# Characterizing the Cryptographic Properties of Reactive 2-Party Functionalities*

R. Amzi Jeffs† Mike Rosulek‡

January 3, 2013

**Abstract**

In secure multi-party computation, a reactive functionality is one which maintains persistent state, takes inputs, and gives outputs over many rounds of interaction with its parties. Reactive functionalities are fundamental and model many interesting and natural cryptographic tasks; yet their security properties are not nearly as well-understood as in the non-reactive case (known as secure function evaluation).

We present new combinatorial characterizations for 2-party reactive functionalities, which we model as finite automata. We characterize the functionalities that have passive-secure protocols, and those which are complete with respect to passive adversaries. Both characterizations are in the information-theoretic setting.

## 1 Introduction

Ever since Yao [Y82] introduced the concept of secure multi-party computation (SMPC) with his famous *Millionaire's Problem*, the majority of research in the area has focused on understanding **secure function evaluation (SFE)** tasks. In an SFE task, all parties provide inputs and then receive outputs according to a (typically) deterministic function, in a single round of interaction with the functionality. The functionality that carries out this task has no need for persistent memory — it simply receives inputs from the parties, computes outputs, and thereafter forgets everything.

Yet, SMPC security models (*e.g.*, [C01]) allow for functionalities that maintain internal state across many rounds of interaction. We call such functionalities **reactive**. The most well-known example of an inherently reactive functionality is bit-commitment, the cryptographic equivalent of a locked box.

In a secure protocol, the parties must achieve the same effect as the functionality. Reactivity introduces new and unique challenges; in particular, there is a tension between the fact that the parties may *individually* have a great deal of uncertainty about the functionality's internal state, and the fact that the parties *collectively* must be able to maintain its internal state in order to correctly simulate its behavior.

To understand reactive functionalities is, therefore, to understand how persistent information can be maintained, updated, kept secret, and computed upon. What's more, from a practical

*This work appears in the proceedings of *TCC: Theory of Cryptography Conference* 2013.

†Department of Computer Science, Harvey Mudd College. rjeffs@g.hmc.edu.

‡Department of Computer Science, University of Montana. mikero@cs.umt.edu. Supported by NSF grant CCF-1149647.

perspective, reactive tasks are fundamental — any task involving time-sensitive release of information or the ability for parties to adapt to new information learned from an interaction must be necessarily reactive.

**Background & Related Work.** The first security model for which SFE tasks were understood is the model of passive security against computationally unbounded adversaries. Beaver [B89] & Kushilevitz [K89] independently characterized secure realizability for 2-party SFE tasks in this model. These results characterized which functionalities have *perfectly* secure protocols; the same characterization was later extended to the case where negligible security error is allowed [MPR09, KMR09]. We strongly leverage this characterization in our own result for the reactive case.

A functionality $\mathcal{F}$ is said to be *complete* (with respect to some security notion for protocols) if every functionality has a secure protocol in which the parties are allowed to make use of ideal instances of $\mathcal{F}$. Kilian [K91] was the first to characterize completeness for 2-party SFE functionalities. The result was later generalized to functionalities with possibly different outputs to the two parties [KMQ11]. As before, we strongly leverage the well-known characterization for the SFE case in our own result for the reactive case.

These characterizations, and many others for SFE tasks (*e.g.*, [CK91, K00, KKMO00]) are exclusively *combinatorial* in nature. Each SFE is associated with its 2-dimensional input/output table and then classified based on whether this table has a certain structure — say, a forbidden kind of $2 \times 2$ submatrix.

In some security settings, there exist secure protocols for every SFE functionality (*e.g.*, standalone security in the computationally bounded setting); it is not hard to see that this also implies secure protocols for all reactive functionalities as well. However, *hardness* (infeasibility) results for reactive functionalities are much rarer in the literature. Some fundamental reactive functionalities like bit commitment have been studied in an *ad hoc* fashion [CF01]. To the best of our knowledge, large *classes* of reactive functionalities have been considered only in [PR08, MPR10, R12]. Of these, only one result of Maji, Prabhakaran, and Rosulek [MPR10] involves a *combinatorial* (decidable) characterization. They characterize the 2-party reactive functionalities which have UC-secure protocols without any setup (the characterization is the same for both the computationally bounded and unbounded settings). They model functionalities as deterministic, finite-state transducers; our work uses the same automata model of reactive functionalities. We note that, while the SMPC paradigm allows one to consider reactive functionalities that cannot be represented as such finite automata, many important and natural functionalities can indeed be modeled in this way (*e.g.*, bit commitment).

## 1.1 Our Results

We derive combinatorial characterizations for the cryptographic properties of 2-party reactive functionalities. In particular, we characterize triviality (*i.e.*, feasibility) and completeness with respect to computationally unbounded, passive (a.k.a. semi-honest, or honest-but-curious) adversaries. Ours is the first work to classify properties of reactive functionalities in this fundamental setting. Following [MPR10], we model reactive functionalities as finite automata.

For a reactive functionality $\mathcal{F}$, define a related *non-reactive* functionality $\mathcal{F}^{(k)}$ which takes a length-$k$ sequence of inputs from each of Alice and Bob, then runs $\mathcal{F}$ for $k$ rounds on these inputs and gives each party their corresponding length-$k$ sequence of outputs. It is not difficult to see that:

- $\mathcal{F}$ is passive-trivial if and only if for all $k \in \mathbb{N}$, $\mathcal{F}^{(k)}$ is passive-trivial.

- $\mathcal{F}$ is passive-complete if and only if $\mathcal{F}^{(k)}$ is passive-complete for some $k \in \mathbb{N}$.

In this way it is possible to reduce the characterizations for reactive functionalities to the corresponding well-known ones for SFE functionalities.

However, the above characterizations are of limited use. Both conditions are infinitary in nature (requiring either the universe of all protocols to be enumerated, or an infinite number of values $k$ to be checked). Our technical contribution is in our analyses showing that only a finite number of values $k$ need to be checked. We obtain characterizations of the following form:

**Main Theorem.** *Let $\mathcal{F}$ be a reactive 2-party functionality. There exist constants $K_t$ and $K_c$, which depend only on the number of states in $\mathcal{F}$, such that:*

1. *$\mathcal{F}$ is passive-trivial if and only if for all $k \leq K_t$, $\mathcal{F}^{(k)}$ is passive-trivial; and*
2. *$\mathcal{F}$ is passive-complete if and only if $\mathcal{F}^{(k)}$ is passive-complete for some $k \leq K_c$.*

Thus we obtain total decision procedures for determining triviality and completeness of reactive functionalities. The characterizations for SFE are combinatorial in nature, and thus ours also inherit that flavor. Also, the statement of the main theorem is valid even if protocols are allowed a negligible error (though the final characterization for passive-triviality is the same whether zero error or negligible error is required).

The bulk of our effort is devoted to proving the existence of the constant $K_t$ above. The main technical challenge when dealing with reactive functionalities is accounting for the uncertainty both parties have about the (hidden) internal state of the functionality. For example, even if the behavior of the functionality is benign in every state, it may still be possible to elicit non-trivial behavior from the functionality when both parties have uncertainty about its internal state. To justify our somewhat complicated analysis, we show that simply inspecting the local behavior of each state does not suffice to characterize the security properties of reactive functionalities.

To properly deal with the complications of a functionality's hidden internal state, we develop a "normal form" for functionalities that explicitly captures the common knowledge both parties have about the internal state. The final characterization follows then by the requirements imposed by this normal form.

Our characterizations are for functionalities that give possibly different outputs to each party. Using the normal form described above, we show that, unless a functionality is passive-complete, it is isomorphic to one with symmetric output. This generalizes an analogous result of [KMQ11] for non-reactive functionalities.

