\text{trunc}(\pi, i)} [\mathcal{R}_A | xy] \frac{1}{2|Y|c} - 12\sqrt{\varepsilon} - \varepsilon_i \left( \frac{1}{2} - \frac{1}{4|Y|c} \right) \\ &\Rightarrow \Pr_{\text{trunc}(\pi, i)} [\mathcal{R}_A | xy] \leq \left( |Y|c + \frac{1}{2} \right) \varepsilon_i + 24|Y|c\sqrt{\varepsilon} \\ &\Rightarrow \Pr_{\text{trunc}(\pi, i)} [\mathcal{R}_A | xy] \leq (26|Y|c) \varepsilon_i \end{aligned}$$

Recall that this is only against malicious Alice. We get the following bound without restricting which party is adversarial:

$$\Pr_{\text{trunc}(\pi, i)} [\mathcal{R} | xy] \leq (52|Y|c) \varepsilon_i = \varepsilon_{i+1}$$

Truncating directly before  $\mathcal{R}$ , then, will only increase the simulation error to  $\varepsilon_{i+1}$ .  $\square$

**Lemma** (Restatement of [Lemma 15](#)). *If, for incomplete and non-unilateral  $f$  and  $g$ ,  $f \sqsubseteq g$  using a protocol with security parameter  $\kappa$  and strict upper bound on number of rounds  $r = O(\log \kappa)$ , then  $f$  embeds in  $g$ .*

*Proof.*  $\text{trunc}(\pi, 0)$  has simulation error  $\varepsilon$  which is surely less than  $\varepsilon_0$ .

Either  $f$  embeds in  $g$  or we can apply the argument in [Lemma 23](#) up to  $r - 1$  times. If  $r = O(\log \kappa)$ , then we are left with a 1-round protocol  $\text{trunc}(\pi, r - 1)$  with simulation error  $\varepsilon_r = (52nc)^{O(\log \kappa)} \sqrt{\varepsilon} = \text{poly}(\kappa) \kappa^{-\omega(1)} = \kappa^{-\omega(1)}$  which is negligible. Then  $f \sqsubseteq_1 g$ , which by [Lemma 5](#) means that  $f$  actually does embed in  $g$ .  $\square$

## E Round Complexity & Parallel Calls to $g$

Our model encompasses protocols that make only a single call to  $g$  in each round. Requiring sequential calls to  $g$  is without loss of generality *with respect to security*, since in the UC model it cannot be *guaranteed* that calls happen in parallel. The adversary can without loss of generality schedule all the calls sequentially (learning the output of one before choosing an input to the next), resulting in a protocol in our model.



However, when considering round complexity, it is more realistic to allow protocols that make *parallel calls* to  $g$ . Although the adversary can schedule these parallel calls in sequence, we still consider the round complexity as that required by the *honest* parties. Most of our technical results apply to such protocols. More formally, consider protocols where at each step the parties may call  $n$  parallel instances of  $g$ , where  $n$  is agreed-upon by both parties. All of our results in [Section 5.3](#) and most of the results in [Section 6](#) apply to such protocols. In particular, we can collapse any  $f \sqsubseteq g$  protocol of  $O(\log \kappa)$  rounds to a single-round protocol that may make *many* parallel calls to  $g$  but uses no additional communication.

To extend our results, it suffices to show that such a protocol (many parallel calls to  $g$ , no additional communication) implies that  $f$  embeds in  $g$ . We have been currently unable to extend this step. The way we currently extract an embedding from a *single-call* protocol ([Lemma 5](#)) works by derandomizing the protocol, crucially using the fact that the number of possible actions in the protocol is *constant*. This property is not true when a protocol makes, say,  $O(\kappa)$  parallel calls to  $g$ .<sup>4</sup>

In this section we gave a  $\omega(\log \kappa)$ -round protocol for specific  $f$  using specific  $g$ . We point out that this particular protocol *cannot* be made constant-round by making all the  $g$ -calls in parallel. The simple attack is to split the  $g$ -calls into two groups and use different effective inputs in both (e.g., Alice uses inputs from the first row-group in half of the calls, and second row-group in the other half). With very good probability, this attack leaks as much as evaluating  $f$  on two inputs. In particular, Alice can learn Bob's input in its entirety from this attack.

We conjecture that there is no  $O(\log \kappa)$ -round protocol for this  $f$  using this  $g$ , even when the protocol allows unlimited parallel calls to  $g$  in each round.

---

<sup>4</sup>Our results hold as stated for protocols that call at most  $O(\log \kappa)$  instances of  $g$  in parallel at a time, where the number of possible actions is polynomial in  $\kappa$ .