## 2 Preliminaries

A probability $p(n)$ is negligible if for all $c > 0$, $p(n) < n^{-c}$ for all but finitely many $n$. We use bold symbols (*e.g.*, $\boldsymbol{x}$, $\boldsymbol{y}$) to denote sequences over some finite alphabet (*e.g.*, $X$ or $Y$). We write $|\boldsymbol{x}|$ to denote the length of a sequence, and we use $\|$ to denote concatenation of sequences (*e.g.*, $\boldsymbol{x}\|x$). When $T$ is a 2-dimensional table, and $a$ and $b$ are appropriate indices, we use the notation $T[a, b]$ to denote the entry of $T$ in row $a$, column $b$.

### 2.1 Passive Security

We use the standard real-ideal paradigm [GMW87] to define protocol security. We exclusively consider security against passive (a.k.a. honest-but-curious, or semi-honest) and computationally unbounded adversaries, and we call protocols which achieve this standard **passive-secure** for short.

We say that a protocol **uses** the functionality $\mathcal{G}$ if the parties are instructed to interact with ideal instances of the functionality $\mathcal{G}$ (*i.e.*, the protocol is in the "$\mathcal{G}$-hybrid model").

We say that a functionality $\mathcal{F}$ is **passive-trivial** if there is a passive-secure protocol for $\mathcal{F}$ without any setups. We say that $\mathcal{F}$ is **passive-complete** if there is a passive-secure oblivious-transfer protocol that uses access to ideal instances of $\mathcal{F}$. In this work, we consider the information-theoretic setting exclusively, so adversaries are computationally unbounded. Unlike the first characterizations for SFE functionalities [B89, K89], we do not restrict our attention to protocols that achieve perfect security. Instead, we use the now-standard notion of passive security, which permits protocols to have a negligible simulation error.

**Isomorphism.** We call a protocol for $\mathcal{F}$ using $\mathcal{G}$ a *local* protocol if it uses just one instance of $\mathcal{G}$ to realize an instance of $\mathcal{F}$, does not use communication between the parties other than $\mathcal{G}$, and each round of outputs for $\mathcal{F}$ is realized in the protocol by the parties making a single call to $\mathcal{G}$. Then call two functionalities $\mathcal{F}$ and $\mathcal{G}$ **isomorphic** if there is a local, passive-secure protocol for $\mathcal{F}$ using $\mathcal{G}$ and vice-versa.

## 2.2 Notation and Characterizations for SFE

We briefly review known characterizations for passive-triviality and passive-completeness of SFE functionalities. We state the characterizations in terms of new notation, which cleanly unifies the cases of symmetric and non-symmetric output for the two parties. The terminology defined here is used throughout the work.

A **2-party SFE** $\mathcal{F}$ is specified by finite sets $X$ and $Y$, and two deterministic functions $f_A : X \times Y \to \{0,1\}^*$ and $f_B : X \times Y \to \{0,1\}^*$. We use these default variable names throughout this work. As a cryptographic functionality, Alice and Bob provide inputs $x \in X$ and $y \in Y$ to $\mathcal{F}$, respectively, and receive outputs $f_A(x,y)$ and $f_B(x,y)$, respectively.[1]

Let $\mathsf{restrict}(\mathcal{F}, A \times B)$ denote the restriction of $\mathcal{F}$ to the input domain $A \times B \subseteq X \times Y$. For $(x,y) \in X \times Y$, we define $\mathsf{rectangle}(\mathcal{F}, x, y) = A_{x,y} \times B_{x,y}$ where $A_{x,y} = \{x' \mid f_B(x',y) = f_B(x,y)\}$ and $B_{x,y} = \{y' \mid f_A(x,y') = f_A(x,y)\}$. We say that $\mathcal{F}$ is **basic** on $\tilde{X} \times \tilde{Y}$ if: for all $y \in \tilde{Y}$, $f_B$ is a constant function on $\tilde{X} \times \{y\}$, and for all $x \in \tilde{X}$, $f_A$ is a constant function on $\{x\} \times \tilde{Y}$. Basic functions require no interaction to evaluate (a party's input has no influence on the other's output). Finally, an OR**-minor** in $\mathcal{F}$ is a tuple $(x, x', y, y') \in X^2 \times Y^2$ with:

$$f_A(x,y) = f_A(x,y'); \qquad f_B(x,y) = f_B(x',y);$$
$$\Big(f_A(x',y), f_B(x,y')\Big) \neq \Big(f_A(x',y'), f_B(x',y')\Big).$$

**Passive-Completeness.** OR-minors exactly characterize passive-completeness for SFE functionalities:

**Lemma 2.1** ([KMQ11])**.** *The following are equivalent for a 2-party SFE $\mathcal{F}$:*

1. *$\mathcal{F}$ is passive-complete.*
2. *$\mathcal{F}$ has an* OR*-minor.*
3. *There exist inputs $x, y$ such that $\mathcal{F}$ is* **not** *basic on* $\mathsf{rectangle}(\mathcal{F}, x, y)$.

[1] In this work we consider security only against passive adversaries. As such, issues of fairness in output delivery are not relevant.

*Proof.* The equivalence of 1 & 2 was shown by Kraschewski & Müller-Quade [KMQ11], generalizing the analogous statement for symmetric-output functions by Kilian [K91]. The equivalence of 2 & 3 follows straightforwardly from the definitions of OR-minor and $\mathsf{rectangle}(\mathcal{F}, x, y)$. □

The following useful lemma was also proven in [KMQ11]:

**Lemma 2.2** (Symmetrization [KMQ11])**.** *Given an SFE $\mathcal{F}$, define the (symmetric) SFE functionality $\mathcal{F}_{sym}$, which on input $x$ from Alice and $y$ from Bob gives both parties output $\mathsf{rectangle}(\mathcal{F}, x, y)$.*

*If $\mathcal{F}$ has no OR-minor, then $\mathcal{F}$ is isomorphic to $\mathcal{F}_{sym}$.*

**Passive-Triviality.** Passive-triviality for SFE functionalities is characterized by a combinatorial condition called *decomposability.*

**Definition 2.3** ([B89, K89])**.** *An SFE $\mathcal{F}$ is* **decomposable** *if one of the following holds:*

1. *$\mathcal{F}$ is basic (defined above); or,*
2. *There is a partition $X = \tilde{X}_1 \cup \tilde{X}_2$ so that for all $x_1 \in \tilde{X}_1, x_2 \in \tilde{X}_2$ and $y \in Y$, $f_B(x_1, y) \neq f_B(x_2, y)$, and furthermore $\mathsf{restrict}(\mathcal{F}, \tilde{X}_1 \times Y)$ and $\mathsf{restrict}(\mathcal{F}, \tilde{X}_2 \times Y)$ are decomposable; or,*
3. *There is a partition $Y = \tilde{Y}_1 \cup \tilde{Y}_2$ so that for all $x \in X$, $y_1 \in \tilde{Y}_1$, and $y_2 \in \tilde{Y}_2$, $f_A(x, y_1) \neq f_A(x, y_2)$, and furthermore $\mathsf{restrict}(\mathcal{F}, X \times \tilde{Y}_1)$ and $\mathsf{restrict}(\mathcal{F}, X \times \tilde{Y}_2)$ are decomposable.*

**Lemma 2.4** ([MPR09, KMR09])**.** *$\mathcal{F}$ is passive-trivial if and only if it is decomposable.*

This lemma was originally proved for the case of perfectly secure protocols by Beaver [B89] & Kushilevitz [K89] (independently); later it was extended for the standard notion of security (allowing negligible error) by Maji, Prabhakaran & Rosulek [MPR09] and Künzler, Müller-Quade & Raub [KMR09] (independently).

## 2.3 Model of Reactive Functionalities

We use the model of reactive functionalities from [MPR10]:

**Definition 2.5** ([MPR10])**.** *A* **(2-party) deterministic finite functionality (DFF)** *is a tuple $\mathcal{F} = (Q, X, Y, \delta, f_A, f_B, q_0)$, where*

- *$Q$ is a finite set of states,*
- *$X$ and $Y$ are finite input sets,*
- *$\delta : Q \times X \times Y \to Q$ is the state transition function,*
- *$f_A, f_B : Q \times X \times Y \to \{0,1\}^*$ are two output functions, and*
- *$q_0 \in Q$ is the start state.*

*The behavior of $\mathcal{F}$ as an ideal functionality is defined formally in Figure 1.*[2] *As before, we use these standard variable names throughout.*

We extend the functions $\delta$, $f_A$, and $f_B$ to *sequences* of inputs in the natural way. Let $\boldsymbol{x} = (x_1, \ldots, x_k) \in X^k$ and $\boldsymbol{y} = (y_1, \ldots, y_k) \in Y^k$. We write $\delta(q, \boldsymbol{x}, \boldsymbol{y})$ to denote the state of $\mathcal{F}$ after receiving inputs $(x_1, y_1), \ldots, (x_k, y_k)$ starting in state $q$. We write $f_A(q, \boldsymbol{x}, \boldsymbol{y})$ to denote the *concatenation* of Alice's $k$ outputs when $\mathcal{F}$ receives inputs $(x_1, y_1), \ldots, (x_k, y_k)$ starting in state $q$. We write $\mathcal{F}^{(k)}$ to denote the SFE functionality which on input $(\boldsymbol{x}, \boldsymbol{y})$ with $|\boldsymbol{x}| = |\boldsymbol{y}| = k$, gives output $f_A(q_0, \boldsymbol{x}, \boldsymbol{y})$ to Alice and $f_B(q_0, \boldsymbol{x}, \boldsymbol{y})$ to Bob. Then we have the following simple observations:

[2] As before, issues of fairness in output delivery are not relevant when considering only passive adversaries.

Set variable $q := q_0$. Then repeatedly do:

- Wait for input $x \in X$ from Alice and input $y \in Y$ from Bob. Give outputs $f_A(q, x, y)$ to Alice and $f_B(q, x, y)$ to Bob. Update $q := \delta(q, x, y)$ and repeat.

Figure 1: Semantics of the DFF functionality $\mathcal{F} = (Q, X, Y, \delta, f_A, f_B, q_0)$

**Proposition 2.6.** *Let $\mathcal{F}$ be a DFF, and $\mathcal{F}^{(k)}$ defined above.*

1. *For all $k$, there is a passive-secure protocol for $\mathcal{F}^{(k)}$ using $\mathcal{F}$.*
2. *There is a passive-secure protocol for $\mathcal{F}$ using $\{\mathcal{F}^{(k)}\}_{k\in\mathbb{N}}$.*

*Hence:*

3. *$\mathcal{F}$ is passive-trivial if and only if, for all $k$, $\mathcal{F}^{(k)}$ is passive-trivial.*
4. *$\mathcal{F}$ is passive-complete if and only if $\mathcal{F}^{(k)}$ is passive-complete for some $k$.*

The secure protocol for $\mathcal{F}$ using (the infinite set of functionalities) $\{\mathcal{F}^{(k)}\}$ requires both parties to maintain their history of inputs $\boldsymbol{x}$ and $\boldsymbol{y}$. In the $(k+1)$th round with histories $\boldsymbol{x}$ and $\boldsymbol{y}$ and new inputs $x$ and $y$, both parties call $\mathcal{F}^{(k+1)}$ with inputs $\boldsymbol{x}\|x$ and $\boldsymbol{y}\|y$.[3]

# 3 Limits of Local Conditions

When classifying a DFF for its cryptographic properties, one is tempted to examine the behaviors of each state, in isolation, for certain properties. We call such a test *local*, and in this section we describe the limitations of such local tests.

One way that local tests fail stems from the fact that local information is not enough to determine even whether two states have identical behavior. Given that, suppose some state has a transition function that contains an OR-minor involving states $q$, $q'$. How this OR-minor affects the triviality/completeness of the functionality depends crucially on whether $q$ and $q'$ have identical behavior. Still, we will show that, even when redundant states have been removed, local tests are insufficient to classify the cryptographic properties of DFFs.

We say that two states $q$ and $q'$ are **redundant** in $\mathcal{F}$ if for all $\boldsymbol{x}, \boldsymbol{y}$ with $|\boldsymbol{x}| = |\boldsymbol{y}|$ we have $f_A(q, \boldsymbol{x}, \boldsymbol{y}) = f_A(q', \boldsymbol{x}, \boldsymbol{y})$ and $f_B(q, \boldsymbol{x}, \boldsymbol{y}) = f_B(q', \boldsymbol{x}, \boldsymbol{y})$. Redundant states can easily be collapsed in $\mathcal{F}$ using the classical Myhill-Nerode DFA minimization algorithm. Throughout this work we will generally assume without loss of generality that redundant states have been collapsed. Non-redundant state pairs $(q, q')$ have a *distinguishing sequence* $(\boldsymbol{x}, \boldsymbol{y})$ satisfying

$$\big(f_A(q, \boldsymbol{x}, \boldsymbol{y}), f_B(q, \boldsymbol{x}, \boldsymbol{y})\big) \neq \big(f_A(q', \boldsymbol{x}, \boldsymbol{y}), f_B(q', \boldsymbol{x}, \boldsymbol{y})\big).$$

Local tests can give an indication of the complexity of some DFFs, but cannot give a complete characterization. We consider local tests which inspect the output and transition functions of each state. To formalize this, we define for a DFF $\mathcal{F}$ a related DFF $\mathcal{F}_{st}$ to be a modification to $\mathcal{F}$ which always announces its internal state to both parties. Then the output function of state $q$ in $\mathcal{F}_{st}$ contains all the relevant information about both the output and transition functions of $q$ in $\mathcal{F}$.

**Lemma 3.1.** *Let $\mathcal{F}$ be a DFF that contains no redundant states.*

[3] Note that we use the fact that the parties honestly follow the protocol, as they must faithfully keep track of their history of inputs to simulate $\mathcal{F}$ using $\{\mathcal{F}^{(k)}\}$.

1. *If any reachable state in $\mathcal{F}_{st}$ has an output function that is not decomposable, then $\mathcal{F}$ is not passive-trivial.*
2. *If any reachable state in $\mathcal{F}_{st}$ has an output function that contains an* OR*-minor, then $\mathcal{F}$ is passive-complete.*
3. *The converses of the above statements are false. In fact, there exist functionalities of arbitrary status (i.e., passive-trivial, passive-complete, neither) without redundant states whose output functions are constant and whose transition functions are decomposable in every state.*

*Proof.* For items (1) and (2), we can assume without loss of generality that it is the *start state* of $\mathcal{F}$ that has the offending transition/output functions. More formally, let $\mathcal{F}[q]$ denote $\mathcal{F}$ with its start state changed to $q$. If $q$ is reachable in $\mathcal{F}$— say, via sequence $(\boldsymbol{x}, \boldsymbol{y})$ — then a passive-secure protocol for $\mathcal{F}[q]$ using $\mathcal{F}$ is to have both parties send an initial "preamble" of $(\boldsymbol{x}, \boldsymbol{y})$ to $\mathcal{F}$ and then proceed with the dummy protocol.

For items (1) and (2), if it is an output function in $\mathcal{F}$ (*i.e.*, not in $\mathcal{F}_{st}$) that is non-decomposable (resp. contains an OR-minor), then the claim follows much more easily. There is a natural passive-secure protocol using $\mathcal{F}$ that realizes the start state's (SFE) output function – the parties simply interact with $\mathcal{F}$ for one round. The claims then follow from the complete characterizations for passive-triviality and passive-completeness of SFE (see Section 2.2).

(1) We fall into the case described above unless the start state's output function is decomposable. So, let $\mathcal{G} = (g_A, g_B)$ denote the SFE which evaluates the first round only of $\mathcal{F}_{st}$ and Let $\tilde{X} \times \tilde{Y}$ denote minimal subsets such that $\mathsf{restrict}(\mathcal{G}, \tilde{X} \times \tilde{Y})$ is not decomposable.

Next we show that the output function of the start state in $\mathcal{F}$ (not $\mathcal{F}_{st}$) is *basic* on $\tilde{X} \times \tilde{Y}$. If not, then since it is decomposable, it induces either a corresponding row- or column-decomposition step in $\mathcal{G}$ (which includes the $\mathcal{F}$-output as well as the state). This splits $\tilde{X} \times \tilde{Y}$ into at least two smaller subdomains, which by the minimality condition are decomposable. Thus $\mathcal{G}$ is decomposable on $\tilde{X} \times \tilde{Y}$, which we have assumed to be false. By this contradiction, we see that the output function of $\mathcal{F}$'s start state must be *basic* on $\tilde{X} \times \tilde{Y}$.

Let $q, q'$ be two distinct states reachable from the start state by single transitions on $(x, y) \in \tilde{X} \times \tilde{Y}$. As these states are non-redundant, let $(\boldsymbol{x}, \boldsymbol{y})$ be a distinguishing sequence for them, with $|\boldsymbol{x}| = |\boldsymbol{y}| = k$. Now consider the SFE functionality $\mathcal{H} = (h_A, h_B)$ which on input $(x, y) \in \tilde{X} \times \tilde{Y}$ gives output $f_A(q_0, x\|\boldsymbol{x}, y\|\boldsymbol{y})$ to Alice and $f_B(q_0, x\|\boldsymbol{x}, y\|\boldsymbol{y})$ to Bob. There is a passive-secure protocol for $\mathcal{H}$ using $\mathcal{F}$ ($\mathcal{H}$ is a submatrix of $\mathcal{F}^{(k+1)}$). By Lemma 2.4 it suffices to show that $\mathcal{H}$ is not decomposable.

We have that $\mathsf{rectangle}(\mathcal{G}, x, y) \subseteq \mathsf{rectangle}(\mathcal{H}, x, y)$, since the first round of $\mathcal{F}$ gives *basic* output for inputs in $\tilde{X} \times \tilde{Y}$. Also, by our choice of $\boldsymbol{x}, \boldsymbol{y}$ as a distinguishing sequence we have that $\mathcal{H}$ itself is not basic. Consider any partition of $\tilde{X}$, say, $\tilde{X} = \tilde{X}_0 \cup \tilde{X}_1$. Since $\mathcal{G}$ is minimal and not decomposable, there exists $x_0 \in \tilde{X}_0$, $x_1 \in \tilde{X}_1$, $y \in \tilde{Y}$ such that $g_B(x_0, y) = g_B(x_1, y)$. Hence, $h_B(x_0, y) = h_B(x_1, y)$ so $\tilde{X} = \tilde{X}_0 \cup \tilde{X}_1$ does not satisfy the requirement for decomposability of $\mathcal{H}$. Symmetrically, no partition of $\tilde{Y}$ satisfies the requirement; hence $\mathcal{H}$ is not decomposable, as desired.

(2) Let $(x_0, x_1, y_0, y_1)$ be the inputs of the relevant OR-minor in the start state of $\mathcal{F}_{st}$; as above, we may assume that the output function of $q_0$ in $\mathcal{F}$ is basic over $\{x_0, x_1\} \times \{y_0, y_1\}$. Hence, the OR-minor occurs entirely in the transition function of $\mathcal{F}$; *i.e.*, $\delta(q_0, x_i, y_j) = r_{i \vee j}$ for some states $r_0 \neq r_1$. Let $(\boldsymbol{x}, \boldsymbol{y})$ be a distinguishing sequence for $r_0, r_1$, with $|\boldsymbol{x}| = |\boldsymbol{y}| = k$. Then it is straightforward to verify that $(x_0\|\boldsymbol{x}, x_1\|\boldsymbol{x}, y_0\|\boldsymbol{y}, y_1\|\boldsymbol{y})$ is an OR-minor in $\mathcal{F}^{(k+1)}$. Note that we crucially use the fact that the output of $\mathcal{F}$ is basic in the first round for the chosen input sequences.

(3) Let $\mathcal{G}$ be an arbitrary *symmetric SFE* functionality to be chosen later, and define $\mathcal{F}$ to do the following: In the first round, $\mathcal{F}$ gives constant output (regardless of the input) and remembers

Alice's input $x$ in its states, ignoring Bob's input. In the second round, $\mathcal{F}$ gives constant output and transitions to state $r_{\mathcal{F}(x,y)}$, where $y$ is the input of Bob in the second round (Alice's input in this round is ignored). Here, states $\{r_i\}_i$ are a set of states distinct from those used to implement rounds 1 & 2. Finally, in state $r_i$, $\mathcal{F}$ gives constant output $i$ and self-loops.

Note that the transition functions of $\mathcal{F}$ are all decomposable (in particular, at each round the transitions depend on at most one party's input), as are the output functions (they are constant functions in each state). A passive-secure protocol for $\mathcal{G}$ can be obtained from $\mathcal{F}$, and vice-versa, in the natural way. Thus, $\mathcal{F}$ and $\mathcal{G}$ have the same status (*e.g.*, trivial, complete, neither). We complete the proof by taking $\mathcal{G}$ to be an appropriate passive-trivial, passive-complete, or intermediate SFE. □

# 4 Characterizing Completeness

**Theorem 4.1.** *Let $\mathcal{F}$ be a DFF with $n$ states. Then $\mathcal{F}$ is passive-complete if and only if there exists $k \leq n^4$ such that $\mathcal{F}^{(k)}$ contains an* OR*-minor.*

*Proof.* The "⇐" direction follows trivially from Proposition 2.6 and the characterization of completeness for SFE functionalities based on OR-minors [K91, KMQ11].

For the other direction, let $\pi$ be a passive-secure protocol for 1-out-of-2 oblivious transfer (OT) using $\mathcal{F}$. For sake of contradiction, suppose that for every $k$, $\mathcal{F}^{(k)}$ has no OR-minor. Following Proposition 2.6, we can without loss of generality modify $\pi$ to obtain a passive-secure OT protocol using the collection of SFE functions$\{\mathcal{F}^{(k)}\}_k$.

Consider an execution of the protocol in which input bits $a_0, a_1$ for Alice and $b$ for Bob are chosen uniformly (*i.e.*, Bob should learn $a_b$ and Alice should learn nothing). Let $V$ denote the messages exchanged in the protocol along with the list of $\mathsf{rectangle}(\mathcal{F}^{(k)}, x, y)$ values for every time $\mathcal{F}^{(k)}$ is invoked with inputs $(x, y)$ in the protocol. Define $P_{s,t} = \Pr[a_s = t \mid V]$ for $s, t \in \{0, 1\}$. Importantly, since no $\mathcal{F}^{(k)}$ contains an OR-minor, both parties can compute $V$ (Lemma 2.2), and hence the $P_{s,t}$ values.

Suppose Bob guesses Alice's input $a_s$ to be the value $t$ that maximizes $P_{s,t}$. By a straightforward argument, this guess will be correct with probability $P_{s,t}$. Hence, by the security of the protocol, $P_{1-b,0}$ and $P_{1-b,1}$ must be close to $1/2$ with high probability (recall that $b$ is Bob's choice bit). However, by the correctness of the protocol we must also have $P_{b,a_b}$ close to one and $P_{b,1-a_b}$ close to zero with high probability as well.[4] Since Alice can also compute these $P_{s,t}$ values, this gives her a way to determine Bob's choice bit $b$ with high probability (*i.e.*, guess the value $b$ such that $P_{b,a_b}$ is maximized). Hence, we contradict the passive-security of the protocol, as desired. We note that this part of the proof is essentially the same as Kilian's proof for the case of SFE functionalities [K91].

So far we have shown only that some $\mathcal{F}^{(k)}$ must have an OR-minor. Fix $k$ to be minimal value such that $\mathcal{F}^{(k)}$ contains an OR-minor. If $k \leq n^4$ then we are done. Otherwise, let $d(\boldsymbol{x}, \boldsymbol{y}, i)$ denote the internal state of $\mathcal{F}$ after the first $i$ rounds when the input sequence is $(\boldsymbol{x}, \boldsymbol{y})$, when $i \leq k = |\boldsymbol{x}| = |\boldsymbol{y}|$. Define

$$D(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}', \boldsymbol{y}', i) := \Big(d(\boldsymbol{x}, \boldsymbol{y}, i), d(\boldsymbol{x}, \boldsymbol{y}', i), d(\boldsymbol{x}', \boldsymbol{y}, i), d(\boldsymbol{x}', \boldsymbol{y}', i)\Big) \in Q^4.$$

[4] The correctness of the protocol implies that Bob's *entire view* determines $a_b$ with high probability, whereas $P_{s,t}$ is computed using less information than Bob's view. In particular, $V$ does not include Bob's inputs to the ideal functionality. However, every input for Bob from $\mathsf{rectangle}(\mathcal{F}^{(k)}, x, y)$ would have had exactly the same effect on the interaction, in the absence of an OR-minor. In other words, an honest Bob only needs to remember his input with as much specificity as $\mathsf{rectangle}(\mathcal{F}^{(k)}, x, y)$.

Let $(\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{y}, \boldsymbol{y}')$ be the OR-minor of $\mathcal{F}^{(k)}$. By the pigeonhole principle (since $k > n^4$) there are distinct indices $i, j \in \{0, \ldots, k\}$ such that $D(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}', \boldsymbol{y}', i) = D(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}', \boldsymbol{y}', j)$. Then removing positions $i$ through $j-1$ in the input sequences $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}', \boldsymbol{y}'$ yields an OR-minor in $\mathcal{F}^{(k-j+i)}$. But this contradicts the minimality of $k$, so we must have originally had $k \leq n^4$. □

# 5 Characterizing Passive Triviality

## 5.1 Overview

Our approach is to reduce our characterization of DFFs as much as possible to the known characterizations for SFE (given in Section 2.2). For intuition, suppose Alice & Bob have performed $k$ rounds with $\mathcal{F}$, giving input sequences $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. When the functionality is passive-trivial, both parties can agree on $A \times B = \mathsf{rectangle}(\mathcal{F}^{(k)}, \boldsymbol{x}, \boldsymbol{y})$, knowing that $(\boldsymbol{x}, \boldsymbol{y}) \in A \times B$. Their uncertainty about the current *state* of $\mathcal{F}$ is then captured by $\mathsf{restrict}(\delta^{(k)}, A \times B)$, where $\delta^{(k)}$ is the extended transition function $\delta^{(k)}(\boldsymbol{x}, \boldsymbol{y}) = \delta(q_0, \boldsymbol{x}, \boldsymbol{y})$.

Intuitively, both parties can maintain the 2-dimensional table $C = \mathsf{restrict}(\delta^{(k)}, \mathsf{rectangle}(\mathcal{F}^{(k)}, \boldsymbol{x}, \boldsymbol{y}))$, along with their respective inputs $\boldsymbol{x}$ and $\boldsymbol{y}$ to this table. Furthermore, these three pieces of information ($\boldsymbol{x}$, $\boldsymbol{y}$, and $C$) are enough to determine all future behavior of $\mathcal{F}$. One could imagine a "canonical" protocol for $\mathcal{F}$ in which parties maintain such information (Alice maintaining $\boldsymbol{x}$ and $C$; Bob maintaining $\boldsymbol{y}$ and $C$).

With this as our starting point, we argue the following. First, duplicate rows & columns within $C$ can be canonically removed. Second, the table $C$ is a submatrix of $\delta^{(k)}$; as such, it can contain no OR-minor when $\mathcal{F}$ is passive-trivial. Finally, we prove a purely combinatorial lemma stating that any table that avoids duplicate rows, duplicate columns, and OR-minors must be bounded in its dimensions (the bound is a function of the number of allowed values in the cells of the table). Hence, when $\mathcal{F}$ is trivial, the table $C$ described above has an *a priori*, finite bound in size.

Our combinatorial lemma reveals some structure of functions which avoid OR-minors. In that sense, our lemma is reminiscent of similar lemmas used in [CK91] and [K11], for the $n$-party setting.

We prove our characterization by converting $\mathcal{F}$ into an equivalent "normal form" $\widehat{\mathcal{F}}$, which simulates $\mathcal{F}$ by keeping track of the information ($\boldsymbol{x}$, $\boldsymbol{y}$, and $C$) described above. In each state of $\widehat{\mathcal{F}}$, we use the internal state variable $C$ to associate a related submatrix of $\mathcal{F}^{(k)}$ for some $k$. We then show that $\mathcal{F}$ is passive-trivial if and only if each of these submatrices of $\{\mathcal{F}^{(k)}\}_k$ is itself passive-trivial. Importantly, there can be only a finite number of states — hence, a finite number of $\mathcal{F}^{(k)}$ submatrices — to inspect when deciding whether $\mathcal{F}$ is passive-trivial.

We highlight one important subtlety in the construction of $\widehat{\mathcal{F}}$. Our combinatorial lemma shows that the table $C$ has bounded size, but we are also associating its rows & columns with $\mathcal{F}$-input sequences of length $k$. Thus, while the table itself has bounded size, conceivably the row- and column-"labels" become unbounded in length. Our construction of $\widehat{\mathcal{F}}$ implicitly shows that these row- and column-labels are not used meaningfully; that is, they can be renormalized to simply be numerical indices into the table. Hence, the entire state-space of $\widehat{\mathcal{F}}$ (which contains this table $C$ as well as two labels indexing into the table) is indeed finite.

## 5.2 Combinatorial Lemma

**Definition 5.1** (Grid colorings and their properties)**.** *A* $k$**-coloring of an** $m \times n$ **grid** *is a function* $C : \{1, \ldots, m\} \times \{1, \ldots, n\} \to \{1, \ldots, k\}$*. A row* $i$ *is a* **duplicate row** *if* $C(i, \cdot) \equiv C(i', \cdot)$ *for some* $i' \neq i$*. Duplicate columns are defined analogously. A tuple* $(i, i', j, j')$ *forms an* OR**-minor** *in* $C$ *if* $C(i, j) = C(i, j') = C(i', j) \neq C(i', j')$*.*

We will use the following lemma, which states that sufficiently large grid colorings cannot avoid duplicate rows, duplicate columns, and OR-minors.

**Lemma 5.2** (Unavoidable structures of grid colorings)**.** *There is a function $R : \mathbb{N} \to \mathbb{N}$ satisfying the following property. For every $k$-coloring $C$ of an $m \times n$ grid, if $\max\{m, n\} \geq R(k)$, then $C$ contains either a duplicate row, duplicate column, or an* OR*-minor.*

*Proof.* We prove the lemma for the bound $R(2) = 3$; $R(k) = k \cdot R(k-1)$. Thus $R(k) = \Theta(k!)$. In fact, we prove the two stronger statements that (1) if $m \geq R(k)$ then $C$ contains either a duplicate row or an OR-minor; (2) if $n \geq R(k)$ then $C$ contains either a duplicate column or an OR-minor. The two proofs are symmetric and we give the proof of (1) here. The case of $R(2) = 3$ can be verified by exhaustion.

For the inductive case, consider a $k$-coloring $C$ with more than $R(k)$ rows. We assume that $C$ has no OR-minors, and will show that there must be a duplicate row. By the pigeonhole principle, there must be some color (by symmetry, color $\#k$) which appears more than $R(k)/k = R(k-1)$ times in the first column. The properties we seek are invariant under permuting rows and columns, so permute the rows and columns so that the north-west corner is colored $\#k$, and the instances of color $\#k$ in the first row and first column are contiguous.

Since $C$ contains no OR-minor, we have that $C$ can be partitioned into four quadrants, NW, NE, SE, SW:

$$C = \left[\begin{array}{c|c} \mathsf{NW} & \mathsf{NE} \\ \hline \mathsf{SW} & \mathsf{SE} \end{array}\right],$$

where NW has more than $R(k-1)$ rows, NW contains only color $\#k$, and NE and SW contain only colors $\{1, \ldots, k-1\}$. Thus, NE is a $(k-1)$-coloring with more than $R(k-1)$ rows and no OR-minors. As such it contains duplicate rows. When augmented to the left with identical sequences of $k$'s, we obtain corresponding duplicate rows in $C$, as desired. □

We proved the existence of such an $R$ with $R(k) = \Theta(k!)$, which suffices for our purposes but which may or may not be optimal. We can obtain a lower bound of $2^{k-1}$ on the optimal value of $R(k)$, by considering the following recursively-defined $k$-colorings of a $2^{k-1} \times 2^{k-1}$ grid:

$$C_1 = [1]; \qquad C_k = \left[\begin{array}{c|c} U_k & C_{k-1} \\ \hline C_{k-1} & U_k \end{array}\right].$$

Here $U_k$ denotes the $2^{k-2} \times 2^{k-2}$ grid filled uniformly with color $k$. The colorings $\{C_k\}_k$ avoid duplicate rows, duplicate columns, and OR-minors. We conjecture that $R(k) = 2^{k-1}$ is the optimal value for $R$ as in the lemma statement.

### 5.3 Normal Form

Let $T$ be a 2-dimensional table with row- and column-labels $A$ and $B$, respectively. Define an equivalence relation, where $a \approx_A a'$ if for all $b \in B$, we have $T[a, b] = T[a', b]$ — that is, $a \approx_A a'$ if rows $a$ and $a'$ of $T$ are identical. We define an equivalence relation $\approx_B$ analogously. Finally, let $[a]_A$ and $[b]_B$ denote equivalence classes under these relations, respectively.

**Definition 5.3.** *Let $T$, $A$, and $B$ be as above. Let $e_1^A, \ldots, e_m^A$ denote the distinct equivalence classes of $\approx_A$, and let $e_1^B, \ldots, e_n^B$ denote the distinct equivalence classes of $\approx_B$.*

*We define* $\mathsf{trim}(i, j, T)$ *for $(i, j) \in A \times B$ to denote a tuple $(i', j', T')$, where:*

1. *$T'$ is a table with row-labels $A' = \{1, \ldots, m\}$ and column-labels $B' = \{1, \ldots, n\}$. For each $i^*, j^*$, we have $T'[i^*, j^*] = T[a, b]$, where $a$ is any representative of $e^A_{i^*}$ and $b$ is any representative of $e^B_{j^*}$.*
2. *$e^A_{i'} = [i]_A$. That is, $i'$ is the index of $i$'s equivalence class.*
3. *$e^B_{j'} = [j]_B$. That is, $j'$ is the index of $j$'s equivalence class.*

By item (1) we see that $T'$ has no duplicate columns or rows. Essentially, $\mathsf{trim}$ removes duplicate rows/columns and re-normalizes the row/column labels. Furthermore, the mapping $i \mapsto i'$ does not depend on $j$, the mapping $j \mapsto j'$ does not depend on $i$, and the mapping $T \mapsto T'$ does not depend on $i$ or $j$.

**Definition 5.4.** *Let $T$ be a 2-dimensional table with row- and column-labels $A$ and $B$, respectively, and whose entries are states of a DFF $\mathcal{F}$. Then $\mathsf{explode}(T)$ is a 2-party SFE with input domain $(A \times X) \times (B \times Y)$. On input $(a, x)$ from Alice and $(b, y)$ from Bob, the output of $\mathsf{explode}(T)$ is $f_A(T[a, b], x, y)$ for Alice and $f_B(T[a, b], x, y)$ for Bob.*

**Normal form $\widehat{\mathcal{F}}$.** Let $\mathcal{F} = (Q, X, Y, \delta, f_A, f_B, q_0)$ be a DFF. Then define $\widehat{\mathcal{F}}$ to be a functionality given by Figure 2. Note that we define $\widehat{\mathcal{F}}$ without explicitly considering whether it is a DFF (that is, whether it has a finite number of states). Whether $\widehat{\mathcal{F}}$ has a finite number of states depends on $\mathcal{F}$ in a way that will be established later Lemma 5.7.

Maintain internal state $(a, b, C)$, initialized to $a = b = 1$ and $C = [q_0]$ (that is, a $1 \times 1$ matrix), where $q_0$ is the start state of $\mathcal{F}$.

With internal state $(a, b, C)$, and on input $x \in Y$ from Alice and $y \in Y$ from Bob:

1. Give output $f_A(C[a, b], x, y)$ to Alice and $f_B(C[a, b], x, y)$ to Bob.
2. Set $A' \times B' = \mathsf{rectangle}(\mathsf{explode}(C), (a, x), (b, y))$. Write $A'$ and $B'$ in some canonical ordering $A' = \{(a'_1, x'_1), \ldots, (a'_m, x'_m)\}$ and $B' = \{(b'_1, y'_1), \ldots (b'_n, y'_n)\}$. (Recall that inputs to $\mathsf{explode}(C)$ are tuples of this form.)
3. Define an $m \times n$ table $C'$ via $C'[i, j] := \delta(C[a'_i, b'_j], x'_i, y'_j)$.
4. Set $a' := \mathsf{indexof}((a, x), A')$ and $b' := \mathsf{indexof}((b, y), B')$, where $\mathsf{indexof}(s, S = \{s_1, \ldots, s_n\})$ denotes the value $i$ such that $s_i = s$.
5. Set $(a, b, C) := \mathsf{trim}(a', b', C')$.

Figure 2: Functionality $\widehat{\mathcal{F}}$: the "normal-form" representation of $\mathcal{F}$.

**Lemma 5.5.** *Let $\mathcal{F}$ and $\widehat{\mathcal{F}}$ be as above, and let $\delta^{(k)}$ denote the function $\delta^{(k)}(\boldsymbol{x}, \boldsymbol{y}) = \delta(q_0, \boldsymbol{x}, \boldsymbol{y})$. Suppose that $\mathcal{F}$ is not passive-complete. Then, after reading inputs $\boldsymbol{x}$ and $\boldsymbol{y}$ (with $|\boldsymbol{x}| = |\boldsymbol{y}|$), $\widehat{\mathcal{F}}$ is in state*

$$(a, b, C) = \mathsf{trim}(\boldsymbol{x}, \boldsymbol{y}, \mathsf{restrict}(\delta^{(k)}, \mathsf{rectangle}(\mathcal{F}^{(k)}, \boldsymbol{x}, \boldsymbol{y}))).$$

*It then follows that $\mathcal{F}$ and $\widehat{\mathcal{F}}$ have identical external behavior (since the above implies that $C[a, b] = \delta^{(k)}(\boldsymbol{x}, \boldsymbol{y})$, and $\widehat{\mathcal{F}}$ gives outputs matching those of state $C[a, b]$ in $\mathcal{F}$).*

*Proof.* The claims are true when $\boldsymbol{x}$ and $\boldsymbol{y}$ are empty sequences. Suppose $\widehat{\mathcal{F}}$ is in state $(a, b, C)$ after receiving inputs $(\boldsymbol{x}, \boldsymbol{y})$, with $|\boldsymbol{x}| = |\boldsymbol{y}| = k$. Suppose that $\widehat{\mathcal{F}}$ receives inputs $(x, y)$ at this point;

we will prove the claims with respect to $\boldsymbol{x}' = \boldsymbol{x}\|x$, $\boldsymbol{y}' = \boldsymbol{y}\|y$. Denote the rows & columns of $C$ as $A \times B$.

First, we prove the desired claims without the call to $\mathsf{trim}$, for a variant of $\widehat{\mathcal{F}}$ that does not call $\mathsf{trim}$. It is straight-forward to verify that it makes no difference to the end result to "postpone" all $\mathsf{trim}$ steps taken by $\widehat{\mathcal{F}}$ until the last step, at which point they are clearly idempotent.

By the inductive hypothesis, we have an isomorphism between $\mathsf{restrict}(\delta^{(k)}, \mathsf{rectangle}(\mathcal{F}^{(k)}, \boldsymbol{x}, \boldsymbol{y}))$ and $C$. Thus, we freely identify $A \times B$ with $\mathsf{rectangle}(\mathcal{F}^{(k)}, \boldsymbol{x}, \boldsymbol{y})$, where $a$ with is identified with $\boldsymbol{x}$, and $b$ is identified with $\boldsymbol{y}$.

Since $A \times B = \mathsf{rectangle}(\mathcal{F}^{(k)}, \boldsymbol{x}, \boldsymbol{y})$, and $\mathcal{F}^{(k)}$ has no OR-minor (recall we assume that $\mathcal{F}$ is not passive-complete), we have that $\mathcal{F}^{(k)}$ is *basic* on $A \times B$ (Lemma 2.1). As such, for any $\boldsymbol{x}' \in A \times X$ and $\boldsymbol{y}' \in B \times Y$ we have that

$$\mathsf{rectangle}(\mathsf{restrict}(\mathcal{F}^{(k+1)}, A \times X, B \times Y), \boldsymbol{x}', \boldsymbol{y}') = \mathsf{rectangle}(\mathcal{F}^{(k+1)}, \boldsymbol{x}', \boldsymbol{y}').$$

That is, within this domain of inputs, a party's input can influence the other's output only in the $k+1$ round.

$\mathsf{explode}(C)$ is an SFE whose inputs are then $(A \times X) \times (B \times Y)$ — which we associate with input sequences of length $k+1$ for Alice & Bob, respectively — and whose output is the corresponding output of $\mathcal{F}$ in the $(k+1)$-th round *only*. But again, when restricted to input domain $A \times B$, the first $k$ rounds of output are *basic*, so

$$\mathsf{rectangle}(\mathsf{explode}(C), \boldsymbol{x}', \boldsymbol{y}') = \mathsf{rectangle}(\mathsf{restrict}(\mathcal{F}^{(k+1)}, A \times X, B \times Y), \boldsymbol{x}', \boldsymbol{y}').$$

Putting things together, from lines 3–4 of $\widehat{\mathcal{F}}$ it follows that $C'$ is exactly $\mathsf{restrict}(\delta^{(k+1)}, \mathsf{rectangle}(\mathcal{F}^{(k+1)}, \boldsymbol{x}', \boldsymbol{y}'))$, $a'$ is identified with $\boldsymbol{x}'$, and $b'$ is identified with $\boldsymbol{y}'$, as desired. □

**Lemma 5.6.** *If $\mathcal{F}$ is not passive-complete, then while interacting with $\widehat{\mathcal{F}}$, Alice has no uncertainty about $(a, C)$ and Bob has no uncertainty about $(b, C)$, where $(a, b, C)$ is the internal state of $\widehat{\mathcal{F}}$.*

*Proof.* The claim is true for the initial configuration of $\widehat{\mathcal{F}}$. In round $k$, both parties inductively know $C$ (and hence $\mathsf{explode}(C)$) from round $k-1$. When $\mathcal{F}$ is not passive-complete, then each $\mathsf{explode}(C)$ contains no OR-minor. Hence, after giving inputs $x$ and $y$ respectively, and receiving their outputs (computed from $\mathsf{explode}(C)$), each party can deduce $R = \mathsf{rectangle}(\mathsf{explode}(C), x, y)$. In $\widehat{\mathcal{F}}$, the value $C$ is updated based only on this common information $R$. The value $a$ is updated based only on $R$ and $x$; the value $b$ is updated based only on $R$ and $y$. Each party thus has enough information to update the values required for the lemma. □

**Lemma 5.7.** *Let $\mathcal{F}$ be a DFF. If $\mathcal{F}$ is not passive-complete, then in $\widehat{\mathcal{F}}$ the internal variable $C$ is bounded in size by a constant that depends only on $\mathcal{F}$. Thus $\widehat{\mathcal{F}}$ has a finite number of states (at most $R(n)^2 \cdot n^{R(n)^2}$).*

*Proof.* It follows from the definition of $\widehat{\mathcal{F}}$ that, in every reachable state $(a, b, C)$, the table $C$ has no duplicate rows or columns. We will show that $C$ also contains no OR-minor. Then it will follow from Lemma 5.2 that $C$ has dimensions at most $R(n) \times R(n)$, where $n$ is the number of states in $\mathcal{F}$. Since $a$ and $b$ are row and column indexes into $C$, there are at most $R(n)^2 n^{R(n)^2}$ states in $\widehat{\mathcal{F}}$.

Without loss of generality, assume that $\mathcal{F}$ contains no redundant states. Suppose for contradiction that $C$ contains an OR-minor $(a_0, a_1, b_0, b_1)$, so that $C(a_i, b_j) = r_{i \vee j}$ for distinct states $r_0$ and $r_1$. Let $\boldsymbol{x}^*, \boldsymbol{y}^*$ be a distinguishing sequence for states $r_0$ and $r_1$, with $|\boldsymbol{x}^*| = |\boldsymbol{y}^*| = \ell$.

From Lemma 5.5, $C$ is a submatrix of $\delta^{(k)}$ for some $k$, so there exist input sequences $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{y}_0, \boldsymbol{y}_1$ with $\delta^{(k)}(\boldsymbol{x}_i, \boldsymbol{y}_j) = C[a_i, b_j]$. Furthermore, $\{\boldsymbol{x}_0, \boldsymbol{x}_1\} \times \{\boldsymbol{y}_0, \boldsymbol{y}_1\} \subseteq \mathsf{rectangle}(\mathcal{F}^{(k)}, \boldsymbol{x}_i, \boldsymbol{y}_j)$, and so (since we are assuming that $\mathcal{F}$ is not passive-complete) $\mathcal{F}^{(k)}$ is *basic* restricted to $\{\boldsymbol{x}_0, \boldsymbol{x}_1\} \times \{\boldsymbol{y}_0, \boldsymbol{y}_1\}$.

But then $(\boldsymbol{x}_0\|\boldsymbol{x}^*, \boldsymbol{x}_1\|\boldsymbol{x}^*, \boldsymbol{y}_0\|\boldsymbol{y}^*, \boldsymbol{y}_1\|\boldsymbol{y}^*)$ is an OR-minor in $\mathcal{F}^{(k+\ell)}$, contradicting our assumption that $\mathcal{F}$ is not passive-complete. The reasoning is exactly the same as in the proof of Lemma 3.1. Importantly, Alice's input does not influence Bob's output (and vice-versa) for the first $k$ rounds, and at least one party's total output depends only on whether $r_0$ or $r_1$ was reached in round $k$. □

### 5.4 Deciding Passive-Triviality

The following theorem and its corollary provide total decision procedures for determining whether a given DFF is passive-trivial.

**Theorem 5.8.** *Let $\mathcal{F}$ be a DFF and $\widehat{\mathcal{F}}$ be as above. Suppose $\mathcal{F}$ is not passive-complete. Then $\mathcal{F}$ is passive-trivial if and only if, for every reachable state $(a, b, C)$ in $\widehat{\mathcal{F}}$, $\mathsf{explode}(C)$ is decomposable.*

*Proof.* ($\Rightarrow$) Let $(a, b, C)$ be a reachable state in $\widehat{\mathcal{F}}$. Then from Lemma 5.5 we have that $C$ is a submatrix of $\delta^{(k)}$ for suitable $k$. As such, $\mathsf{explode}(C)$ is a submatrix of $\mathcal{F}^{(k+1)}$. By Proposition 2.6, $\mathcal{F}^{(k+1)}$, and hence all of its submatrices, is decomposable.

($\Leftarrow$) A passive-secure protocol for $\widehat{\mathcal{F}}$ (and hence $\mathcal{F}$, since they have identical external behavior — Lemma 5.5) is the following. Alice maintains $(a, C)$ and Bob maintains $(b, C)$ corresponding to the internal state of $\widehat{\mathcal{F}}$ at all times, as in Lemma 5.6. Inductively they will at each round compute the correct outputs and can thus update these $(a, b, C)$ values. When Alice receives input $x$ and Bob receives input $y$, both parties run a passive-secure protocol for evaluating $\mathsf{explode}(C)$ at inputs $a\|x$ and $b\|y$, respectively. Since $\mathsf{explode}(C)$ is decomposable, it follows that such a secure protocol exists; furthermore, both parties know a common $C$ and can agree upon this protocol. □

**Corollary 5.9.** *Let $\mathcal{F}$ be a DFF with $n$ states, and let $K := R(n)^2 \cdot n^{R(n)^2}$, where $R$ is the function from Lemma 5.2. Then $\mathcal{F}$ is passive-trivial if and only if, for all $k \leq K$, $\mathcal{F}^{(k)}$ is decomposable.*

*Proof.* The forward direction ($\Rightarrow$) follows trivially from Proposition 2.6.

For the other direction, if each of $\{\mathcal{F}^{(k)}\}_{k \leq K}$ is decomposable, then $\mathcal{F}$ is not passive-complete (Theorem 4.1). Then from Lemma 5.7, there are at most $K$ distinct states in $\widehat{\mathcal{F}}$. Any reachable state in $\widehat{\mathcal{F}}$ is therefore reachable by an input sequence of length at most $K - 1$. If state $(a, b, C)$ is reachable by an input sequence of length $k$, then $\mathsf{explode}(C)$ appears as a submatrix of $\mathcal{F}^{(k+1)}$; so if each of $\{\mathcal{F}^{(k)} \mid k \leq K\}$ is decomposable, then so is each $\mathsf{explode}(C)$. Hence, $\mathcal{F}$ is passive-trivial by Theorem 5.8. □

### 5.5 Symmetrization

**Theorem 5.10.** *Let $\mathcal{F}$ be a DFF. If $\mathcal{F}$ is not passive-complete, then there exists a symmetric functionality (that is, one which gives identical output to each party in every round) $\mathcal{G}$ that is isomorphic to $\mathcal{F}$.*

This theorem is a generalization of an analogous theorem of Kraschewski and Müller-Quade [KMQ11] for the special case of SFE.

*Proof.* As described in the discussion after Lemma 2.1, when $\mathcal{F}$ is an **SFE**, we can take $\mathcal{G}$ to be the SFE that gives output $\mathsf{rectangle}(\mathcal{F}, x, y)$ to both parties on input $x$ and $y$.

Now consider when $\mathcal{F}$ is a DFF, and recall its associated $\widehat{\mathcal{F}}$. If $\mathcal{F}$ is not passive-complete, then $\widehat{\mathcal{F}}$ is also a DFF (Lemma 5.7). In state $(a, b, C)$ and on inputs $(x, y)$ in $\widehat{\mathcal{F}}$, the parties are given the output of $\mathsf{explode}(C)$ on inputs $(a, x)$ and $(b, y)$. Define $\mathcal{G}$ to be the same as $\widehat{\mathcal{F}}$, except that both parties are given output $z = \mathsf{rectangle}(\mathsf{explode}(C), (a, x), (b, y))$. Since both parties know $C$,

Alice without loss of generality knows $(a, x)$, and Bob without loss of generality knows $(b, y)$, the output $z$ is enough for both parties to infer the corresponding output of $\widehat{\mathcal{F}}$. Similarly, both parties can infer $z$ from their outputs from $\widehat{\mathcal{F}}$. Thus, $\mathcal{F}$ and $\mathcal{G}$ are isomorphic. □

# 6 Conclusion and Discussion

We presented two new characterizations for cryptographic properties of reactive functionalities, in the setting of computationally unbounded passive adversaries. We highlight several remaining areas of inquiry:

**Active adversaries.** While there is a characterization of triviality of reactive functionalities in the UC model [MPR10], there is no such characterization for the standalone model. There is a characterization for completeness of DFFs as well [MPR10], but it is in the polynomial-time setting. No characterization exists for completeness of reactive functionalities against active adversaries in the information-theoretic setting.

We conjecture that the characterization for active-completeness of DFFs will follow that of the SFE case [KMQ11]. That is, we expect there to be a suitable definition of *redundant inputs* for DFFs so that $\mathcal{F}$ is active-complete if and only if $\mathcal{F}$ is passive-complete after removing all redundant inputs. We note that [MPR10] do in fact define a notion of redundant inputs for DFFs, but only for inputs in the first round. The characterization we seek would require the simultaneous removal of redundant inputs in all states.

A characterization of active (standalone) triviality for DFFs will require a significantly different approach than the one here for passive triviality. We highlight several fundamental aspects of our techniques that seem incompatible with active adversaries:

- We use the fact that $\mathcal{F}$ can be securely realized using $\{\mathcal{F}^{(k)}\}_k$ and vice-versa. Neither direction of this equivalence holds with respect to active security. When emulating the $k$th round of $\mathcal{F}$ using $\mathcal{F}^{(k)}$, we rely on the fact that parties *honestly* maintain their history of inputs and provide them as part of their input to $\mathcal{F}^{(k)}$. To emulate $\mathcal{F}^{(k)}$ using $\mathcal{F}$, the protocol invokes $k$ rounds of $\mathcal{F}$. An active adversary could (depending on $\mathcal{F}$) violate security by adaptively changing its behavior based on the partial information it learns about the other party's input in the first $k-1$ rounds.

- In our proofs we use the fact that certain SFE functionalities appear as a submatrix of some $\mathcal{F}^{(k)}$, to demonstrate their triviality. In the passive security setting, every submatrix of an SFE inherits the triviality of the parent SFE. This property is not true in the active security setting; the characterization of standalone-triviality for SFE [MPR09, KMR09] is not closed under the submatrix relation.

**Randomized functionalities.** Compared to deterministic functionalities, our understanding of randomized functionalities is practically non-existent (an exception is for completeness of certain classes of SFE functionalities; cf. [K00]). For example, there is still no analog of the Beaver-Kushilevitz characterization of passive-trivial SFE [B89, K89] in the randomized case (not even for perfectly-secure protocols).

Our characterization in this work reduces the reactive case to the non-reactive case in some sense. It may be that a similar approach would work even for DFFs with randomized output (even if the actual characterization for SFE is unknown). However, we expect that a randomized *transition function* would lead to complications that are not present in the deterministic case.

## References


[B89] 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.

[C01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In Moni Naor, editor, *FOCS*, pages 136–145. IEEE Computer Society, 2001. Revised version (2005) on Cryptology ePrint Archive: http://eprint.iacr.org/2000/067.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 2001.

[CK91] Benny Chor and Eyal Kushilevitz. A zero-one law for boolean privacy. *SIAM J. Discrete Math.*, 4(1):36–47, 1991.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229. ACM, 1987.

[K91] Joe Kilian. A general completeness theorem for two-party games. In *STOC*, pages 553–560. ACM, 1991.

[K00] Joe Kilian. More general completeness theorems for secure two-party computation. In *STOC*, pages 316–324. ACM, 2000.

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

[KMQ11] Daniel Kraschewski and Jörn Müller-Quade. Completeness theorems with constructive proofs for finite deterministic 2-party functions. In Yuval Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 364–381. Springer, 2011.

[KMR09] Robin Künzler, Jörn Müller-Quade, and Dominik Raub. Secure computability of functions in the IT setting with dishonest majority and applications to long-term security. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 238–255. Springer, 2009.

[K11] Gunnar Kreitz. A zero-one law for secure multi-party computation with ternary outputs. In Yuval Ishai, editor, *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 382–399. Springer, 2011.

[K89] Eyal Kushilevitz. Privacy and communication complexity. In *FOCS*, pages 416–421. IEEE, 1989.

[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*, volume 5444 of *Lecture Notes in Computer Science*, pages 256–273. Springer, 2009.

[MPR10] 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*, volume 6223 of *Lecture Notes in Computer Science*, pages 595–612. Springer, 2010.

[PR08] Manoj Prabhakaran and Mike Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2008.

[R12] Mike Rosulek. Universal composability from essentially any trusted setup. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 406–423. Springer, 2012.

[Y82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *FOCS*, pages 160–164. IEEE, 1982.